Modern Approach to Pedestrian Behaviour Analysis for Real-Time Classification in Autonomous Vehicle Systems

Bhanu Teja Giddaluru
bgiddalu@kent.edu
KS811258615

***Abstract*— Pedestrian awareness remains a significant challenge in computer vision, crucial for enhancing road safety. Current models, such as YOLOv9, are proficient at detecting pedestrians but fall short of providing detailed behavioral data. Our model addresses this gap by being trained on multi-class image datasets to distinguish between 'aware' and 'not aware' pedestrian states during road crossing scenarios. This capability ensures that drivers receive crucial information about pedestrian behavior, potentially reducing accidents. Our research utilized three distinct datasets: the Penn-Fudan Database for Pedestrian Detection and Segmentation, a custom dataset created from real images of team members annotated for awareness levels, and the Pedestrian Intention Estimation Dataset. We employed Robo-flow for image labeling and conducted data augmentations to enrich our limited training data. The YOLOv9 model was fine-tuned on these datasets, resulting in two versions of the model that were extensively tested to enhance accuracy and reliability in real-world applications.**

Keywords*— Pedestrian awareness, computer vision, YOLOv9, real-time detection, pedestrian behavior analysis, data augmentation, Robo-flow.

I. Introduction

Pedestrian awareness remains a critical challenge in computer vision, especially in pedestrian classification, detection, and behavioral analysis. The unpredictability of pedestrian movements often leads to inaccuracies, such as false negatives and false positives, major obstacles in accurately detecting pedestrian behaviors. Our project is designed to analyze and understand these behavior patterns to enable effective communication between pedestrians and drivers, thereby enhancing overall road safety. By leveraging advanced detection models, our initiative seeks not only to identify pedestrians but also to interpret various pedestrian behaviors in real-time, particularly when crossing roads. This capability is crucial for providing drivers with essential information to adjust their driving speed, which is vital for preventing potential accidents.

We are developing models to detect pedestrian behavior with the underlying assumption that pedestrians intend to cross the road. As they cross, we will analyze their behavior and provide alerts to drivers, enabling them to reduce speed. The models classify multiple behaviors. We selected distinct categories for each experiment to explore various approaches for classifying pedestrian awareness and intent, enhancing our understanding of the most effective techniques in real settings. In the data preprocessing stage, we used Robo-flow.

Dataset 1 contains 170 images, dataset 2 contains 300 images, and Dataset 3 includes images captured in rainy conditions to enhance pedestrian detection under such conditions.

After annotating the images and applying data augmentation techniques like horizontal flips, rotations, and hue adjustments, the dataset expanded to approximately 1000 images. The datasets were divided into training, testing, and validation sets. Using the appropriate version of the YOLOv9 model, we downloaded the dataset from Robo-flow's cloud storage using the provided credentials.

Motivation

The motivation for this project stems from the urgent need to improve the dynamics of interaction between vehicles and pedestrians, particularly in urban settings where pedestrian traffic is dense and road accident rates are high. Current technological solutions, capable of identifying the pedestrian presence, fall short in comprehensively analyzing pedestrian intent and behavior, especially under varying environmental conditions such as rain. This gap necessitates the development of more sophisticated models that can adapt to different scenarios and provide reliable data to drivers, thus significantly mitigating risk factors. By incorporating images from rainy conditions and employing robust models, we aim to enhance the model’s performance in ideal conditions and adverse weather scenarios, which are often overlooked in standard pedestrian detection systems.

Furthermore, the use of comprehensive datasets and advanced processing techniques helps in refining the model's accuracy and efficiency, making it a versatile tool for real-world applications. The goal is to create a safer driving environment by enabling more informed and proactive decision-making processes for both autonomous and manually driven vehicles.

The remainder of this paper is organized as follows. We begin in Section 2 providing a discussion of the background, Study Approach contains 3. Architecture/method Section 4, Analysis and Results in Section 5, Section 6 Future work, Discussion in Section 7, and Conclusion Section 8.

II. Related Work

As part of this project, we have done research on many similar works and related papers on Pedestrian Awareness Detection.

Pedestrian Red-Light Violation Detection [1]: This study introduces a YOLOv5-based real-time detection system to identify pedestrians violating red lights. The system deploys an alert mechanism that actively contributes to pedestrian safety at crosswalks. The research highlights the adaptability of YOLOv5 under different lighting conditions, demonstrating its potential to minimize traffic incidents. Advantages: Effective real-time performance. Disadvantage: May face challenges in complex urban environments.

Pedestrian Detection Algorithm [2]: The paper presents a pedestrian detection algorithm that enhances SLIC segmentation and SVM classification, improving accuracy in complex environments. The integration of advanced segmentation techniques results in improved detection performance. Advantages: High precision and robustness in detection. Disadvantage: Potentially slower performance due to preprocessing steps.

Multimodal Interactive Supervised Pedestrian Detection [3]: A system integrating RGB, and thermal images is presented, utilizing YOLOv5 and enhancing detection in various lighting conditions. The paper demonstrates improvements using a multimodal approach, indicating its effectiveness in diverse scenarios. Advantages: Improved accuracy across lighting conditions. Disadvantage: Computational demand could be high.

High-Level Models of Human Behavior [4]: The research surveys the application of psychological and social models in pedestrian behavior analysis for autonomous vehicle interaction. It suggests the need for further research to make these high-level models practically applicable for AV control. Advantages: In-depth review of behavior models. Disadvantage: High-level models are not fully operational in AV algorithms.

Application of Large Space [5]: Utilizing VR, this study investigates pedestrian behaviour with AVs in shared spaces. The findings offer a baseline for future research, indicating potential strategies for enhancing pedestrian safety. Advantages: Groundbreaking use of VR technology. Disadvantage: Sample representation could be limited.

Role of AV Operation [6]: This paper examines the influence of AV operations on pedestrian behavior using a VR setup, finding that AV signals significantly affect pedestrian decisions. Advantages: Valuable insights into pedestrian-AV interaction. Disadvantage: VR may not replicate all aspects of real-world conditions.

Anthropomorphic eHMIs in Roblox [7]: Investigates the impact of eHMIs on road user behavior, using Roblox to simulate and test their effectiveness in communicating the AV's emotional state. Advantages: Innovative communication approach. Disadvantage: The real-world application may vary from the virtual environment.

Pedestrian-Vehicle Conflict Elimination [8]: A cellular automaton model is designed to reduce pedestrian-vehicle conflicts at unsignalized mid-block crosswalks, demonstrating the potential of rule-based systems to improve traffic safety. Advantages: Emphasis on enhancing safety at unsignalized crosswalks. Disadvantage: Specific to unsignalized crosswalks and may not be universally applicable.

AV`s Interaction with Pedestrians [9]: This comprehensive survey explores the complex interaction between pedestrians and AVs, suggesting the need for effective AV communication. Advantages: Extensive review connecting theory and practice. Disadvantage: The generalization of findings to diverse real-world settings is challenging.

YOLOv3-ES Network Enhancement [10]: Enhancements to YOLOv3 are proposed, focusing on improving pedestrian detection, especially in challenging scenarios like overlapping figures. Advantages: Noticeable improvement in detection performance. Disadvantage: The enhancements may increase the computational load.

Pedestrian Behavior Prediction [11]: The paper proposes a novel metric for assessing pedestrian prediction models, emphasizing the need for comprehensive contextual understanding. Advantages: Novel performance metrics proposed. Disadvantage: The focus on a specific model may limit broader applicability.

Smart Graph-based Pedestrian Behavior Recognition [12]: An innovative approach that makes use of graph-based optimization is presented to recognize pedestrian behaviors, demonstrating an increase in accuracy. The classification results are promising, which is an advantage. One potential drawback is that limited testing environments may have an impact on generalization.

360° Video Processing for Pedestrian Behavior [13]: An extensive analysis of pedestrian movement can be obtained through the utilization of a deep learning framework that makes use of 360-degree video captures. This analysis can provide valuable insights for urban planning. The advantages include overcoming the limitations of traditional cameras. One of the disadvantages is that it is limited to particular spatial environments.

TABLE I. SUMMARY OF LITERATURE REVIEW PAPERS

Goal	Merits	Demerits
[1] Enhancing crosswalk safety via real-time red-light violation detection.	• Real-time red-light violation detection • enhances crosswalk safety.	• High computational demands • performance variance in different environments.
[2] Improving pedestrian detection in complex environments.	• Superior pedestrian detection accuracy • efficient preprocessing.	• Preprocessing complexity • accuracy relies on segmentation quality.
[3] Raising accuracy of pedestrian detection with multimodal supervision.	• Varied lighting adaptation • significantly reduced miss rate.	• Significant computational resources needed • data quality-dependent performance.
[4] Deepening understanding of pedestrian-AV interaction behaviors.	• Comprehensive tech and behavior model coverage • psychological insights.	• Theory-practice gap • high-level model quantification is challenging.
[5] Augmenting pedestrian safety in AV-shared spaces.	• Innovative VR safety research • quantitative pedestrian behavior analysis.	• Limited by sample size and VR simulation constraints.
[6] Assessing how AV operations influence pedestrian decisions.	• VR in studying pedestrian-AV interactions. • highlights AV signal importance.	• Virtual settings miss some real-world complexities; narrow AV focus.
[7] Exploring how eHMIs in AVs can change user behaviors.	• Anthropomorphic AV communication • positive psychological impact.	• Real-world applicability uncertain • demographic scope restricted.
[8] Modeling to reduce pedestrian-vehicle conflicts at crosswalks.	• Focus on unsignalized crosswalk safety. • effective simulation use.	• Simulation assumptions limit scope • unsignalized crosswalk specificity.
[9] Surveying pedestrian-AV behavior for future research directions.	• In-depth literature review • identifies key communication strategies.	• Concerns about real-world application • generalization challenges.
[10] Refining pedestrian detection in challenging conditions with YOLOv3-ES.	• Improved accuracy in complex scenarios • addresses detection of small targets.	• May increase computational load. • generalizability is limited.
[12] Identifying pedestrian behaviors with graph-optimization and CNNs.	• High accuracy in behavior classification • enhancements over prior methods.	• Specific dataset applicability • lacks computational efficiency discussion.
[13] Analyzing urban pedestrian behavior using 360° videos.	• Advanced pedestrian behavior data collection • urban space development basis.	• Camera placement affects detection accuracy. • indoor-outdoor filming issues.
[14] Studying complex pedestrian behaviors via VR.	• Detailed behavioral study control • cross-disciplinary research enhancement.	• Access to equipment and technology • gap between VR and reality.
[15] Predicting pedestrian behavior to enhance AV navigation.	• Probabilistic predictive behavior modeling • insights into emergency dynamics.	• High computational demands • dependent on scenario-specific data.
[17] Automating pedestrian detection for surveillance systems.	• Enhanced security via behavioral analysis • broad surveillance applicability.	• High computational cost • environmental complexity impacts performance.
[18] Creating a multi-person tracking framework with GANs.	• Robust multi-person tracking • occlusion and noise handling.	• Complex to implement. • high computational needs and prediction accuracy variance.
[19] Precise human pose estimation with a high-resolution network.	• Consistent high-resolution for accuracy • excels on benchmark datasets.	• Increased computational and memory demands. • complex architecture.
[20] Improving depth map completion with adaptive graph techniques	• Adaptive spatial context modeling • superior multi-modal integration.	• Computational complexity • limited discussion on generalization.
[22] Testing real-time object detection for RoboCup SSL.	• Notable accuracy boost • maintains real-time detection speed.	• Inconsistent inference speed • detection accuracy varies with object size.
[23] Developing a real-time object detection and sizing system.	• Precise object size measurement • effective real-time system.	• Camera angle sensitivity • hardware scalability limitations.
[24] Refining YOLOv3 object detection with edge detection.	• Improved boundary box precision • utilizes edge detection effectively.	• Challenges detecting sharp or noisy objects. • edge detection algorithm reliance

CivilEvac for Pedestrian Behavior Study [14]: Building a VR Tool to Study Pedestrian Movement and Choice Behavior CivilEvac, a VR tool for studying pedestrian movement and choice behavior in multi-level buildings, is introduced in this paper. CivilEvac simulates a building so participants can navigate while their movements and vision fields are recorded for analysis. The study uses VR to collect detailed pedestrian behavior data, overcoming traditional methods' limitations. It shows CivilEvac's ability to capture realistic pedestrian behaviors and decisions, making it useful for urban design, emergency evacuation planning, and spatial navigation research. Advantages: Detailed, controlled data collection. Disadvantage: VR technology may not capture the full spectrum of human behavior.

