DEEP LEARNING ALGORITHMS FOR COMPLEX TRAFFIC BEHAVIOUR RECOGNITION IN MONO-CAMERA INTELLIGENT VEHICLES

A Thesis Submitted for the Degree of Doctor of Philosophy

By

LUIZ GOLDMAN GALVAO

Department of Electronic and Electrical Engineering, Brunel University London

Date
2025

I dedicate this work to my beloved family—my wife, Damaris Thompson Dias Galvao, and my children, Davi Thomson Galvao and Amy Thompson Galvao—who have always been my highest priority.

Acknowledgements

I am profoundly grateful to everyone who has supported me throughout the course of this PhD journey. It has been a long and challenging road, and the encouragement, guidance, and help I received have been essential in carrying me through.

First and foremost, I want to thank God for opening the door to this opportunity and for surrounding me with so many wonderful people who have supported and inspired me along the way.

I extend my deepest appreciation to my supervisor, Dr. Md Nazmul Huda, whose invaluable guidance, management, encouragement, and patience were instrumental to the completion of this thesis. His expertise and insight have shaped my research and enriched my academic growth.

I am also deeply grateful to my supervisory team members and review panel members, Prof. Maysam Abbod, Prof. Gareth Taylor, Prof. Wamadeva Balachandran, Dr. Nikolaos Boulgouris, Dr. Ruiheng Wu, and Prof. Hongying Meng, for their insightful feedback. My heartfelt thanks go as well to my colleagues in the IT team, particularly Nalin Soni, for his countless assistance with IT support, which was crucial for my research.

I gratefully acknowledge the financial support provided by UK Research and Innovation and the EPSRC DTP Scholarship, without which this research would not have been possible.

To my family—especially my wife and children—thank you for your unwavering belief in me. Your love, patience, and encouragement have been my greatest source of strength. I am also deeply thankful to my brothers in Christ for their constant support and prayers, which sustained me through the toughest moments.

Finally, I would like to thank everyone who has been part of this journey, in one way or another. Your support has been truly invaluable.

Abstract

Intelligent vehicles have the potential to revolutionise transport by enhancing safety, reducing congestion, and improving efficiency. A critical component of IVs is perception—the ability to accurately interpret the environment for informed driving decisions. While multi-sensor systems are widely used, camera-only solutions offer a cost-effective alternative but face challenges in recognising the intentions of diverse traffic agents in complex urban environments. This thesis addresses limitations in current mono-camera approaches, which often overlook explicit behavioural cues and lack integrated pipelines for discrete intention behaviour recognition.

A literature review identified key gaps, including the absence of methods for predicting discrete intention behaviours across various traffic agents, insufficient public datasets capturing complex urban scenes for intention prediction, and underutilisation of explicit cues such as vehicle light signals and object orientation. Additionally, existing studies rarely consider integrated pipelines that combine detection, tracking, and behaviour recognition while accounting for error propagation across stages.

To address these gaps, a monocular traffic hazard dataset was developed, capturing diverse traffic agents and explicit behavioural cues relevant for hazard recognition. Deep learning models, including Vision Transformers and Convolutional Networks, were designed to leverage these features, demonstrating improved accuracy in recognising complex traffic behaviours from single images. Experiments exploring different input features, observation horizons, and class granularities revealed that combining explicit and implicit cues enhances recognition performance.

A complete hazard recognition pipeline was implemented, integrating detection, tracking, and behaviour recognition to assess system-level performance. Results highlighted the challenges of error propagation across modules while demonstrating the feasibility of end-to-end monocular pipelines for complex traffic behaviour recognition.

The key contributions of this thesis include the development of a targeted monocular dataset for behaviour recognition, the creation of models utilising underexplored visual cues, and the integration of these models into a unified hazard recognition pipeline for camera-only IV systems. This work demonstrates the potential of monocular approaches for traffic behaviour recognition and provides a foundation for cost-effective, scalable intelligent vehicle solutions. Future research should focus on expanding dataset diversity, improving model robustness, and incorporating multi-agent interactions to enhance real-world applicability.