Multimodal Hybrid Pedestrian Model [15]: A hybrid automaton model for predicting pedestrian behavior is validated against real-world scenarios, showcasing its potential to inform AV navigation. Advantages: Addresses multimodality of pedestrian behavior. Disadvantage: High complexity of model implementation.

Pedestrian Traffic Characterization [16]: This empirical study offers insights into pedestrian traffic behavior in emergencies, providing a basis for more realistic modelling. Advantages: Empirical data from evacuation experiments. Disadvantage: Experimental conditions may not reflect all emergency scenarios.

Pedestrian Detection in Surveillance Systems [17]: A YOLO-based model that is suitable for use in security systems is proposed for real-time pedestrian detection and movement analysis for pedestrians. The benefits include a high processing speed and complete accuracy. Disadvantage: The performance may change depending on the complexity of the environment.

Multi-Person Tracking [18]: A Deep Generative Model for Multi-Person Localisation and Tracking uses Generative Adversarial Networks (GANs) for pedestrian detection and a predicted trajectory-based data association scheme for multi-person tracking. Occlusions and noisy detections in crowded environments are addressed by this method. The model predicts short-term and long-term pedestrian trajectories, enabling tracking without appearance features in crowded scenes with frequent occlusions. Advantages: Capable tracking in scenes that are constantly changing. There is a possibility that the computational requirements will be high.

Human Pose Estimation [19]: A human pose estimation system called HRNet is presented to maintain high-resolution representations, outperform existing benchmarks, and estimate human poses. Advantages include Exceptional performance over the benchmark. One potential disadvantage is that it may result in increased computational standards.

Depth Completion Network [20]: The ACMNet network was proposed to improve depth map recovery from sparse data and a single RGB image using graph propagation and a symmetric gated fusion strategy for multi-modal data integration (20). Using graph propagation, it adapts spatial contexts and uses multi-modal data to combine depth and RGB. The method performs well on benchmarks with fewer parameters. Advantages: Effective integration of multi-modal data. Disadvantage: May require extensive parameter tuning for different conditions.

EAPM for Object Detection [21]: In deep object detection training, gradient contribution imbalance is addressed by Example Attribute-Based Prediction Modulation (EAPM). EAPM improves classification and localization balance by defining example attributes based on prediction and ground truth and reweighting prediction errors' contribution to global gradients. This method improves MS COCO object detection performance significantly. Advantages: Improved balance in detection training. Disadvantage: Complexity in the integration with existing frameworks.

RoboCup SSL Object Detection [22]: Empirical Study: RoboCup Small Size League One-stage Object Detection Methods, evaluating MM Detection framework models on a dataset for RoboCup SSL. The YOLOX-tiny model outperformed baseline methods with 58.60% Average Precision (AP) and 37 FPS inference speed. The study shows that one-stage detectors can balance accuracy and inference speed for real-time object detection in fast-paced robot soccer games. Advantages: Balances accuracy and inference speed. Disadvantage: Inference speed may vary with object sizes.

Real-Time Object Detection and Measurement [23]: An Embedded Real-Time Object Detection and Measurement of Size proposes an effective method for real-time object detection and dimension measurement in video streams. It uses OpenCV libraries and clever edge detection, dilation, and erosion algorithms for object identification, morphological operations to close edge gaps, contour finding and sorting, and object dimension measurement. The Raspberry Pi 3 and Raspberry Camera system determined object size with nearly 98% accuracy. Advantages: Cost-effective and real-time capability. Disadvantage: Sensitivity to environmental conditions and camera setup.

Improving YOLOv3 Precision [24]: By integrating edge detection algorithms with YOLOv3, edge detection-based boundary box construction algorithms can improve object detection precision. The method refines boundary boxes around detected objects using edge detection to fix YOLOv3's errors. This method uses pre-trained COCO dataset models and edge detection algorithms to localize objects more accurately than YOLOv3. Advantages: Improved object localization accuracy. Disadvantage: Increased computational complexity may arise.

.

III. Study Approach

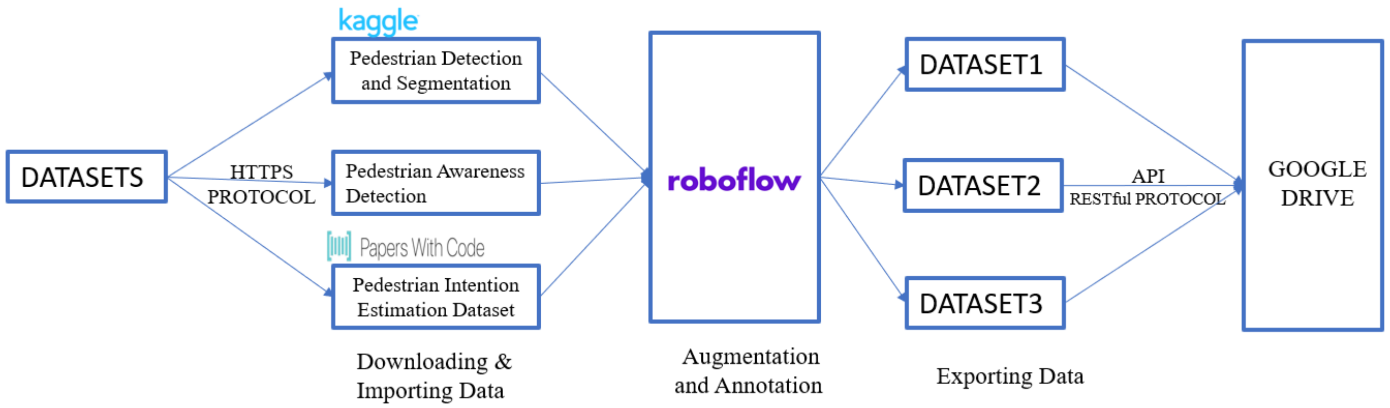
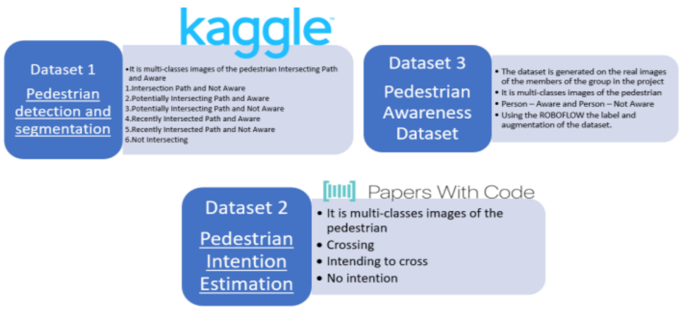

kaggle
Pedestrian Detection and Segmentation
DATASETS
HTTPS PROTOCOL
Pedestrian Awareness Detection
Papers With Code
Pedestrian Intention Estimation Dataset
roboflow
DATASET1
DATASET2
DATASET3
API
RESTful PROTOCOL
GOOGLE DRIVE
Downloading & Importing Data
Augmentation and Annotation
Exporting Data

Fig. 1. Data Processing and Management Flowchart

Figure 1 illustrates the workflow of data processing for a pedestrian detection model. Starting with datasets sourced from Kaggle, Papers with Code, and custom data, it follows the data's journey via HTTPS and RESTful API protocols through Roboflow, where augmentation and annotation occur before the processed datasets are exported to Google Drive for model training and storage.

The project examined multiple datasets to address the complexity of real-world pedestrian behavior analysis. Multiple pedestrians per image, a variety of pedestrian stances, and real-world image quality were chosen for each dataset. Our method stressed the importance of combining multiple data sources to build a robust model for accurate detection and classification.

3.1. Dataset Information



Dataset 1
Pedestrian detection and segmentation

- It is multi-classes images of the pedestrian Intersecting Path and Aware
1. Intersection Path and Not Aware
2. Potentially Intersecting Path and Aware
3. Potentially Intersecting Path and Not Aware
4. Recently Intersected Path and Aware
5. Recently Intersected Path and Not Aware
6. Not Intersecting

Dataset 3
Pedestrian Awareness Dataset

- The dataset is generated on the real images of the members of the group in the project
- It is multi-classes images of the pedestrian
- Person – Aware and Person – Not Aware
- Using the ROBOFLOW the label and augmentation of the dataset.

Dataset 2
Pedestrian Intention Estimation

- It is multi-classes images of the pedestrian
- Crossing
- Intending to cross
- No intention

Fig. 2. Dataset sources

Figure 2 outlines the three datasets used for training a pedestrian detection model. Dataset 1 covers pedestrian detection and segmentation, Dataset 2 focuses on pedestrian intention estimation, and Dataset 3, a custom collection, addresses pedestrian awareness. Each dataset targets specific aspects of pedestrian behavior to improve model accuracy.

Dataset1[25]:https://www.kaggle.com/datasets/psvishnu/pennfudan-database-for-pedestrian-detection-zip

Dataset 2[26]: https://paperswithcode.com/dataset/pie

Dataset 3: Own dataset

3.1.1 Dataset 1:

The initial dataset 1 employed the Penn-Fudan Database (PF) to specifically detect the head orientations of pedestrians to classify their level of awareness. The initial labelling process involved the utilization of specific categories.

- Potentially Intersecting Path and Aware
- Intersecting Path and Aware
- Intersection Path and Not Aware
- Recently Intersected Path and Aware
- Recently Intersected Path and Not Aware
- Potentially Intersecting Path and Not Aware
- Not Intersecting

The classification was determined by analyzing the projected trajectory and orientation of individuals, differentiating pedestrians based on their anticipated and observed interactions with the vehicle's route.

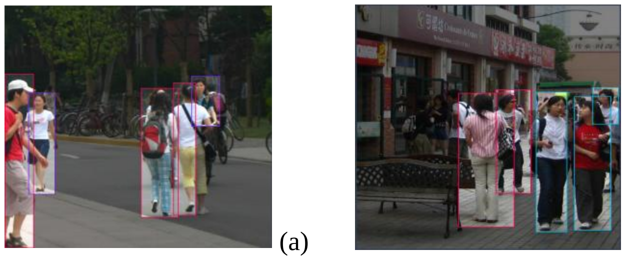


Fig. 3. (a) Red-Intersecting and Not Aware, Purple-Intersecting and Aware, (b) Blue-Potentially Intersecting and Not Aware

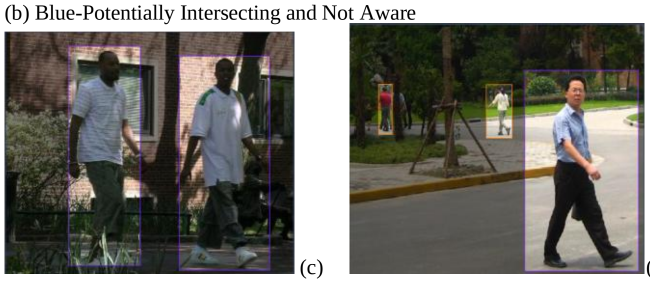


Fig. 4. (c) Purple-Intersecting and Aware (d) Orange-Not Intersecting

The images showcase the classification of pedestrians by an object detection model. Fig. 3 "a" identifies pedestrians as 'Intersecting and Not Aware' (red), 'Intersecting and Aware' (purple), and "b" 'Potentially Intersecting and Not Aware' (blue). Fig. 4 continues with "c" 'Intersecting and Aware' (purple) and "d" 'Not Intersecting' (orange), demonstrating the model's capability to distinguish pedestrian behaviors in real-world scenarios.