Contents

List of Tables

List of Figures

Nomenclature

$A3D$ Anan Accident Detection

AAD Anticipating Accidents In Dashcam

$ADAS$ Advanced Driver Assistance Systems

ADE Average Displacement Error

$AI-TP$ Attention-Based Interaction-Aware Trajectory Prediction

$ANDE$ Average Non-Linear Displacement Error

ANN Artificial Neural Network

$ARIMA$ Auto-Regressive Integrated Moving Average

AUC Area Under The Curve

AV Autonomous Vehicle

BEV Bird's Eye View

BN Bayesian Network

CDD Car Crash Dataset

CDD Charge-Coupled Device

$CMOS$ Complementary Metal-Oxide Semiconductors

CNN Convolutional Neural Networks

$ConvGRU$ Convolutional GRU

CTA Causality In Traffic Accident

$CVAE$ Conditional Variational Auto-Encoder

$DARPA$ Defence Advanced Research Projects Agency

$DESIRE$ Deep Stochastic Inverse Optimal Control Rnn Encode-Decoder

$DOTA$ Detection Of Traffic Anomaly

EV Ego Vehicle

FAD Final Average Displacement

FCN Fully Connect Networks

FDE Average Final Displacement Error

FPS Frames Per Second

GAN Generative Adversarial Network

GAT Graph Attention Network

$GCNN$ Graph Convolutional Neural Networks

$GNSS$ Global Navigation Satellite Systems

$GRIP$ Graph-Based Interaction-Aware Trajectory Prediction

GRU Gated Recurrent Unit

$GTAV$ Grand Theft Auto V

HDD Honda Research Institute Driving Dataset

$HEAT$ Heterogeneous Edge-Enhanced Graph Attention Network

$HTMI$ Heterogenous Interaction Model

IEB Iterative Elastic Bins

IMU Inertial Measurement Unit

IOC Inverse Optimal Control

IV Intelligent Vehicle

LC Lane Change

LCE Lane Change Event

$LIDAR$ Light Detection And Ranging

LK Lane Keeping

LLC Left Lane Change

$LSTM$ Long-Short Term Memory

$LSTM - RNN$ Long Short Term Memory Recurrent Neural Networks

MAD Mean Average Displacement

MAE Mean Absolute Error

$MATF$ Multi-Agent Tensor Fusion

ML Machine Learning

MLP Multi-Layer Perceptron

$NIDB$ Near-Miss Incident Database

$NIDB$ Near-Miss Incident Dataset

OGM Occupancy Grid Map

OHT Observation Horizon Time

OOI Object Of Interest

PAF Part-Association-Fields

PF Particle Filter

PHT Prediction Horizon Time

$PHTR$ Prediction Horizon Time Ratio

PID Proportional-Integral-Derivative

PIF Part-Intensity-Fields

$RADAR$ Radio Detection And Ranging

RLC Right Lane Change

$RMSE$ Root Mean Square Error

RNN Recurrent Neural Networks

$ROC - AUC$ Receiver Operating Characteristic Curve

SAE Society Of Automotive Engineers

$SLAM$ Simultaneous Localisation And Mapping

SO Surrounding Objects

$SONAR$ Sound Navigation System

$ST - GCN$ Spatio-Temporal GCN

$STAG$ Spatio Temporal Action Graph

SV Surrounding Vehicle

TIM Temporal Integration Method

TO Target Object

TSM Target Selection Method

TTE Time-To-Event

TV Target Vehicle

$V2I$ Vehicle To Infrastructure

$V2V$ Vehicle To Vehicle

$VIENA$ Virtual Environment For Action Analysis

$VIP - LSTM$ Vehicle Intention Prediction Long Short Term Memory

WHO World Health Organisation

$YOLOP$ You Only Look Once For Panoptic

Chapter 1

Introduction

The rapid advancement of Intelligent Vehicles (IV)s has the potential to revolutionise transportation by enhancing road safety, reducing congestion, and improving driving efficiency. A critical component of IV technology is **perception**—the ability to accurately interpret the surrounding environment to make informed driving decisions. While multi-sensor fusion approaches incorporating LiDAR, radar, and ultrasonic sensors are widely adopted, recent discussions in the autonomous vehicle community have explored the feasibility of camera-only systems. This thesis investigates the development of deep learning algorithms and a specialised dataset to enable mono-camera IVs to learn complex traffic behaviours, aiming to enhance perception capabilities while reducing system cost and complexity.

According to the World Health Organisation (WHO) there were 1.19 million road traffic fatalities in 2021 [1]. In the UK, 2,055 and 119,850 road casualties were reported in 2020 and 2021, respectively, with 1,676 resulting in fatalities [2, 3]. The European Commission reported over 20,000 traffic-related deaths in 2022 [4]. The UK Department for Transport and the US National Highway Traffic Administration reported that 94% of traffic collisions resulted from human error and imprudence, including factors such as driver distraction, impairment, disobedience, and fatigue [5, 6]. The American Transportation Research Institute reported that the freight sector has a cost of $74.5 billion due to congestion issues [7]. Congestion greatly impacted the UK in 2019, where drivers lost an average of 115 hours and cost the country a value of £6.9 billion [7]. Other factors influencing road traffic collisions are vehicle design, traffic conditions, geometric characteristics of the road, and weather conditions [8].

Enforced legislation, other transportation methods, road improvements, and congestion charges have been proposed to tackle these road traffic issues. Enforced legislation includes road speed limit, drinking and driving, use of motorcycle helmets, use of seat belts, and child restraints [9]. Different methods of transportation include electric scooters and bicycle sharing [10, 11, 12], and telecommuting transportation [13, 14]. Road improvements include smart highways and smart roads, [15, 16]. The congestion charge is an imposed fee for drivers to drive in areas prone to congestion, such as main cities like London and Stockholm [17, 18, 19]. The fee aims to reduce traffic congestion, improve air quality, and incentivise individuals to use another method of transportation. Although these measures are in place, the number of road users is expected to double by 2050 [20], and nearly two billion cars are projected to be on the road within the next twelve years [21]. Therefore, these

solutions may prove insufficient. Moreover, they do not directly address the primary cause of traffic accidents: *human drivers*.

IVs have the potential to solve or mitigate the issues mentioned earlier. For example, ADAS systems can take control of the vehicle in situations where humans overlook a potential traffic collision. A fully automated vehicle is expected to operate without any human intervention, eliminating the potential for human errors and imprudence [6]. IVs are also expected to communicate with their environment and other road users, enabling them to make better use of road space, reducing congestion, pollution, and journey time [5]. The authors [22] evaluated the innovative technologies integrated into vehicles to enhance driver safety and showed that automated technologies like automatic emergency braking and lane departure assistance have already demonstrated prominent benefits.

Since IVs can offer the benefits mentioned above, many automotive companies worldwide have been testing and deploying IV technologies. For instance, in the United States companies such as Tesla [23] and Waymo [24]; in China, Baidu [25] and Tencent [26]; in Germany, Mercedes-Benz [27], BMW [28] and Volkswagen [29]; in Japan, Toyota [30] and Honda [31], and in South Korea, Hyundai [32] and Kia [33].

Recent debates in the IV community have centred on whether camera-only systems can eventually replace the need for additional sensors such as radar, lidar, and ultrasonic devices [34, 35, 36, 37, 38, 39, 40, 41, 42, 43]. In a notable discussion, former Tesla AI director Andrej Karpathy outlined Tesla's rationale for eliminating these extra sensors in favour of a vision-centric approach. According to Karpathy, relying solely on cameras offers several potential advantages:

1. **Cost and Complexity Reduction:**
   Karpathy and [41] argue that additional sensors not only increase the cost but also add significant complexity to the hardware and software stacks. Extra sensors demand elaborate calibration, sensor fusion algorithms, and maintenance of multiple data pipelines. By focusing on cameras, manufacturers can streamline the design, reduce production costs, and simplify supply chain logistics—a critical consideration for mass-market vehicles [41].

2. **Software Simplification:**
   Removing sensors like radar and ultrasonics reduces the burden on software systems. The fewer the sensor modalities, the less data fusion is required, which can lead to a cleaner and potentially more robust perception pipeline. Karpathy emphasises that an effective vision system could, in theory, extract all the necessary information for safe driving, mirroring the human reliance on vision.

3. **Fleet Learning and Data Accumulation:**
   A camera-only approach can leverage a large fleet of vehicles to gather extensive visual data. Over time, this accumulated dataset can be used to continuously improve deep learning algorithms, potentially compensating for the current limitations of a single-sensor modality. Tesla's strategy relies on real-world data to enhance the accuracy and reliability of its vision-based system.

4. **Philosophical and Practical Considerations:**
   Tesla's "no part" philosophy advocates for minimising system components to

reduce potential points of failure. From this perspective, a camera-only system is not only a cost-saving measure but also aligns with a broader strategic vision of relying on deep learning to overcome the inherent limitations of sensor hardware.

Despite these arguments, the broader autonomous driving community remains divided. Critics point out that current camera-only systems face challenges in reliably handling adverse weather conditions, poor lighting, and edge cases—scenarios where redundant sensor modalities have traditionally provided valuable backup. For instance, some teams argue that technologies like lidar offer superhuman sensing capabilities that are crucial for detecting obstacles that cameras might miss. Moreover, while vision systems show promise, many researchers remain uncertain about the timeline for achieving the required level of safety and reliability solely through camera inputs.

In conclusion, while current implementations of camera-only systems have not yet reached the robustness provided by multi-sensor fusion, ongoing advances in deep learning and large-scale data collection may eventually bridge this gap. Karpathy's perspective highlights the potential benefits of reducing sensor complexity and cost, suggesting that with continued improvement in vision algorithms and fleet learning, cameras alone might suffice for future IV perception. This approach represents a promising avenue for research, although its ultimate viability will depend on overcoming the current limitations in challenging driving scenarios.

1.1 Challenges of IVs

The development and deployment of IVs represent a significant transformation in transportation, offering the potential for enhanced safety, efficiency, and convenience. However, despite notable technological progress, IVs still face substantial barriers to widespread adoption. These challenges span several domains, including regulatory compliance, safety standards, cybersecurity, and the ability to operate in complex, unpredictable environments. Additional concerns—such as the need for high-definition mapping (HDM), the modelling of rare and extreme scenarios, and broader social responsibility—further complicate the integration of IVs into real-world traffic systems. This section outlines the primary obstacles hindering large-scale deployment and identifies key areas for future research and innovation.

Legislation and Regulation

The deployment of autonomous vehicles presents substantial legislative and regulatory challenges. As noted by [44] key issues include enforcement, compliance with national and international policies, adherence to safety and environmental standards, and meeting technical conformity requirements.

Safety Standards

The deployment of IVs has far-reaching implications across multiple domains, including public safety, human behaviour, urban planning, economic systems, and public health [45]. Regulatory frameworks are essential in managing these impacts;

however, existing legislation is often insufficient to support the widespread adoption of IVs. As [46] notes, there is a pressing need for harmonised social frameworks and standardised regulations to enable large-scale deployment. Manufacturers must also navigate a complex landscape of considerations, including human rights, data privacy, fairness, transparency, ethical standards, cultural and legal norms, and potential public resistance—such as opposition to banning non-autonomous vehicles or concerns over the right to drive.

The growing integration of AI-driven features in IVs further complicates compliance with safety standards. According to [47] AI algorithms introduce challenges related to performance consistency, predictability, and reliability, particularly in unforeseen scenarios. This underscores the need for a robust safety architecture, including fail-soft mechanisms for Automotive Safety Integrity Level (ASIL) D, AI system integration, and machine learning-based safety protocols [48].

Modelling Extreme and Rare Scenarios

One of the primary challenges in evaluating the safety of IVs lies not in the total distance driven without incidents, but in the context and conditions under which those miles are accumulated. Metrics such as "miles per shutdown"—which track system failures—offer more meaningful insights into safety performance [46].

Despite accumulating thousands of miles without disengagements, IVs may still lack the robustness required to handle rare and safety-critical scenarios [49]. Most existing datasets are derived from naturalistic driving environments, which contain very few such critical events, limiting the ability of models to learn from them. Generating realistic safety-critical scenarios introduces several challenges, including fidelity (balancing realism and difficulty), efficiency, diversity, transferability (adapting scenarios across environments), and controllability (targeting specific situations rather than relying on randomness) [49].