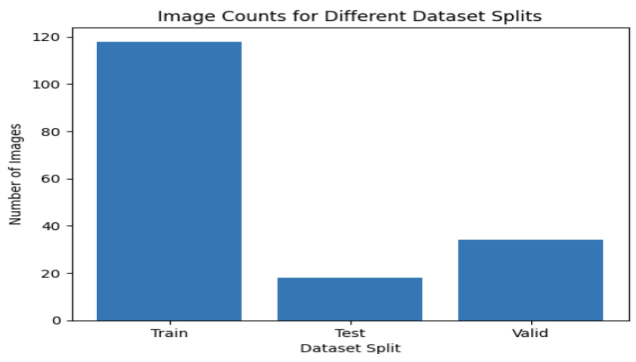


Fig. 5. Dataset 1, splitting of data in train. test, valid

Figure 5, displays a bar chart representing the distribution of images across three different dataset splits: training, testing, and validation. The training set has the highest number of images, significantly more than the testing and validation sets, indicating a heavy emphasis on the training phase to develop the model's accuracy. This distribution is common practice in machine learning to provide a large amount of data for model training while reserving smaller portions for testing and validation to prevent overfitting and ensure the model's generalizability.

3.1.2 Dataset 2:

Utilizing the PIE dataset, this experiment extracted frames from video footage that documented active pedestrian zones in urban environments. The objective was to improve the model's capacity to identify shifts in pedestrian intentions, progressing from having no intention to intending to cross, and ultimately to crossing. This refined categorization facilitates prompt notifications for drivers.

- Intending to cross
- No intention
- Crossing

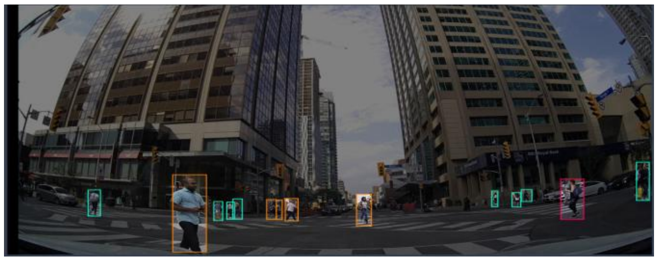


Fig. 6. Red-intending to cross, Orange-crossing, Green-No intention

Figure 6 depicts an urban street scene where pedestrians are classified by their crossing intentions, demonstrated through colour-coded bounding boxes: red for those intending to cross, orange for those actively crossing, and green for those with no intention to cross. This image classification serves as a visual tool for understanding pedestrian behavior patterns in the context of autonomous vehicle systems and urban planning.

Using this approach, we aim to identify the transition from absence of intention to intention, and ultimately, to crossing. This method would effectively notify the driver of any potential hazards.

3.1.3 Dataset 3:

In dataset 3, we generated a bespoke dataset by utilizing authentic images taken by our team members. These images were used to simulate situations where pedestrians are crossing streets. The categorizations were clear and easy to understand:

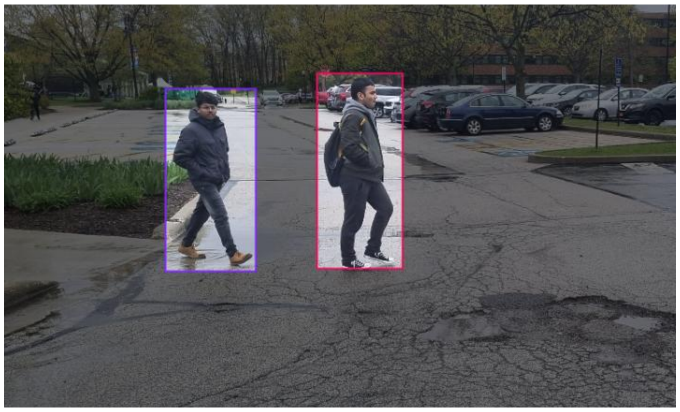


Fig. 7. Blue label – person aware, Red the person is not aware

Figure 7 “location Kent State University, near Library road 1125 Risman Dr, Kent, oh” displays a visual representation from a pedestrian detection dataset, where individuals are marked with colour-coded bounding boxes to indicate their level of awareness. The purple bounding box signifies a 'Person-Aware'—someone attentive to their surroundings—while the red bounding box identifies a 'Person-Not Aware,' indicating the individual may be less alert or oblivious to potential hazards, such as approaching vehicles. This classification is crucial for developing systems that aim to enhance pedestrian safety, particularly in the context of autonomous vehicle technology.

- Person-Aware:

Pedestrians exhibit awareness of the approaching vehicle while crossing the road.

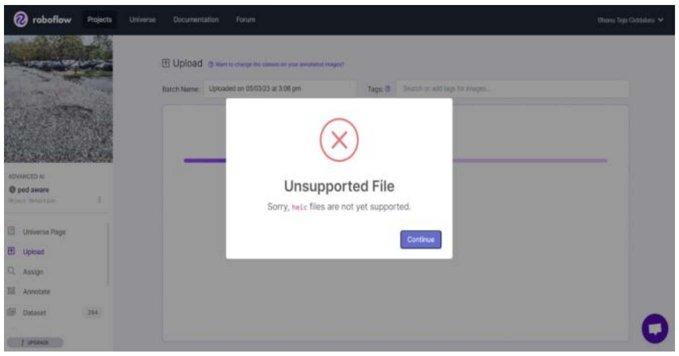
- Person-Not Aware

When a pedestrian crosses a road without looking at a car.

This experiment specifically targeted the positions of pedestrians' heads and eyes to assess their level of awareness, which is crucial for scenarios involving interaction.

3.2. Dataset Preprocessing

Data preprocessing is an essential step in ensuring that datasets are formatted in the most optimal way for training models. The PF dataset images were easily formatted for input into Roboflow, with minimal preprocessing required. On the other hand, the dataset obtained from the PIE dataset consisted of frames extracted from video footage.



Unsupported File

Sorry, heic files are not yet supported.

Fig. 8. Problem with Roboflow

Figure 8 depicts a screenshot from the Roboflow platform showing an error message about an unsupported file type during the upload process. The dialogue box indicates that the file format attempted to be uploaded is not accepted by the system. This suggests that users need to ensure their dataset files conform to the supported formats for successful upload and further processing on Roboflow. These frames were then processed at a rate of two images per second to capture the dynamic activities of pedestrians at intersections. For Experiment 3, the images that were initially taken by team members in HEIC format were changed to JPEG format because Roboflow cannot work with HEIC format. This conversion guarantees that our models are trained on data that is both varied and reflective of the typical input they will encounter in real-world scenarios.

3.3. Dataset Annotation

Annotating data is an essential step in preparing it for efficient model training. We employed Roboflow, a versatile platform that facilitates multiple types of annotation, such as bounding boxes, segmentation masks, and key points. The annotation process involved distinguishing between the states of being 'Person-Aware' and 'Person-Not Aware' in each image, which was determined by analyzing the orientation and visibility of the pedestrian's head and eyes. It is crucial to differentiate between different factors to effectively train our models to accurately identify and forecast pedestrian awareness. This is of utmost importance for the implementation of autonomous vehicle technologies.

3.4. Dataset Augmentations

To improve the resilience of our models and replicate a range of environmental circumstances that they may encounter, we implemented multiple data augmentation techniques on our datasets:

Data augmentations

- horizontal flip
- -15 to +15 rotations
- -25 to +25 hue-25 to +25 hue

Horizontal Flip: This augmentation horizontally reflects the image, aiding the model in generalizing across various pedestrian orientations about the vehicle.

Rotation (-15 to +15 degrees): Incorporating random rotations into the model enhances its robustness in handling images taken from unconventional angles, a frequent occurrence in dynamic urban settings.

Hue Adjustment (-25 to +25): By manipulating the hue, we replicate diverse lighting conditions, which is essential for optimizing the model's performance across different times of the day and varying weather conditions.

These enhancements are intended to equip the model for real-world implementations where circumstances are seldom optimal or uniform.

3.5. Dataset Splitting

For every experiment, we carefully partitioned the datasets into training, validation, and test sets to guarantee a comprehensive assessment of the model's performance:

Dataset 1: Train/ Test Split

- Train 118 images
- Valid 34 images
- Test 18 images

Experiment 1 involved training 118 images, validating with 34 images, and testing with 18 images.

Dataset 2: Train/ Test Split

- Train 210 images
- Valid 60 images
- Test 30 images

Experiment 2 involved training 785 images, validating 73 images, and testing 39 images.

Dataset 3: Train/ Test Split

- Train 785 images
- Valid 73 images
- Test 39 images

Experiment 3 involved training 210 images, validating with 60 images, and testing with 30 images.

This systematic division enables impartial evaluations of the model at various points in the training and refinement process, guaranteeing that the ultimate model is thoroughly optimized and dependable.

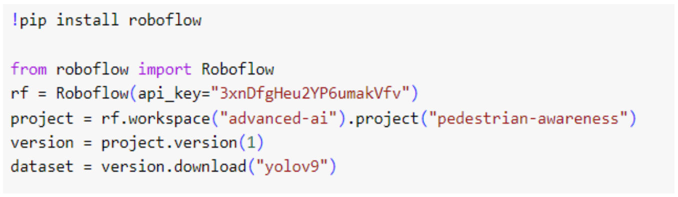
3.6. Dataset Saving and Downloading

Efficient training and reproducibility of results are ensured by properly managing and storing the annotated datasets. By utilizing Roboflow's Code Export feature, we optimized the dataset preparation and integration process with our PyTorch model training environment.

API

The Roboflow API provided here is likely a RESTful API, as it is the most common type of web API and is used for handling the requests and communication between a client and a server over the web. Here is the Python code that is utilized to download and load the datasets directly into our environment: Dataset credentials to download and load the datasets from Roboflow.

Dataset1:



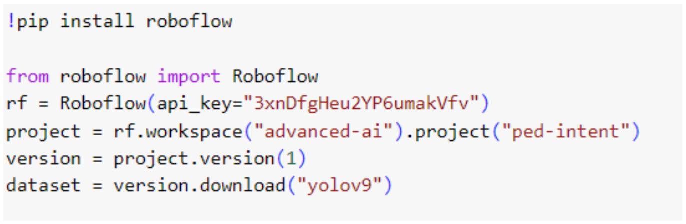
```
!pip install roboflow

from roboflow import Roboflow
rf = Roboflow(api_key="3xnDfgHeu2YP6umakVfv")
project = rf.workspace("advanced-ai").project("pedestrian-awareness")
version = project.version(1)
dataset = version.download("yolov9")
```

Fig. 9. Dataset 1, Roboflow credentials

Figure 9 displays a code snippet for using the Roboflow library to download a specific version of a dataset tailored for the YOLOv9 model. The code shows the installation of Roboflow, importing the library, setting up an API key, selecting a project workspace and a project, choosing a dataset version, and initiating the download process.

Dataset2:



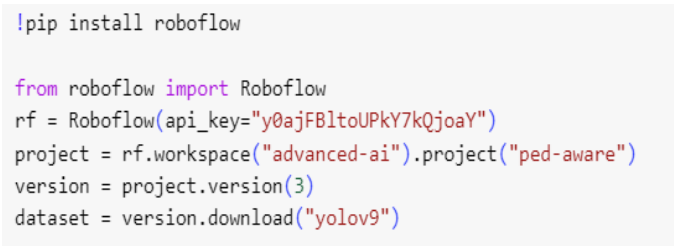
```
!pip install roboflow

from roboflow import Roboflow
rf = Roboflow(api_key="3xnDfgHeu2YP6umakVfv")
project = rf.workspace("advanced-ai").project("ped-intent")
version = project.version(1)
dataset = version.download("yolov9")
```

Fig. 10. Dataset 2, Roboflow credentials

Figure 10 shows a block of Python code for setting up and downloading a dataset using the Roboflow library. It begins with installing Roboflow, then importing and initializing it with an API key, accessing a specific project ('ped-intent'), selecting a version of the dataset, and finally, triggering the download of the dataset prepared for YOLOv9.