Additionally, IVs often struggle in unfamiliar or uncertain environments, such as unmarked roads, variable traffic conditions, or obstructed views. As [50] highlights, current research tends to prioritise criticality and search efficiency over comprehensive scenario coverage, leaving a gap in the validation of IV safety systems [48].

High-Definition Maps and Road Work Adaptation

Accurate perception of the driving environment is essential for the safe operation of IVs. As [51] notes, achieving comprehensive and up-to-date knowledge of traffic conditions is critical, which has led companies like Waymo to adopt HDMs as a core component of their navigation systems.

However, HD maps require frequent and precise updates to reflect real-time changes, such as road construction, lane reconfigurations, or temporary closures. Maintaining this level of accuracy is both technically demanding and financially costly, as it necessitates the deployment of fleets of sensor-equipped vehicles to continuously scan and update the environment.

Cybersecurity Threats

Beyond the challenges of perception and navigation, IVs are increasingly vulnerable to cybersecurity threats. As highlighted by [52, 53] these threats include sensor

spoofing and hacking, particularly targeting critical components such as cameras, LiDAR, and GPS, as well as the interception or manipulation of vehicle-to-everything (V2X) communications.

In addition, [54] emphasises that data privacy and protection remain significant barriers to the widespread deployment of IVs. These concerns necessitate the development of robust, multi-layered cybersecurity frameworks capable of safeguarding both vehicle systems and user data [48].

Social Responsibility Considerations

In addition to regulatory, safety, and cybersecurity concerns, the development and deployment of IVs must also address a range of social responsibility challenges. As noted by [55], these include potential biases in detection algorithms, limited accessibility for low-income or underserved populations, ethical concerns regarding data usage, and the displacement of jobs in sectors such as transportation and logistics [48, 44].

Addressing these issues requires a multidisciplinary approach that incorporates perspectives from ethics, public policy, and social equity to ensure that IV technologies are developed and deployed in a manner that is inclusive, fair, and socially sustainable.

Navigating Dynamic and Complex Environments

Despite substantial advancements, IVs continue to face challenges in navigating dynamic and unpredictable environments [56]. As [57] observes, IVs remain particularly vulnerable to edge cases—scenarios involving erratic driver behaviour, interactions with emergency vehicles, and complex or poorly marked road conditions.

The California OL 316 Reports suggest that current U.S. IV prototypes may overstate their operational capabilities, highlighting the need for further refinement before large-scale deployment [6]. Similarly, evaluations of Waymo's performance reveal limitations in collision avoidance, especially in response to unpredictable human actions [58].

Although human error remains the leading cause of IV-related accidents [59], autonomous systems must significantly improve their ability to anticipate and respond to non-standard behaviours, such as failure to yield, sudden lane changes, or disobedience of traffic signals [57, 60].

Traffic Hazard Events

Traffic hazard events represent scenarios that pose varying levels of risk to road users and can be classified into three progressive stages [61]:

- Potential Events: Situations where no immediate collision threat exists, but the conditions could evolve into one (e.g., a cyclist rapidly approaching an intersection).

- Developing Events: Scenarios where a collision threat is imminent and requires immediate corrective action (e.g., the cyclist does not decelerate, prompting evasive manoeuvres).

- Materialised Events: The collision has occurred.

For IV systems to be effective, they must be capable of early detection and response to potential events, thereby preventing escalation into developing or materialised stages. However, most current systems rely heavily on Time-To-X metrics (e.g., Time to Collision), which often lack the predictive depth required to manage the complexity of highly dynamic traffic environments [60, 62].

Limitations of Current IV Research

A review of current mono-camera IV systems designed to learn traffic road users' behaviour reveals several critical limitations:

- **Limited focus on discrete intention recognition**: Existing research has paid insufficient attention to recognising and predicting the specific, discrete intentions of traffic agents.
- **Narrow scope of traffic agent types**: Most studies have focused exclusively on pedestrians and vehicles, neglecting other important traffic participants such as cyclists, traffic lights, and animals.
- **Restricted behavioural diversity**: While vehicles can exhibit a wide range of behaviours—such as cutting in, reversing, turning, emerging, and stopping—the majority of current research has concentrated primarily on lane change behaviour, overlooking other critical manoeuvres.
- **Insufficient datasets for discrete intention behaviour**: There is a notable lack of datasets specifically designed for training and evaluating machine learning algorithms to recognise and predict the discrete intention behaviours of traffic agents. Moreover, existing datasets do not adequately represent heterogeneous traffic participants in complex urban environments.
- **Absence of algorithms for heterogeneous intention recognition**: To the best of the authors' knowledge, no existing algorithms have been developed to recognise the discrete intention behaviours of heterogeneous traffic agents, such as cyclists, animals, or traffic light interactions.
- **Neglect of object-side visibility cues**: No prior studies have addressed the importance of recognising which side of a target object is visible to the ego vehicle (EV). For instance, visibility of the right side of a vehicle may indicate that it is about to cross the EV's path or is emerging from a left-side intersection.
- **Underutilisation of vehicle light signals**: Although vehicles communicate intentions through light signals—such as indicators, brake lights, reversing lights, and hazard lights—most previous research has only implicitly considered these cues, without explicitly modelling them in behaviour recognition systems.
- **Bias toward materialised hazard events**: Existing datasets predominantly contain traffic hazard events that are already in the materialised stage, limiting the ability to train models to detect and respond to earlier, preventable stages.

- **Lack of multi-class hazard event recognition**: There is currently no research that investigates the recognition of multiple distinct types of traffic hazard events, which is essential for comprehensive risk assessment.
- **Limited exploration of full pipeline systems**: Few studies have examined a complete behaviour recognition pipeline, encompassing detection, tracking, and recognition. This is critical, as the cumulative uncertainty from each module can significantly impact the final prediction accuracy.

In conclusion, IVs have made considerable advancements, but key challenges remain in regulatory compliance, safety standards, cybersecurity, environmental adaptation, and predictive behaviour modelling. Future research should address these gaps by developing more comprehensive datasets, enhancing AI-driven scenario modelling, and refining safety-critical event recognition to improve the reliability and trustworthiness of autonomous vehicles.

1.2 Research Questions

From the above discussion and the current limitations in IVs, the following research questions were formulated:

Question 1: *"What are the limitations of existing datasets used to study complex traffic agent behaviour in urban scenarios, and how can these limitations be effectively addressed?"*

Question 2: *"Can the performance of traffic agent behaviour recognition algorithms be improved by introducing novel input features (e.g., object visible side from the EV's perspective, alternative image representations, or explicit feature encoding)?"*

Question 3: *"Are machine learning algorithms capable of accurately recognising different rear light statuses and object visible sides when trained on an appropriately designed dataset?"*

Question 4: *"Can machine learning models, when trained on a dedicated traffic hazard dataset, effectively recognise various types of traffic hazard events in complex urban environments?"*

Question 5: *"How does the performance of behaviour recognition algorithms change when integrated into a complete pipeline system, and what are the implications of module-level uncertainties on overall system accuracy?"*

1.3 Aims and Objectives

This research aims to address the limited capability of existing mono-camera IV systems to navigate safely in complex urban scenarios by enabling them to recognise traffic agent behaviours prone to potential traffic hazard events.

To address the research questions posed above and achieve the project's aim, the following objectives have been formulated:

- *Objective 1*: Explore existing methods for recognising and predicting the behaviour of traffic agents, as well as detecting traffic hazard events within traffic scenarios.

- *Objective 2*: Create a Road Traffic Hazard Dataset.
- *Objective 3*: Develop, implement, and evaluate a Vehicle Rear Light Signal Recognition algorithm.
- *Objective 4*: Develop, implement, and evaluate an Object Visible Side Recognition algorithm.
- *Objective 5*: Develop, implement, and evaluate a Vehicle Intention Recognition and Prediction algorithm.
- *Objective 6*: Develop, implement, and evaluate a Potential Traffic Hazard Event Recognition algorithm.
- *Objective 7*: Develop, implement, and evaluate a Complete Potential Traffic Hazard Event Recognition Pipeline System.

1.4 Contributions

Based on the current limitations of IVs and the research questions outlined above, the contributions of this research are as follows:

1.4.1 Novel Dataset

- Created a novel dataset by combining existing public datasets to address the challenge of recognising potential traffic hazard events in IVs.
- Introduced the first Potential Traffic Hazard Dataset, featuring 16 distinct types of traffic hazards.
- The dataset includes 600 videos of traffic hazard samples and 300 videos of no-hazard samples. Each video is annotated with detailed information, such as object detection, object tracking, rear light status, object visible side from the EV's perspective, lane detection, and event timing (start and end).
- Designed to be versatile, this dataset supports various research studies, including detection, tracking, rear light status recognition, object visible side analysis, and hazard type recognition.

1.4.2 Overlooked Cues

- Developed deep learning algorithms specifically for recognising overlooked cues, such as the rear light status of target vehicles (TV)s and the visible side of objects relative to the EV.
- Focused exclusively on classifiers developed after 2020 to ensure the use of the latest advancements in the field.

1.4.3 Vehicle Intention Recognition and Prediction Algorithm

- Enhanced existing algorithms by incorporating manually extracted features, allowing for a deeper investigation into their performance. This method enables explicit consideration of critical inputs such as rear light indicators and object's visible side, which were only implicitly addressed in previous studies.

- Introduced a novel metric for evaluating the effectiveness of vehicle intention recognition and prediction algorithms.

- Conducted an ablation study to assess the impact of different components on the algorithm's performance.

1.4.4 Traffic Hazard Event Recognition Algorithm

- Employed various machine learning models to recognise different types of potential traffic hazard events, utilising the proposed novel dataset for evaluation.

- Proposed two new input image representations:

  - The Object of Interest (OOI) input image, which includes only the detected objects and the road.
  - The blended input image, which combines RGB and depth information into a single image.

- Unlike previous studies that implicitly considered discrete features such as rear light indicators and object visible side, this research explicitly integrates them into the potential traffic hazard event recognition process.

- Utilised a state-of-the-art Transformer network in conjunction with LSTM to recognise traffic hazard events.

1.4.5 Complete Potential Traffic Hazard Event Recognition Pipeline

- Developed a comprehensive potential traffic hazard event recognition pipeline, incorporating modules for detection and tracking, rear light recognition, object visible side recognition, and potential traffic hazard event recognition.

- The pipeline allows for a detailed analysis of which modules require further improvement and how each stage impacts the final result.

- Provides a more realistic assessment of algorithm performance in real-world scenarios, contributing to the design of more effective IV systems.