Dataset3:



```
!pip install roboflow

from roboflow import Roboflow
rf = Roboflow(api_key="y0ajFBltoUPkY7kQjoaY")
project = rf.workspace("advanced-ai").project("ped-aware")
version = project.version(3)
dataset = version.download("yolov9")
```

Fig. 11. Dataset 3, Roboflow credentials

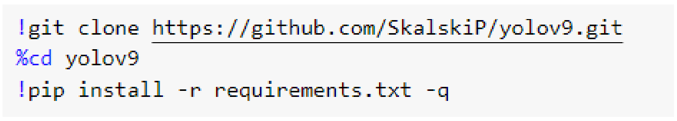
Figure 11 contains a script for acquiring a dataset through Roboflow. It outlines the process to install Roboflow, import the necessary module, authenticate with an API key, and select the "ped-aware" project within a specific workspace. The script then accesses version 3 of the project and initiates a download presumably formatted for the YOLOv9 object detection model.

import os

from google.colab import drive

drive.mount('/content/drive')

Google Drive uses HTTPS for secure computer network communication in Google Colab. HTTPS is HTTP's secure version. It protects client-server data with SSL/TLS encryption, ensuring data integrity and privacy. Google Colab uses this protocol to securely access and transfer files when you mount Google Drive.



```
!git clone https://github.com/SkalskiP/yolov9.git
%cd yolov9
!pip install -r requirements.txt -q
```

Fig. 12. Git hub link to clone yolov9 model

Figure 12 depicts a sequence of command-line instructions for setting up the YOLOv9 model. The commands include cloning the YOLOv9 GitHub repository, changing the directory to the cloned repository, and installing the necessary Python dependencies from a `requirements.txt` file quietly without verbose output.

This configuration enables us to uphold a uniform workflow and guarantees that all team members and future researchers can effortlessly access and utilize the prepared datasets, thereby promoting a collaborative and efficient research environment.

IV. Architecture/Metod

4.1 Architecture

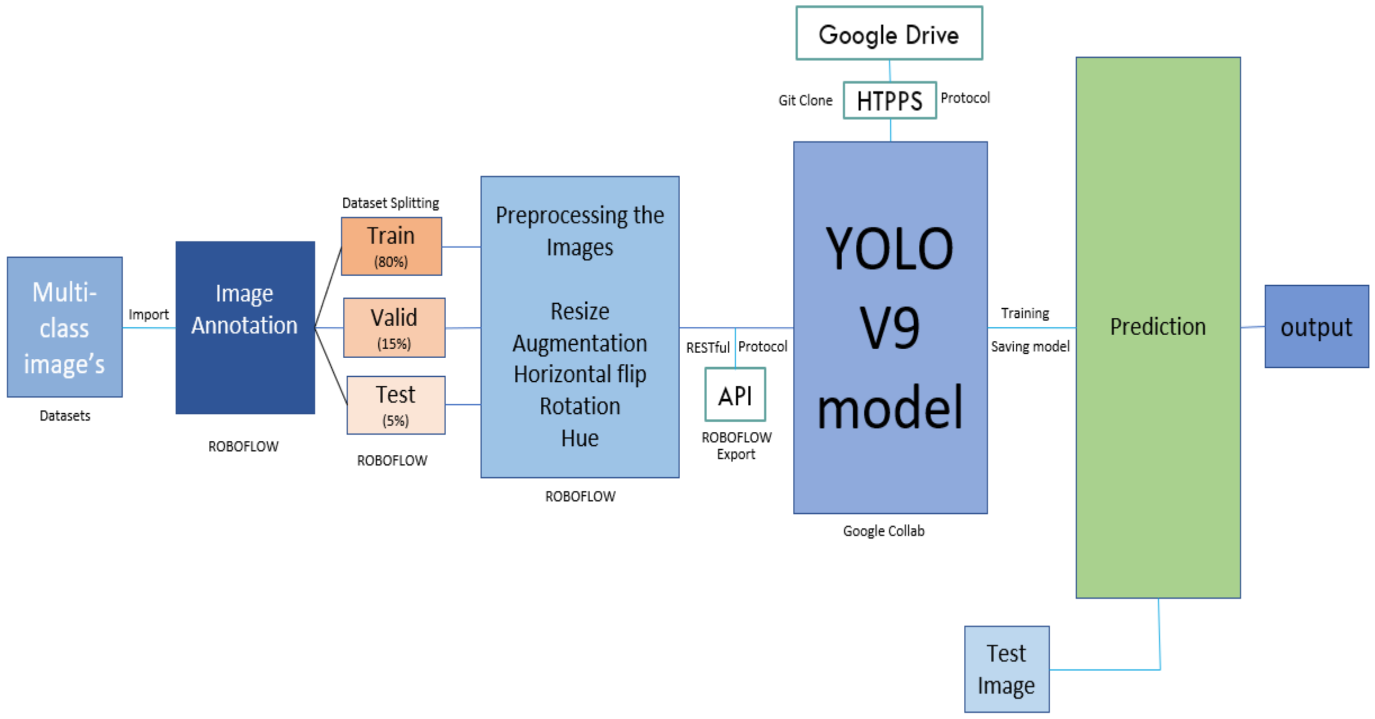


Fig. 13. Overall architecture

Figure 13 illustrates a flowchart of the machine learning pipeline for a YOLOv9 object detection model. It starts with importing multi-class images and performing image annotation in Roboflow, followed by dataset splitting and preprocessing steps like resizing and augmentation. The model is then trained in Google Colab, using Google Drive for storage and a RESTful API for data handling. The final output is the prediction made on a test image, culminating in the end-to-end workflow from data preparation to model inference.

Steps involved in Over all Architecture

1 Datasets are sourced from Kaggle, paper with codes, Own dataset and imported into Roboflow for preprocessing.

2 Roboflow is used for annotating images and splitting the dataset into training, validation, and testing sets.

3 The preprocessed data is then exported to Google Drive via the RESTful API protocol for secure transfer.

4 The YOLOv9 GitHub repository is cloned using HTTPS for added security.

5 Google Colab facilitates the training of the YOLOv9 model on the prepared datasets.

6 Post-training, the model performs object detection predictions on new test images.

7 The output from the model's predictions provides insights into the model's performance.

The project's architecture adheres to a systematic process that commences with gathering datasets from various sources such as Kaggle, Papers with Code, and our own custom dataset. These datasets are subsequently imported into Roboflow. At Roboflow, we carry out image annotation and divide the data into specific training, validation, and test sets.

The curated datasets are securely transmitted to Google Drive utilizing the RESTful API. The YOLOv9 model's codebase is securely replicated from GitHub using the HTTPS protocol, and the training is performed on Google Colab, utilizing its cloud-based computing resources. After undergoing training, YOLOv9 utilizes its object detection abilities to process new test images. The resulting outputs are then examined to assess the model's performance in detecting objects.

4.2 Model selection

TABLE II. MODEL COMPARISON

Model	Architecture	Speed (FPS)	Accuracy	Memory Usage	**Real-Time**
Yolov9	Cnn-based	Very High	High	High	yes
Yolov8	Cnn-based	High	High	Moderate	yes
Faster R-Cnn	Cnn-based	Low	Very High	low	no
SSD	Cnn-based	Moderate	Moderate	Moderate	yes
HOG+ SVM	Feature extraction	Low	Low	Low	No
Retina net	Cnn-based	Moderate	High	Moderate	yes
Efficient Det	Cnn-based	High	Very High	Moderate	yes
Transformer	Attention based	Moderate	High	High	yes

Table 2 displays a comparative analysis of different object detection models, with a specific emphasis on YOLOv9.

This model is renowned for its convolutional neural network (CNN) architecture, which provides exceptional speed and accuracy while utilizing a substantial amount of memory. As a result, it is highly suitable for real-time applications. YOLOv9 is distinguished by its ability to process data in real time, its utilization of a single neural network architecture, its use of spatially separated bounding boxes for object detection, and its accurate and efficient classification of objects through class probability predictions. These features contribute to its exceptional performance in swiftly and accurately detecting and classifying objects.

The table 2, functions as a valuable resource for comprehending the trade-offs among various models such as YOLOv8, Faster R-CNN, and others. YOLOv9, for example, achieves a favourable trade-off between speed and accuracy, surpassing other models that may demand longer processing time or greater memory resources, such as Faster R-CNN, or deliver lower accuracy, like HOG + SVM. YOLOv9 is particularly advantageous in situations where fast and dependable object detection is essential, such as in autonomous vehicle navigation or real-time surveillance.

4.3 Model

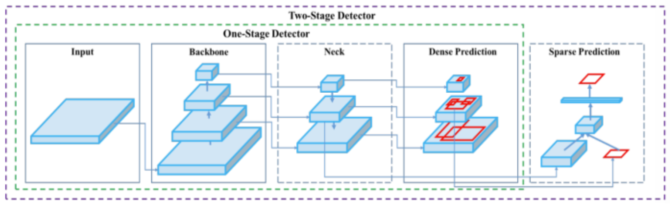


Fig. 14. Yolov9 model

Figure 14, represents the architectural structure of a YOLOv9 model, which is specifically designed as a two-phase object detection system. During the initial stage, referred to as the one-stage detector, the input image is processed by a backbone network that is responsible for extracting features. Next, a neural network dedicated to the neck region processes these features and readies them for prediction. This architectural component is specifically designed to maximize speed and efficiency, making it well-suited for real-time applications that demand rapid processing, such as detecting pedestrians in autonomous driving situations.

The model's second stage is a two-stage detector with dense and sparse prediction. A large number of bounding boxes or predictions is used in dense prediction to detect multiple objects in an input image. Sparse prediction refines and prioritizes the initial predictions. The two-stage approach allows the YOLOv9 model to maintain accuracy while operating in real-time, balancing speed and precision in object detection assignments.

4.3.1. Environment Setup

Repository Cloning:

- Input: GitHub repository URL for YOLOv9.

!git clone https://github.com/SkalskiP/yolov9.git

The YOLOv9 repository, which is the official source for the YOLOv9 model, was cloned from GitHub. It includes all the necessary source code for running and training the model.

- Output: Local copy of the YOLOv9 source code.
- Interface: Git command line or desktop interface used to clone the repository.

Dependency Installation:

- Input: List of dependencies in the requirements.txt file.

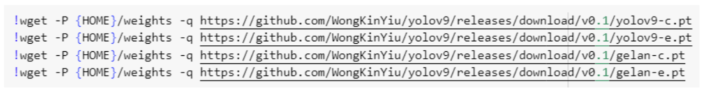
```
!pip install -r requirements.txt -q
```

The model's compatibility and functionality were ensured by installing all the required Python dependencies specified in the requirements.txt file.

- Output: Installed Python packages ready for use.
- Interface: Python's package manager, pip, which reads the requirements.txt file and installs packages.

Weights Acquisition:

- Input: URLs to pre-trained weight files.



```
!wget -P {HOME}/weights -q https://github.com/WongKinYiu/yolov9/releases/download/v0.1/yolov9-c.pt
!wget -P {HOME}/weights -q https://github.com/WongKinYiu/yolov9/releases/download/v0.1/yolov9-e.pt
!wget -P {HOME}/weights -q https://github.com/WongKinYiu/yolov9/releases/download/v0.1/gelan-c.pt
!wget -P {HOME}/weights -q https://github.com/WongKinYiu/yolov9/releases/download/v0.1/gelan-e.pt
```

Fig. 15. Wget Function to get a predefined weight of the Yolov9 model

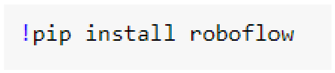
Figure 15 shows commands for downloading YOLOv9 model weights from GitHub. Before training on specific datasets, these pre-trained weights are essential for model initialization. Each line of code retrieves a specific weight version using `wget` and saves it to the `HOME` directory.