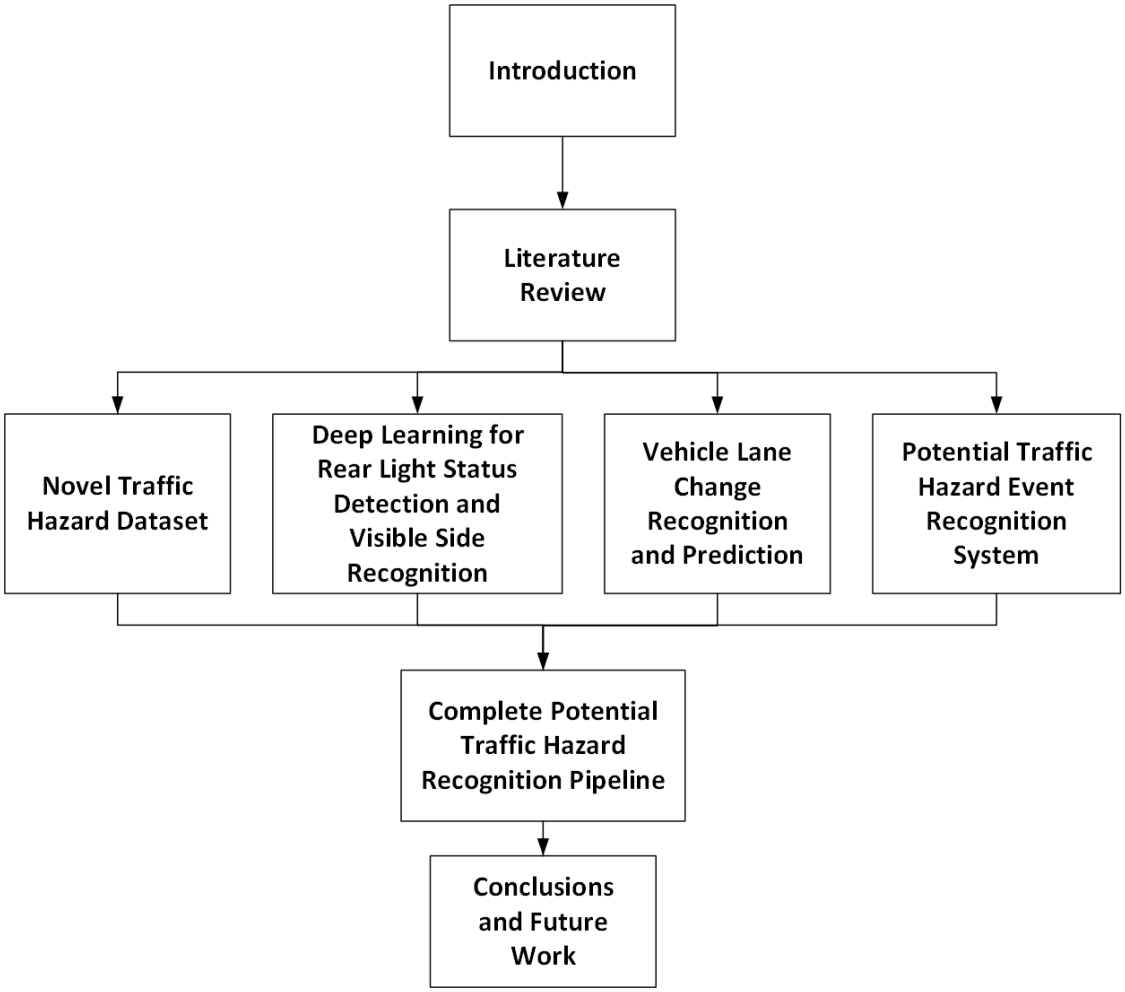


Figure 1.1: Thesis chapters flow diagram.

1.5 Thesis Outline

The flow diagram depicted in Figure 1.1 was created to provide a clearer understanding of the chapters included in the thesis and how they are interconnected.

The literature review was used to identify gaps in the existing body of work, which led to the development of the novel traffic hazard dataset, deep learning models for rear light status detection and visible side recognition, vehicle lane change recognition and prediction, the potential traffic hazard event recognition system, and the complete potential traffic hazard event recognition pipeline. The results and discussions of each chapter informed the conclusions and future work chapter. Overall, the thesis is structured as follows:

- **Chapter 2**: Provides an overview of IVs, including their history, benefits, challenges, and system architecture. It presents an in-depth review of existing algorithms and datasets for recognising and predicting traffic agent behaviours, as well as current methods for detecting road traffic hazards. Additionally, it briefly discusses techniques for detecting and recognising rear lights and their status.

- **Chapter 3**: Introduces the novel traffic hazard dataset, detailing the methods

used for data collection and labelling, and describing the dataset's format and statistical characteristics.

- **Chapter 4**: Presents the rear light status and object visible side recognition systems, which address cues that have been overlooked in previous research.
- **Chapter 5**: Describes the proposed vehicle intention recognition and prediction system, which incorporates manually extracted features and introduces a novel prediction metric.
- **Chapter 6**: Presents the proposed traffic hazard event recognition system, which utilises various deep learning algorithms and input representations.
- **Chapter 7**: Describes the complete potential traffic hazard event recognition system, including modules for detection and tracking, rear light status recognition, object visible side recognition, and traffic hazard event recognition.
- **Chapter 8**: Summarises the key findings, discusses their implications, and outlines the final conclusions and suggestions for future work.

1.6 Written Articles

During the thesis, two literature review papers were published and three contribution papers were submitted. Below are the paper's title and references:

- Galvão, Luiz G., et al. "Pedestrian and vehicle detection in autonomous vehicle perception systems—A review." Sensors 21.21 (2021): 7267.
- Galvão, Luiz G., and M. Nazmul Huda. "Pedestrian and vehicle behaviour prediction in autonomous vehicle system—A review." Expert Systems with Applications (2023): 121983.
- Galvão, Luiz G., and M. Nazmul Huda. "Deep Learning Vehicle Intention Prediction for Autonomous Vehicles Using Onboard View Data" Expert Systems with Applications. (Under Review)
- Galvão, Luiz G., and M. Nazmul Huda. "An AI System and a Novel Dataset for Recognising Potential Traffic Hazard Events in Complex Traffic Scene" Expert Systems with Applications. (Under Review)
- Galvão, Luiz G., and M. Nazmul Huda. "A Complete Potential Traffic Hazard Event Recognition System for Intelligent Vehicle Systems" Neurocomputing. (Under Review)

Chapter 2

Literature Review

This chapter aims to provide the research's background and context, identify gaps in the literature, and draw insights from previous works. It begins with a list of key terms and their definitions commonly used in the realms of IVs, traffic agent behaviour, and road traffic hazards. Next, it briefly overviews Intelligent Vehicle (IV) systems, covering their history, benefits, challenges, and system architecture. For a more comprehensive literature review on IVs, please refer to [63]. The chapter then conducts an in-depth review of existing algorithms and datasets for recognising and predicting traffic agents' behaviours and current methods for detecting road traffic hazards. Finally, it reviews the detection and recognition status of vehicle rear lights.

Some of the commonly used terminologies in the pedestrian and vehicle behaviour prediction and traffic hazard detection literature are listed below [64]:

- **IVs**: An IV is an automobile equipped with advanced systems integrating perception, decision-making, and control mechanisms to automate various driving functions, including lane-keeping, obstacle avoidance, overtaking, maintaining traffic flow, responding to hazards, and planning optimal routes. [65, 66].
- **Observation horizon (OH):** The duration over which an algorithm analyses past behaviours of an object to predict its future behaviour.[67, 68, 69, 70].
- **Prediction horizon (PH):** Prediction horizon refers to the time frame within which an algorithm forecasts an object's future behaviour before it occurs [67, 68, 69, 70]. Some works differentiate between prediction and anticipation, where the latter refers to predicting behaviour before it begins [71]. This project adopts the former definition.
- **Ego Vehicle (EV):** The vehicle equipped with onboard sensors that perceive other traffic agents and surroundings. [64, 70].
- **Target Object (TO):** The specific object that the EV observes to predict its behaviour. [64, 70].
- **Surrounding Objects (SO):** Objects in the environment that can interact with or influence the behaviour of the target object [64, 70].
- **Multi-modal behaviour:** A situation where an observed history of behaviours leads to multiple possible future behaviours [64, 70].

- **Object behaviour:** The movement patterns or intentions exhibited by an object [64, 70].
- **Object trajectory:** A sequence of data points representing an object's movement path, typically derived from tracking information [64, 70].
- **Trajectory prediction:** Refers to predicting an object's future motion within a given time frame based on its past trajectory, contextual information, and interactions with surrounding objects. [64, 70].
- **Object intention:** The planned course of action an object intends to take to achieve a goal. In the vehicle domain, these are known as manoeuvres (e.g., turning, lane-changing, stopping), while in the pedestrian domain, they include crossing, stopping, etc [64, 70].
- **Intention prediction:** Uses historical trajectory and contextual information to predict an object's future discrete actions rather than its precise trajectory [64, 70].
- **Interaction:** the influence that one or more objects exert on each other's behaviour [70].
- **Cues:** observable characteristics used to infer a traffic agent's behaviour, for example, past motions, scene context, interactions between multiple agents, edges, gradients, coloured segments, or colour distribution [72, 73, 74].
- **Explicit feature**: input features that are clearly stated, such as object speed, position, and type. Another example would be the indicator light status of a vehicle predicted by CNN that is specifically trained and evaluated for this purpose [75, 76].
- **Implicit feature**: An input feature that is not explicitly stated and must be inferred, such as the status of a vehicle's indicator lights from a raw RGB image [77, 75].
- **Explicit factors**: factors affecting a traffic agent's behaviour that are evident, for example, weather, road condition, lane marking, etc [60].
- **Implicit factor**: factors affecting a traffic agent's behaviour that are not obvious, for example, traffic agent behaviour [60].
- **Traffic scenario**: is a particular event within the context of road traffic, for instance, driver manoeuvre, traffic accidents, traffic rules, pedestrian intention, and surrounding vehicles' intention [78].
- **Taillight lights**: Red lights at the rear of a vehicle that illuminate when the headlights are on, improving visibility in low-light conditions [79].
- **Brake lights**: Red lights at the rear of a vehicle that illuminate when the driver applies the brakes, signalling deceleration or stopping [79, 80].
- **Rear indicator lights**: Amber or yellow lights at the rear of a vehicle that flash when the driver signals a turn, lane change, or hazard [80, 81].

- **Rear lights**: A general term referring to taillights, brake lights, and rear indicator lights [82, 83].
- **Rear lights detection**: The process of detecting single or multiple rear lights in an image, localising them with bounding boxes [81].
- **Rear light pairing**: The process of identifying and matching the left and right rear lights of the same vehicle [84].
- **Rear lights recognition status**: Determining whether any of the rear lights are 'ON' or 'OFF' in an image [81].