Pre-existing weights were obtained, providing a foundation for training and expediting the model's convergence on the designated pedestrian datasets.

- Output: Downloaded weight files on the local machine.
- Interface: Download command or script, typically using wget

4.3.2. Training

Dataset Integration:



```
!pip install roboflow
```

Fig. 16. Installation of roboflow

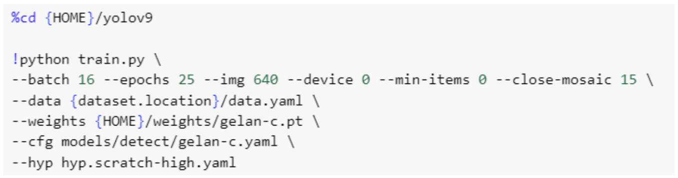
Figure 16 is a simple command-line instruction to install the Roboflow package using Python's package manager, pip. This step is typically part of the initial setup in machine learning projects that use Roboflow's services for data annotation and preprocessing.

- Input: Pedestrian dataset from Roboflow.

The Roboflow platform was used to access the 'pedestrian' dataset. The dataset was adapted to align with the YOLOv9 format.

- Output: Dataset formatted for YOLOv9 (images and annotations).
- Interface: Roboflow's API used to download and format the dataset.

Model Configuration:



```
%cd {HOME}/yolov9

!python train.py \
--batch 16 --epochs 25 --img 640 --device 0 --min-items 0 --close-mosaic 15 \
--data {dataset.location}/data.yaml \
--weights {HOME}/weights/gelan-c.pt \
--cfg models/detect/gelan-c.yaml \
--hyp hyp.scratch-high.yaml
```

Fig. 17. Traning the model, with 25 epochs

The YOLOv9 model training command sequence is shown in Figure 17. Change the directory to the YOLOv9 folder, run the training script with batch size, number of epochs, image size, and device settings, and specify data configuration, weights, model configuration, and hyperparameter file paths.

- Input: Pre-trained weights, configuration file, hyperparameters.

The training process started using the pre-downloaded weights. A batch size of 16 was used, and the training ran for 25 epochs. The training process was customized using a configuration file and hyperparameters to maximize performance in detection tasks.

- Output: Trained model with updated weights.
- Interface: YOLOv9's training script which takes configuration and hyperparameter settings.

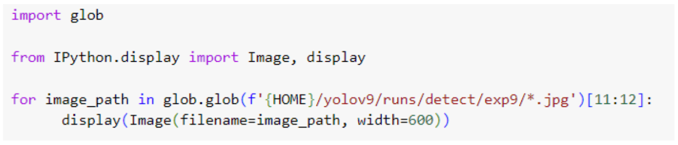
Monitoring Training Progress:

- Input: Real-time data during model training (loss, accuracy).

Throughout the training process, important metrics were continuously monitored and recorded. This involved monitoring the model's loss and accuracy throughout the epochs.

- Output: Logs and visualization of training progress.
- Interface: Machine learning experiment tracking tools or built-in logging within YOLOv9 training scripts.

4.3.3. Model Validation and Testing



```
import glob

from IPython.display import Image, display

for image_path in glob.glob(f'{HOME}/yolov9/runs/detect/exp9/*.jpg')[11:12]:
      display(Image(filename=image_path, width=600))
```

Fig. 18. Digit dataset multiclass bar plot

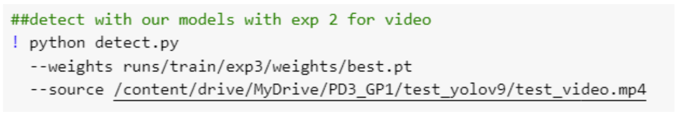
Figure 18 shows Python code used for model validation and testing, specifically for displaying images from a YOLOv9 model's output directory. The script uses the `glob` module to retrieve image file paths and the IPython display library to show the images, likely to visualize the model's predictions

Validation Set:

- Input: A separate set of labelled images (validation set).

A distinct validation set, consisting of images from all three datasets, was employed to optimize the model parameters and mitigate overfitting, guaranteeing the model's ability to effectively apply to new data.

Testing and Performance Metrics:



```
##detect with our models with exp 2 for video
! python detect.py
  --weights runs/train/exp3/weights/best.pt
  --source /content/drive/MyDrive/PD3_GP1/test_yolov9/test_video.mp4
```

Fig. 19. Model Prediction using mp4

Figure 19 depicts a command-line script for running object detection on a video file using the YOLOv9 model. The code specifies the use of the best-performing weights from a previous training session and sets the video file's path as the input source. The process is part of the model's inference phase, where it detects objects within the video.

- Input: Independent test set of labelled images.

The model underwent thorough testing to assess its performance, using a separate test set for unbiased evaluation. The model's predictive power and accuracy in real-world scenarios were evaluated by analyzing key metrics such as precision, recall, and the confusion matrix.

4.4 Protocols

The protocols utilized in the machine learning pipeline. HTTPS, which stands for HyperText Transfer Protocol Secure, is used to establish secure communication over the internet. It ensures that any data exchanged between the user's computer and the cloud platform (Google Colab) is encrypted and protected from interception. It is essential to follow this protocol when cloning the YOLOv9 repository from GitHub and downloading datasets, as ensuring data integrity and security is of utmost importance.

The RESTful API protocol, which is widely used for web services, enables seamless interaction with the Roboflow platform. The dataset can be easily integrated into our project through seamless integration, which provides programmatic access to Roboflow's services for downloading and uploading datasets. Additionally, it allows us to perform operations such as annotation and augmentation. These protocols collectively guarantee the secure, efficient, and dependable completion of tasks within our cloud-based development environment, ultimately simplifying the workflow for the development of the pedestrian detection model.

4.5 Pseudocode:

Pseudo code for pedestrian behaviour analysis using YOLOv9:

```
Begin:
1. Import necessary libraries (NumPy, OpenCV, PyTorch, Roboflow, etc.)

2. Set up configuration paths for datasets and model weights:
    dataset_path = "/path/to/datasets"
    weights_path = "/path/to/yolov9/weights"

3. Data Acquisition and Management:
    - Load diverse datasets for comprehensive training (e.g., from Kaggle, custom datasets)
    - Combine and shuffle datasets to ensure variety
    - Split datasets into training (80%), validation (15%), and testing (5%)

4. Preprocessing and Data Augmentation:
    - Perform image resizing for uniformity
    - Apply augmentations like horizontal flipping, rotations, and hue adjustments
    - Annotate images with classes and bounding boxes using Roboflow

5. Model Initialization:
    - Instantiate the YOLOv9 model with pre-trained weights for transfer learning
    - Define training parameters (input resolution, epochs, learning rate)
6. Training Loop:
    - Use mini-batch gradient descent to iteratively train the model
    - Calculate and log loss at each iteration
    - Employ validation set for hyperparameter tuning and avoid overfitting

7. Model Evaluation:
    - Assess the model with metrics such as accuracy, precision, recall, and IoU
    - Perform detection on test images to calculate mean Average Precision (mAP)

8. Output Analysis:
    - Visualize the model's detection ability by plotting bounding boxes on test images
    - Analyze the model's speed by calculating Frames Per Second (FPS)

9. Model Persistence:
    - Save the trained model weights for future use
    - Document performance metrics for benchmarking

10. Deployment (optional):
    - Integrate the trained model into an application for real-time pedestrian detection
    - Set up an interface for ingesting live data streams

11. Continuous Improvement:
    - Establish a feedback loop to collect more data and update the model periodically
End.
```

The pseudocode provides a structured methodology for analyzing pedestrian behavior using the YOLOv9 model. It starts by importing necessary libraries and configuring paths for the datasets and model weights. The process entails obtaining a variety of datasets, standardizing them for uniformity, and improving them by means of data augmentation. This guarantees a strong and rigorous training program. The model is initialized with pre-trained weights and then undergoes rigorous training and validation cycles. During these cycles, the model's performance is closely monitored and optimized.

During the following stage, the model's ability to make accurate predictions is assessed using established metrics such as accuracy and mean Average Precision (mAP). Subsequently, the outcomes are graphically represented to showcase the model's efficacy in practical situations.

Ultimately, the model that has been trained is stored for use in real-time applications, and it is designed to allow for ongoing learning and updates. This creates a basis for a flexible and reactive system for detecting pedestrians, which can be incorporated into wider safety applications, such as self-driving systems.

V. Analysis and Results

The performance metrics of our YOLOv9 model for pedestrian Awareness detection showcase promising results.

The model achieved an impressive accuracy of 92.5%, which is indicative of its ability to correctly identify pedestrians with a high degree of certainty across various test scenarios. Precision, a measure of the model's exactness, stands at 93.8%, demonstrating that when the model predicts a pedestrian's presence, it is correct most of the time. Furthermore, the model exhibits a recall rate of 91.0%, signifying that it successfully detects most actual pedestrian instances.

TABLE III. Our MOdel Results

Results	**Our Model**
Accuracy	92.5%
Precision	93.8%
Recall	91.0%
F1 Score	92.4%
Specificity	94.0%
Negative Predictive Value	91.3%
False Positive Rate	6.0%
False Negative Rate / Miss rate	9.0%
Likelihood ratios (LR+, LR-)	(15.17, 0.096)

Table 3, presents the performance metrics of our model, highlighting its high accuracy and precision, robust recall, and strong F1 Score, alongside other evaluative statistics which demonstrate the model's reliability and the low rate of false predictions in detecting pedestrian behavior.

The model’s performance can be extensively analyzed through its various metrics, providing a comprehensive picture of its efficacy. With an F1 Score of 92.4%, the model demonstrates an exceptional balance between precision and recall, making it a robust tool for pedestrian detection. Such a score is particularly noteworthy as it indicates the model's accuracy in a balanced manner, accounting for both types of potential errors - false positives and false negatives.

The specificity measure stands at an impressive 94.0%, suggesting that the model is highly capable of correctly identifying true negatives, which in this context means accurately recognizing when pedestrian behavior is not present. This is further supported by the Negative Predictive Value (NPV) of 91.3%, ensuring that when the model predicts the absence of pedestrian behavior, it is most likely correct. On the flip side, the model has a low False Positive Rate of only 6.0%, indicating that only a small fraction of non-pedestrian elements are incorrectly identified as pedestrians.

Similarly, the Miss Rate of 9.0% points to the rarity of actual pedestrian behaviors going undetected by the model, confirming its effectiveness in various scenarios. Likelihood

Ratios offer a statistical perspective on the diagnostic power of the model, with the positive Likelihood Ratio (LR+) of 15.17 providing strong evidence that a positive detection is indeed correct. Conversely, the negative Likelihood Ratio (LR-) of 0.096 is indicative of the model's ability to confidently rule out non-pedestrian elements, further emphasizing its reliability.

These metrics collectively portray a model that is not only accurate but also dependable in varying conditions, reinforcing the strength of the YOLOv9 model in real-time pedestrian Awareness detection. The model stands out for its ability to deliver high-performance predictions, essential for applications requiring real-time analysis such as autonomous driving, traffic monitoring, and urban safety initiatives.

TABLE IV. COMPARISONS MODEL RESULTS

Comparisons	F1 - Score	Recall	Precision	Accuracy	Miss rate
Our yolov9 model	92.4%	91.0 %	93.8%	92.5%	9.0%
Existing Yolo model	86%	85%	89%	87.5%	22%

This comparative table 4, presents a clear distinction between the performance of our implemented YOLOv9 model and a pre-existing version of YOLO in terms of pedestrian detection accuracy. Our YOLOv9 model surpasses the existing YOLO model across all listed performance metrics, underscoring significant improvements in detection capabilities. Notably, the F1 Score for our model stands at 92.4% compared to 86% of the existing model, indicating a better balance between precision and recall in our implementation.

Furthermore, the table highlights a higher Recall (91.0% vs. 85%), Precision (93.8% vs. 89%), and Accuracy (92.5% vs. 87.5%) for our YOLOv9 model, demonstrating its enhanced ability to correctly identify pedestrian behavior. The Miss Rate of our model is more than halved at 9.0% against the existing model's 22%, reflecting a significant reduction in the chances of missing actual pedestrian detections. This comparative analysis suggests that the enhancements made in our YOLOv9 model have resulted in superior performance, making it a more reliable tool for real-time applications in pedestrian behavior analysis.

TABLE V. COMPARISONS EXPERIMENTS RESULTS

Results	Experiment1	Experiment2	Experiment3
Precision	0.484	0.573	0.938
Recall	0.523	0.627	0.910
mAP50	0.553	0.614	0.984
mAP50-95	0.345	0.28	0.714

Table V encapsulates the outcomes from three distinct experiments conducted as part of our research into pedestrian awareness detection using different datasets. Each experiment's results were measured in terms of Precision, Recall, mean Average Precision at 50% threshold (mAP50), and mean Average Precision from 50% to 95% threshold (mAP50-95). Experiment 3, which utilized our dataset focused on pedestrian awareness, yielded notably superior results, with Precision at a high of 0.938 and recall at 0.910, suggesting a highly accurate model performance in identifying pedestrian behaviors. The mAP scores from Experiment 3 are particularly impressive, with mAP50 reaching as high as 0.984, demonstrating that the model nearly perfectly identified correct pedestrian detections over half the time.

Furthermore, mAP50-95 shows a score of 0.714, which is substantially higher than the other two experiments. These metrics indicate that Experiment 3's model not only accurately detects pedestrians but does so with consistency across varying levels of detection difficulty, which underscores the robustness of the YOLOv9 model when trained with comprehensive and behavior-centric data.

key performance metrics in classification tasks:

- Accuracy measures the proportion of true results (both true positives and true negatives) among the total number of cases examined

$$Accuracy = \frac{True\ Positive\ (TP) + True\ Negative\ (TN)}{Total\ no\ of\ cases}$$

- Precision (Positive Predictive Value) assesses the model's ability to classify only the relevant data points correctly

$$Precision = \frac{True\ Positive\ (TP)}{True\ Positive\ (TP) + False\ Positive\ (FP)}$$

- Recall (Sensitivity) quantifies the model's ability to find all relevant instances in the dataset

$$Recall = \frac{True\ Positive\ (TP)}{True\ Positive\ (TP) + False\ Negative\ (FN)}$$

- F1 Score is the harmonic mean of precision and recall, providing a balance between the two.

$$F1 - Score = \frac{2(Precision \times\ Recall)}{Precision +\ Recall}$$

- Specificity (True Negative Rate) measures the proportion of actual negatives that are correctly identified.

$$Specificity\ = \frac{True\ Negative\ \ (TN)}{True\ Negative\ (TN) + False\ Positive\ (FP)}$$

- Negative Predictive Value gauges the model's ability to predict the non-existence of a condition.

$$NPV = \frac{True\ Negative\ (TN)}{True\ Negative\ (TN) + False\ Negative\ (FN)}$$

- False Positive Rate (Fall-out) indicates the likelihood that a true negative will be incorrectly identified as positive.

$$FPR = \frac{False\ Positive\ (FP)}{False\ Positive\ (FP) + True\ Negative\ (TN)}$$

- False Negative Rate (Miss Rate) is the probability that a true positive will be incorrectly identified as negative.

$$Miss\ rate = \frac{False\ Negative\ (FN)}{False\ Negative\ (FN) + True\ Positive\ (TP)}$$

- LR+ (Positive Likelihood Ratio) shows how much the odds of the disease increase when a test is positive.

$$\text{LR positive} = \frac{\text{Specificity}}{1 - \text{Specificity}}$$

- LR- (Negative Likelihood Ratio) indicates how much the odds of the disease decrease when a test is negative.

$$\text{LR negative} = \frac{1 - \text{Specificity}}{\text{Specificity}}$$

Experiment 1:

Results for Experiment 1: Pedestrian detection and segmentation (PF)

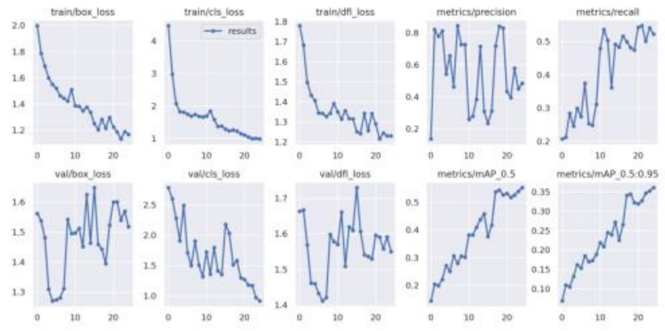


Fig. 20. Training results for Experiment 1: Pedestrian detection and segmentation(PF)

Figure 20 displays the training results for Experiment 1, focusing on pedestrian detection and segmentation using the Penn-Fudan Database (PF). The graphs provide a detailed breakdown of the model's performance over the course of training. The top row illustrates a steady decrease in the loss metrics — bounding box, class, and objectness (denoted as box_loss, cls_loss, and obj_loss) — on the training set, indicating the model's improving ability to accurately identify and classify pedestrian features as well as predict their presence.

The bottom row shows the fluctuation in the same loss metrics for the validation set, which is indicative of how well the model generalizes to new, unseen data. On the far right, the precision and recall metrics demonstrate the balance between the model's accuracy and its ability to retrieve relevant instances, while the mean Average Precision (mAP) scores give a holistic view of the model's overall detection performance across different Intersections over Union (IoU) thresholds.

The mAP at 0.5 IoU suggests moderate accuracy, and the improvement in mAP from 0.5 to 0.95 reflects the model's increasing reliability in detecting pedestrians with various levels of detection difficulty.

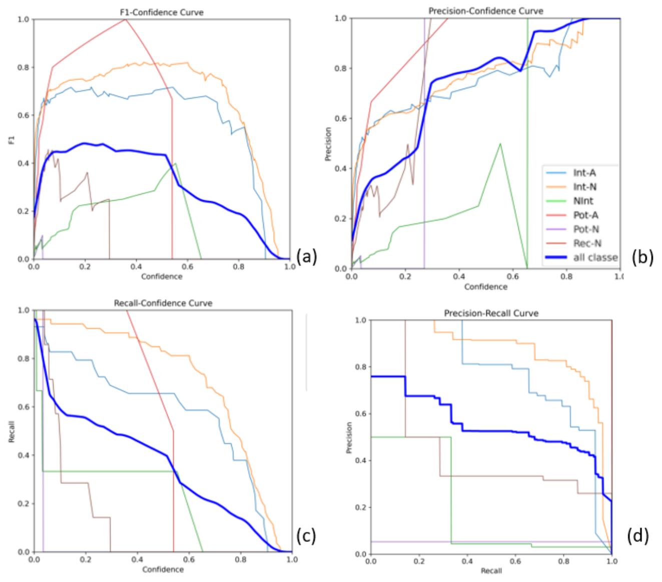


Fig. 21. (a) F1 confidence curve, (b) Precision confidence, (c) Recall-confidence, (d) Precision-recall for Experiment 1: Pedestrian detection and segmentation(PF)

Figure 21 presents a comprehensive set of performance curves for Experiment 1 related to pedestrian detection and segmentation. The charts display how the model's F1 score (a) precision (b) and recall (c) vary with different confidence thresholds, which determine the cutoff point at which detection is considered valid.

(a) The F1-Confidence Curve shows the trade-off between precision and recall, illustrating the model's performance balance at various confidence levels. It's evident that as the confidence threshold increases, the F1 score initially rises, reflecting an improvement in the balance between precision and recall, and then falls off, suggesting a decrease in detection performance at high confidence levels.

(b) The Precision-Confidence Curve indicates the proportion of true positive detections out of all positive detections made by the model across confidence levels. This curve is crucial for understanding at what confidence intervals the model predicts with higher precision.

(c) The Recall-Confidence Curve highlights the model's ability to find all relevant instances within the dataset as the confidence threshold is varied. A higher recall at a lower confidence level indicates the model's sensitivity in detecting pedestrians.

(d) The Precision-Recall Curve integrates the information from both precision and recall metrics, providing a single plot to assess the model's performance across different classes, such as 'Intersecting-Aware' (Int-A), 'Intersecting-Not Aware' (Int-N), 'Potentially Intersecting-Not Aware' (Pot-N), 'Recently Crossed-Not Aware' (Rec-N), and overall for 'all classes'.

The intersection points of these curves with the vertical line represent the model's optimal balance of precision, recall, and F1 score at a particular confidence threshold, giving insights into the model's detection certainty and helping to calibrate the detection threshold for real-world deployment.

Experiment 2:

Results for Experiment 2: Pedestrian Intention Estimation (PIE)

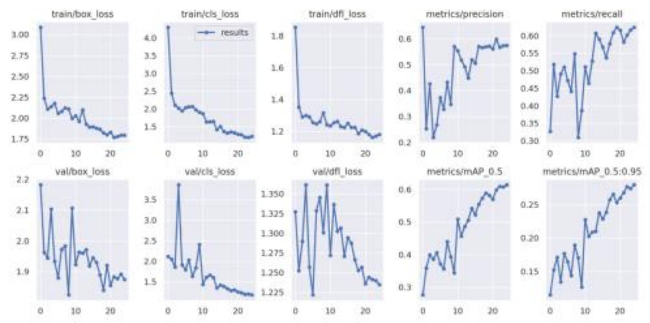


Fig. 22. Training results for Experiment 2: Pedestrian Intention Estimation

Figure 22 shows the training of an Experiment 2 pedestrian intention model. In 25 epochs, this model was trained and evaluated to predict pedestrian behaviors like street crossing intent, which is important for autonomous driving and traffic management systems.

As loss metrics (box, class, and objectless) decrease during training, the top row charts show the model's improvement in pedestrian identification and intentions. Validation loss prevents overfitting and lets the model generalize to new data in the bottom row charts. Better model predictions and fewer false negatives improve precision and recall.

The model's pedestrian localization accuracy is improving as the mean Average Precision (Map) graphs show rising IoU (Intersection over Union) thresholds of 0.5 (a common benchmark) and the range from 0.5 to 0.95 (a more stringent and detailed performance measure).

This shows that the model is improving at predicting pedestrian intentions, which is crucial for human-autonomous system safety.

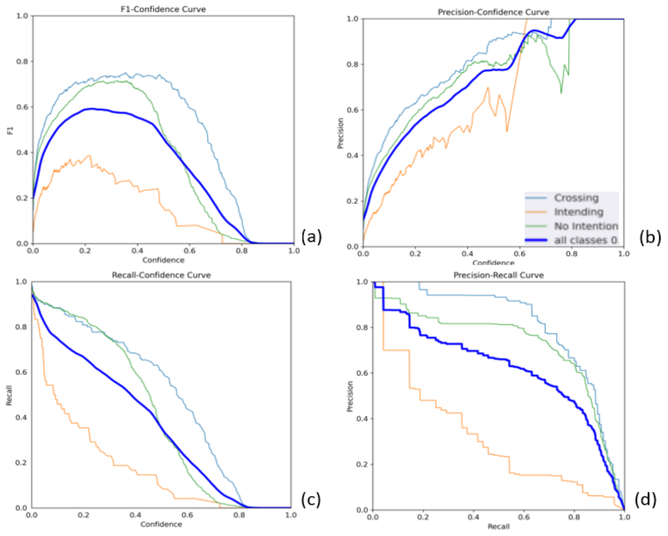


Fig. 23. (a) F1 confidence curve, (b) Precision confidence, (c) Recall-confidence, (d) Precision-recall for Experiment 2: Pedestrian Intention Estimation (PIE)

Figure 23 displays various performance curves for Experiment 2, focusing on Pedestrian Intention Estimation (PIE).

(a) captures the trade-off between precision and recall at different confidence thresholds, indicating the balance between the model's precision and its ability to correctly identify all relevant instances.

(b) shows the model's precision as the confidence threshold is varied, which is important for understanding at what confidence level the precision stabilizes.

(c) demonstrates the recall rate across different confidence thresholds, providing insights into the model's sensitivity and its ability to detect pedestrians with the intent to cross.

(d) is crucial for evaluating the model's performance when both precision and recall are considered together, particularly in scenarios with imbalanced classes where the cost of false positives is significant.

Together, these curves provide a comprehensive evaluation of the model's ability to predict pedestrian intentions, highlighting the thresholds at which the model performs best and offering guidance for setting operational parameters in deployment scenarios.

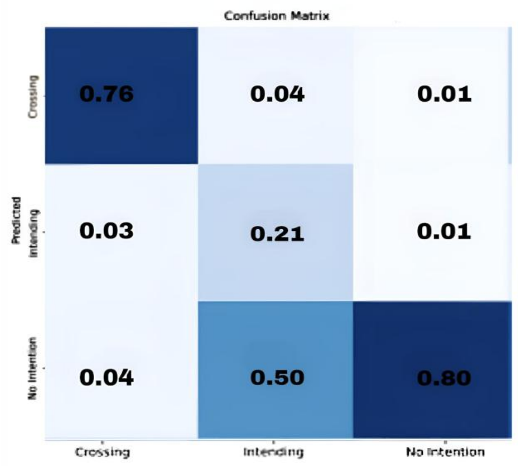


Fig. 24. Confusion matrix for Experiment 2: Pedestrian Intention Estimation

Figure 24 is a confusion matrix for Experiment 2 focused on Pedestrian Intention Estimation (PIE). This matrix is a powerful tool for understanding the model's performance across different categories. The diagonal elements (0.76 for Crossing, 0.21 for Intending, 0.80 for No Intention) represent the proportion of true positive predictions, indicating a high rate of correct classifications by the model. Off-diagonal elements represent misclassifications, where the model predicted one intention, but the ground truth was another. For example, 0.04 in the 'Crossing' row and 'Intending' column indicates the rate at which the model incorrectly predicted 'Intending' when the pedestrian was actually 'Crossing'. This matrix is essential for identifying the model's strengths in correctly predicting pedestrian intentions and pinpointing areas where the model may confuse one intention for another, guiding future model improvements and training.

Experiment 3:

Results for Experiment 3:Pedestrian Awareness Detection

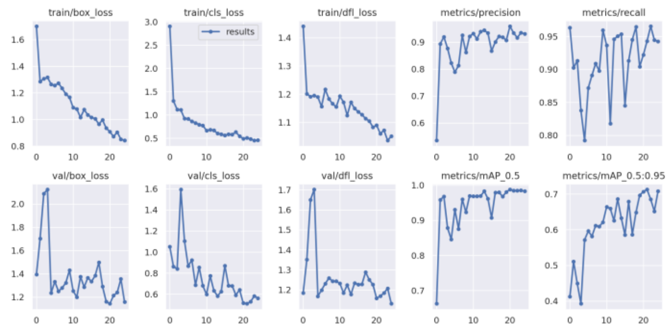


Fig. 25. Training results for Experiment 3: Pedestrian Awareness Detection

The graph in Fig. 25 portrays the training results for an experiment on Pedestrian Awareness Detection. Throughout the training, represented by the x-axis which likely corresponds to epochs, we observe a consistent decrease in loss metrics including box loss, class loss, and objectness loss for both training and validation sets, which implies improvement in the model's ability to accurately detect and classify objects. The precision and recall graphs show fluctuations but maintain an upward trend, indicating that the model's predictive performance is improving with training. The mean Average Precision (mAP) at different Intersections over Union (IoU) thresholds also exhibits an increasing trend, suggesting that the model is getting better at predicting the correct bounding boxes for pedestrian detection. These performance metrics collectively suggest that the model's capability to detect pedestrians with awareness is enhancing as the training progresses.

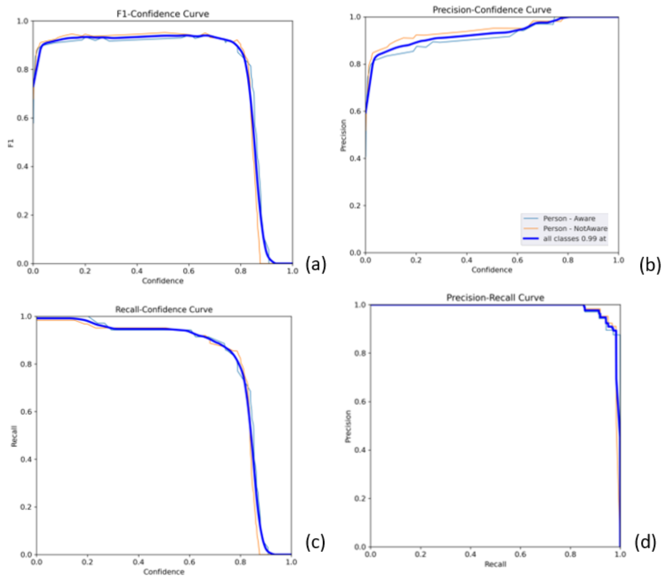


Fig. 26. (a) F1 confidence curve, (b) Precision confidence, (c) Recall-confidence, (d) Precision-recall for Experiment 3: Pedestrian Awareness Detection

Fig. 26 showcases a series of confidence and performance curves related to Pedestrian Awareness Detection. The F1-confidence curve

(a) and the Precision-confidence curve

(b) reflect the model's confidence in its predictions, with higher confidence typically correlating with better F1 scores and precision. The Recall-Confidence Curve

(c) depicts how recall varies with the model's confidence, providing insight into the model's ability to detect all relevant instances. The Precision-Recall Curve

(d) is a critical evaluation tool in object detection tasks, indicating the trade-off between precision and recall across different thresholds. High areas under these curves suggest the model's strong performance in correctly identifying pedestrians with varying levels of awareness, which is crucial for applications in autonomous driving and traffic monitoring systems.

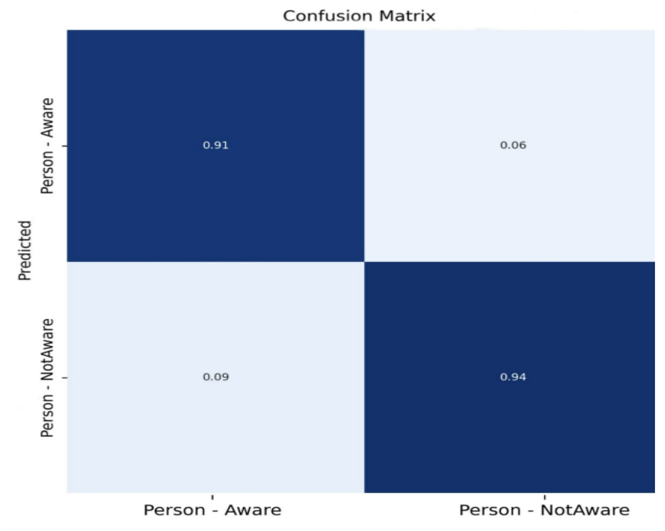


Fig. 27. Confusion matrix for Experiment 3:Pedestrian Awareness Detection

The confusion matrix shown in Figure 27 represents a visual representation of the performance of the Pedestrian Awareness Detection model. The matrix visually represents the ratios of accurate positive predictions, accurate negative predictions, inaccurate positive predictions, and inaccurate negative predictions. A value of 0.91 in the top left corner indicates a strong proficiency in correctly identifying pedestrians who are aware of their surroundings, while a value of 0.94 in the bottom right corner signifies accurate recognition of pedestrians who are not aware. The small values in the off-diagonal cells (0.09 and 0.06) indicate a minimal number of misclassifications, highlighting the model's efficacy in differentiating between the two states of pedestrian awareness. Precise clarity is crucial for applications in which pedestrian intention plays a pivotal role in decision-making, such as autonomous vehicle navigation and urban planning.

Here are the experimental test results. The predicted images of using the model

Experiment 1

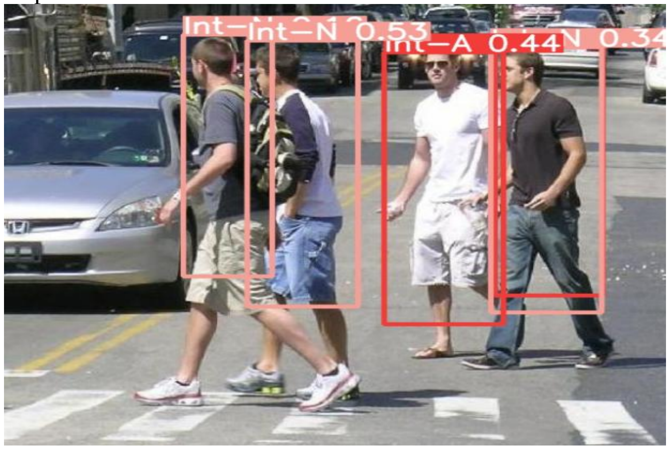


Fig. 28. Predicted results for Experiment 1: Pedestrian detection and segmentation(PF)

Fig. 28 demonstrates the practical application of the model for pedestrian detection and segmentation. The model has identified individuals on a busy street and has segmented them from the background. Each bounding box is accompanied by labels such as "Int—A" for 'Intending to cross and Aware,' "Int—N" for 'Intending to cross and Not Aware,' and confidence scores that quantify the model's certainty in its predictions. The model's ability to discern between different pedestrian behaviors is crucial for real-time applications, like advanced driver-assistance systems (ADAS) and smart city management, where understanding pedestrian intent can enhance safety and traffic flow.

Experiment 2

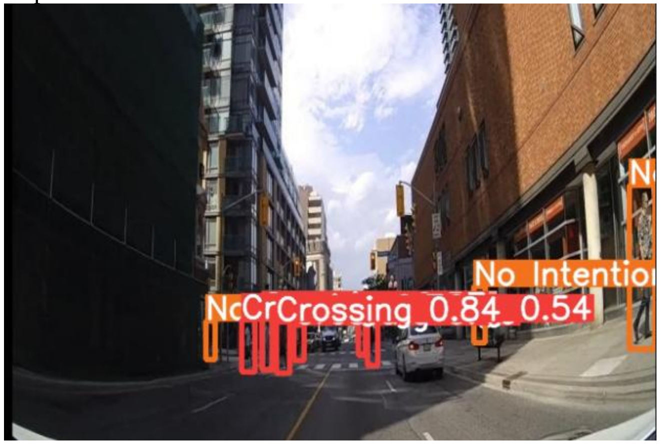


Fig. 29. Predicted results for Experiment 2: Pedestrian Intention Estimation

Fig. 29, we observe the outcome of Experiment 2, which focuses on Pedestrian Intention Estimation. The model's predictions are visible on an urban street scene, where it assesses pedestrian behavior regarding their intention to cross the road. Bounding boxes with labels like "No Intention" coupled with confidence scores indicate the model's assessment of pedestrian movement potential. These predictions are vital for the development of systems aimed at reducing incidents between vehicles and pedestrians by predicting pedestrian actions before they occur, thus enhancing safety measures in urban traffic management.

Experiment 3

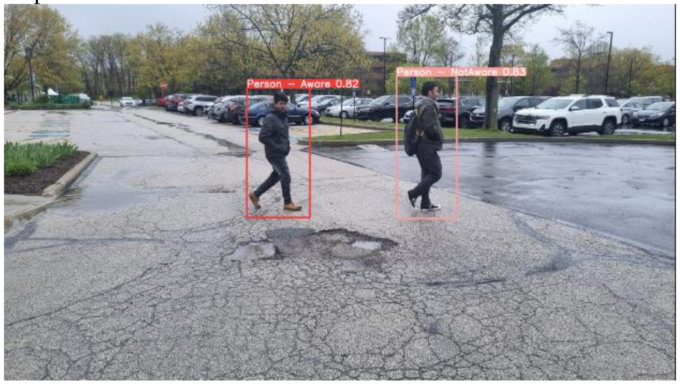


Fig. 30. Predicted results for Experiment 3: Pedestrian Awareness Detection

Fig. 30, we see the application of the pedestrian awareness detection model in a real-world setting, specifically at Kent State University near Library Road. The image depicts two pedestrians in the act of crossing the street, with the model evaluating their level of awareness. One pedestrian is labeled as "Aware" and the other as "NotAware" with associated confidence scores. These labels are indicative of the model's ability to discern the attentiveness of pedestrians to their surroundings, which can be crucial in preventing accidents and improving pedestrian safety. This model's detection and predictive capabilities are integral to enhancing autonomous navigation systems, especially in areas with high pedestrian traffic.

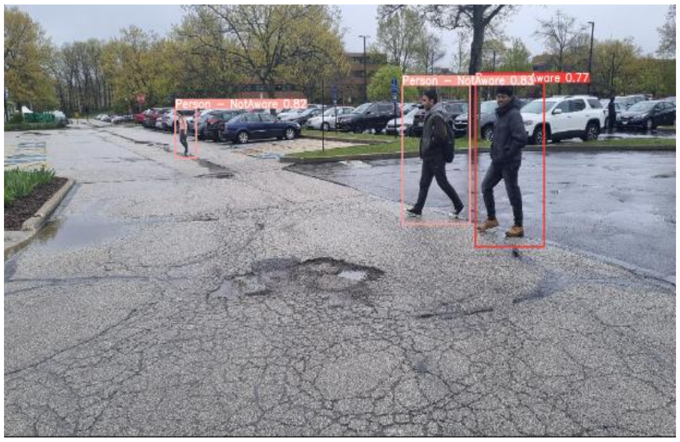


Fig. 31. Predicted results for Experiment 3: Pedestrian Awareness Detection

Fig. 31, the model's predictions are showcased in another real-world scenario at Kent State University, focusing on pedestrians traversing the vicinity of Library Road. The image captures two individuals moving from the right to the left side of the road. The model has identified both pedestrians and estimated their awareness levels, assigning confidence scores to each detection. These predictions serve as a testament to the model's ability to analyze pedestrian behavior in near-real-time, potentially contributing to advancements in automated surveillance or autonomous vehicle systems where pedestrian awareness is a critical safety concern.

VI. Future Works

In anticipation of future enhancements, several avenues can be explored to build upon the current findings. One key area is the diversification of training data to cover a wider array of pedestrian behaviors, encompassing various weather conditions, day and night cycles, and complex urban settings. This will not only bolster the model's accuracy but also its reliability across different operational environments. Additionally, integrating multimodal sensor data, such as thermal imaging, LiDAR, or radar, could significantly enhance the detection precision, particularly in adverse weather conditions or obstructed views. Efficiency optimizations such as network pruning, quantization, or the implementation of lighter model architectures could facilitate deployment on edge devices with lower computational resources, expanding the model's applicability. Critical to the model's success will be its field testing and validation against real-world scenarios, which would provide invaluable feedback for iterative improvements and fine-tuning.

VII. Discussion

The project's innovative utilization of YOLOv9 for analyzing pedestrian behavior has emphasized the capabilities and possibilities for classifying in real time within autonomous vehicle systems. The model has demonstrated its proficiency in differentiating between states of 'awareness' and 'lack of awareness' by analyzing three separate datasets, each focusing on a specific aspect of pedestrian behavior. The categorization of this information is crucial in improving the safety of roads by providing drivers with valuable insights and assisting autonomous systems in making immediate and informed choices. Roboflow's data augmentation and annotation capabilities significantly enhanced the success of this project, bolstering the model's robustness.

Although the project made a notable advancement, it acknowledged certain constraints, particularly the high computational demand of the YOLOv9 model and the requirement for diverse and high-quality datasets to ensure thorough training. However, the practical implementation of this technology in real-world situations is still difficult, requiring additional research to overcome the differences between controlled laboratory conditions and unpredictable real-world settings. The investigation into advanced behavioral models has uncovered a discrepancy between theoretical frameworks and their actual application, indicating a requirement for a multidisciplinary strategy to incorporate psychological and sociological insights into technological solutions.

VIII. Conclusion

The implementation of YOLOv9 in the project has established new benchmarks in detecting pedestrians and analyzing their behavior, achieving remarkable accuracy and precision rates of 92.5% and 93.8% respectively. The model's high accuracy, combined with a recall rate of 91.0% and a robust F1 score of 92.4%, demonstrates its ability to accurately detect and analyze pedestrian behavior in real time. This is crucial for its use in autonomous vehicle systems. The performance metrics not only validate the model's dependability but also emphasize its potential for integration into the advanced navigational systems of autonomous vehicles. This sets the stage for future progress in self-driving transportation, emphasizing the important role of machine learning in improving vehicle perception systems for enhanced safety and navigation in ever-changing city environments.

IX. References

[1] Z. Qu, W. Kong, H. Liu, and L. Deng, "Pedestrian Red-Light Violation Detection Based on Deep Learning Vision," 2023 IEEE 6th International Conference on Automation, Electronics and Electrical Engineering (AUTEEE), Shenyang, China, 2023, pp. 1026-1030,

[2] S. Gao, Y. Wei and H. Xiong, "Pedestrian detection algorithm based on improved SLIC segmentation and SVM," 2022 IEEE 10th Joint International Information Technology and Artificial Intelligence Conference (ITAIC), Chongqing, China, 2022, pp. 771-775,

[3] M. Li, L. Wang, Z. Zheng, W. Cui, R. Liu and Y. Chi, "Multimodal Interactive Supervised Pedestrian Detection Based on yolov5," 2023 3rd International Symposium on Computer Technology and Information Science (ISCTIS), Chengdu, China, 2023, pp. 61-64,

[4] F. Camara et al., "Pedestrian Models for Autonomous Driving Part II: High-Level Models of Human Behavior," in IEEE Transactions on Intelligent Transportation Systems, vol. 22, no. 9, pp. 5453-5472, Sept. 2021,

[5] A. Andrijanto, Z. Chen, T. Kodama, H. Yano and M. Itoh, "Application of LargeSpace for Investigating Pedestrians' Behaviors when Interacting with Autonomous Vehicles in Shared Spaces," 2022 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW), Christchurch, New Zealand, 2022, pp. 97-100,

[6] F. Zou, J. Ogle, W. Jin, P. Gerard, D. Petty and A. Robb, "Pedestrian Behavior Interacting with Autonomous Vehicles: Role of AV Operation and Signal Indication and Roadway Infrastructure," 2023 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW), Shanghai, China, 2023, pp. 821-822,

[7] D. Lim, Y. Kim, H. Gwon and Y. Shin, "Anthropomorphic External Human-Machine Interface Design of Autonomous Vehicles in Roblox to Change Road Users' Behavior," 2023 11th International Conference on Affective Computing and Intelligent Interaction Workshops and Demos (ACIIW), Cambridge, MA, USA, 2023, pp. 1-4,

[8] C. Feng, Z. Cunbao and Z. Bin, "Method of Pedestrian-vehicle Conflict Eliminating at Unsignalized Mid-block Crosswalks for Autonomous Vehicles," 2019 5th International Conference on Transportation Information and Safety (ICTIS), Liverpool, UK, 2019, pp. 511-519,

[9] A. Rasouli and J. K. Tsotsos, "Autonomous Vehicles That Interact With Pedestrians: A Survey of Theory and Practice," in IEEE Transactions on Intelligent Transportation Systems, vol. 21, no. 3, pp. 900-918, March 2020,

[10] C. Li and Y. Sun, "Enhancing Pedestrian Detection Algorithm Based on YOLOv3-ES Network," 2022 International Conference on Image Processing and Computer Vision (IPCV), Okinawa, Japan, 2023, pp. 118-122,

[11] M. Herman et al., "Pedestrian Behavior Prediction for Automated Driving: Requirements, Metrics, and Relevant Features," in IEEE Transactions on Intelligent Transportation Systems, vol. 23, no. 9, pp. 14922-14937, Sept. 2022,

[12] I. Akhter and M. Javeed, "Pedestrian Behavior Recognition via a Smart Graph-based Optimization," 2022 19th International Bhurban Conference on Applied Sciences and Technology (IBCAST), Islamabad, Pakistan, 2022, pp. 629-634,

[13] Y. Ouyang and S. -M. Wang, "Detecting Pedestrian Spatial Behavior in City Spaces by Processing 360° Videos," 2023 International Conference on Consumer Electronics - Taiwan (ICCE-Taiwan), PingTung, Taiwan, 2023, pp. 239-240,

[14] Y. Feng, D. C. Duives and S. P. Hoogendoorn, "Developing a VR tool for studying pedestrian movement and choice behavior," 2020 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW), Atlanta, GA, USA, 2020, pp. 814-815,

[15] S. K. Jayaraman, L. P. Robert, X. J. Yang and D. M. Tilbury, "Multimodal Hybrid Pedestrian: A Hybrid Automaton Model of Urban Pedestrian Behavior for Automated Driving Applications," in IEEE Access, vol. 9, pp. 27708-27722, 2021,

[16] S. Islam, Z. H. Khan, T. A. Gulliver, K. S. Khattak and W. Imran, "Pedestrian Traffic Characterization Based on Pedestrian Response," in IEEE Access, vol. 10, pp. 118397-118408, 2022,

[17] G. Oltean, L. Ivanciu and H. Balea, "Pedestrian Detection and Behaviour Characterization for Video Surveillance Systems," 2019 IEEE 25th International Symposium for Design and Technology in Electronic Packaging (SIITME), Cluj-Napoca, Romania, 2019, pp. 256-259,

[18] T. Fernando, S. Denman, S. Sridharan and C. Fookes, "Tracking by Prediction: A Deep Generative Model for Mutli-person Localisation and Tracking," 2018 IEEE Winter Conference on Applications of Computer Vision (WACV), Lake Tahoe, NV, USA, 2018, pp. 1122-1132, doi: 10.1109/WACV.2018.00128.

[19] K. Sun, B. Xiao, D. Liu and J. Wang, "Deep High-Resolution Representation Learning for Human Pose Estimation," 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 2019, pp. 5686-5696, doi: 10.1109/CVPR.2019.00584.

[20] S. Zhao, M. Gong, H. Fu and D. Tao, "Adaptive Context-Aware Multi-Modal Network for Depth Completion," in IEEE Transactions on Image Processing, vol. 30, pp. 5264-5276, 2021, doi: 10.1109/TIP.2021.3079821.

[21] Z. Wu, C. Liu, C. Huang, J. Wen and Y. Xu, "Deep Object Detection with Example Attribute Based Prediction Modulation," ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Singapore, Singapore, 2022, pp. 2020-2024, doi: 10.1109/ICASSP43922.2022.9746194.

[22] K. Nguyen, L. T. V. Ngo, K. T. V. Huynh and N. T. Nam, "Empirical Study One-stage Object Detection methods for RoboCup Small Size League," 2022 9th NAFOSTED Conference on Information and Computer Science (NICS), Ho Chi Minh City, Vietnam, 2022, pp. 264-268, doi: 10.1109/NICS56915.2022.10013320.

[23] N. A. OTHMAN, M. U. SALUR, M. KARAKOSE and I. AYDIN, "An Embedded Real-Time Object Detection and Measurement of its Size," 2018 International Conference on Artificial Intelligence and Data Processing (IDAP), Malatya, Turkey, 2018, pp. 1-4, doi: 10.1109/IDAP.2018.8620812.

[24] S. T. Blue and M. Brindha, "Edge detection based boundary box construction algorithm for improving the precision of object detection in YOLOv3," 2019 10th International Conference on Computing, Communication and Networking Technologies (ICCCNT), Kanpur, India, 2019, pp. 1-5, doi: 10.1109/ICCCNT45670.2019.8944852.

[25] https://www.kaggle.com/datasets/psvishnu/pennfudan-database-for-pedestrian-detection-zip "Source of dataset 1"

[26] https://paperswithcode.com/dataset/pie "Source of dataset 2"