2.1 Intelligent Vehicle Systems

The concept of IVs originated around 1920. During this period, they were often referred to as "phantom autos" because they lacked a human driver and were instead remotely controlled [85]. AVs only progressed in the 1980s when [86] created the project "Autonomous Land Vehicle In a Neural Network", where they concluded that neural networks could significantly contribute to autonomous navigation systems. The Defence Advanced Research Projects Agency (DARPA) organised the first Grand Challenge in 2004 to motivate research and development of AVs. DARPA organised other grand challenge events in 2005 and 2007, also known as the Urban Challenge [87]. In 2008, Rio Tinto started the trials of an autonomous haul truck fleet to transport ore and waste material in Pilbara. Nowadays, they have more than 130 autonomous trucks [88]. In 2009, Google secretly started developing its first IV, and they passed the first self-driving test on 1 May 2012 in Las Vegas [89]. The UK government launched a driverless competition in 2014 to support and encourage IV [5]. Between 2010 and 2017, major automotive manufacturers such as General Motors, BMW, Nissan, Volkswagen, Tesla, Volvo, Mercedes-Benz, Toyota and Audi recognised the potential benefits of AVs; therefore, they adopted the concept and started their research and development [90, 91]. In 2019, the European Parliament and Council released the Regulation (EU) 2019/2144 for the first time, specifying requirements associated with automated and fully automated vehicles [92]. A big step for IV was achieved when Waymo (Google's self-driving vehicle became Waymo in 2016) reported that their "Waymo Driver" reached 20 million self-driven miles and 15 billion simulated miles [24]. This is an essential achievement since these self-driven miles are considerable training experiences that can be used as a dataset for other AI systems.

2.1.1 Benefits and Challenges

AVs are expected to offer many benefits. For example, they can follow traffic laws and respond quickly to unexpected scenarios. Therefore, a significant reduction in road traffic accidents is anticipated since most accidents are caused by human error and imprudence. AVs are also expected to predict the behaviour of vehicles, enabling them to reduce braking, acceleration, and consequently, fuel consumption, air pollution, traffic shock-wave propagation, and congestion [93]. However, they face many challenges that need to be addressed. For instance, AVs could replace

taxi, truck, and bus drivers, potentially leading to an increase in unemployment. Table 2.1 presents additional benefits and challenges of IV systems.

Table 2.1: Benefits and implications of AVs [94, 93].

Benefits	**Challenges**
• It is expected that AVs will be able to have vehicle to vehicle/infrastructure (V2V, V2I) communication, therefore, it would be able to choose more efficient routes, reduce or even remove intersection delays and collisions; • Make the best use of the road lanes by maintaining short gaps between vehicles; • Improve social inclusion since unlicensed, young, disabled and elderly people would be able to use them; • Improve freight transportation, for example, travel long distances in less time, offer cheaper freight since drivers are not required, and trucks would drive more efficiently; • Increase economic opportunities; • Reduce parking space; • Enhance driver experience by offering comfort during the trip by avoiding harsh braking and jerking; • It is estimated that each year, driver spend 6 working weeks driving [5]. IV could enable people to have free time to relax or work while going to his/her destination; • Less CO_2 emission; • IV is expected to reduce traffic accidents, resulting in lower expense for legal proceedings and compensation. Additionally, car insurance prices are anticipated to decrease.	• Currently, human drivers are better in recognising pedestrians, cyclists, and other small traffic objects; • Human driver is better at recognising different types of materials, for example, if an object ahead is made of cardboard, wood, or concrete. • A reasonable quantity of AVs is required to validate their benefits; • Not everyone will be able to afford IV technology; • Since unlicensed young, elderly, and vulnerable people will be able to travel, this would increase the number of trips and could cause more congestion; • Initially, the public may have some resistance to accept and become comfortable with AVs [95]; • Initially, conventional drivers would not predict IV behaviour. • Creating new legislation, regulations, certification, testing standards and insurance for AVs; • Security against cyber-attacks.

2.1.2 IV Taxonomy

Society of Automotive Engineers (SAE) created the J3016-2018 guidelines outlining the taxonomy and definitions for driving automation systems. The document describes the six levels of driving automation: in Level 0, there is no automation; in Level 1, there is some automation assistance, such as ADAS features that can control the steering or speed. However, the driver is responsible for supervising and acting when required. Level 2 enables partial driving automation, where the autonomous system can control both steering and speed; however, the driver is still responsible for observing the environment and supporting the autonomous system. Level 3 enables conditional driving automation, where the car is fully automated when certain conditions are met, such as good weather and visibility. When conditions are not favourable, the driver must be in control; Level 4 enables high automation where the automated system does not require the driver to be in control; however, the system only works if certain conditions are met; and Level 5 enables full driving automation where the automated system is always under control and can drive in any condition [96, 97].

2.1.3 Intelligent Vehicles System Architecture

The functional requirements of an IV can be compared to those of an autonomous mobile robot system, requiring perception, communication, localisation, path planning, trajectory, and motion control [98, 99, 100, 101]. In IV systems, these requirements are commonly categorised as “sense, plan, and act,” or more formally,

"perception, planning, and control" [98]. While typical IV system architectures include these modules, the Waymo Driver system introduces an additional behaviour prediction module preceding the planning module [24]. This work adopts the Waymo Driver approach, recognising behaviour prediction as a distinct module within the IV system. The adopted IV system architecture is illustrated in Figure 2.1.

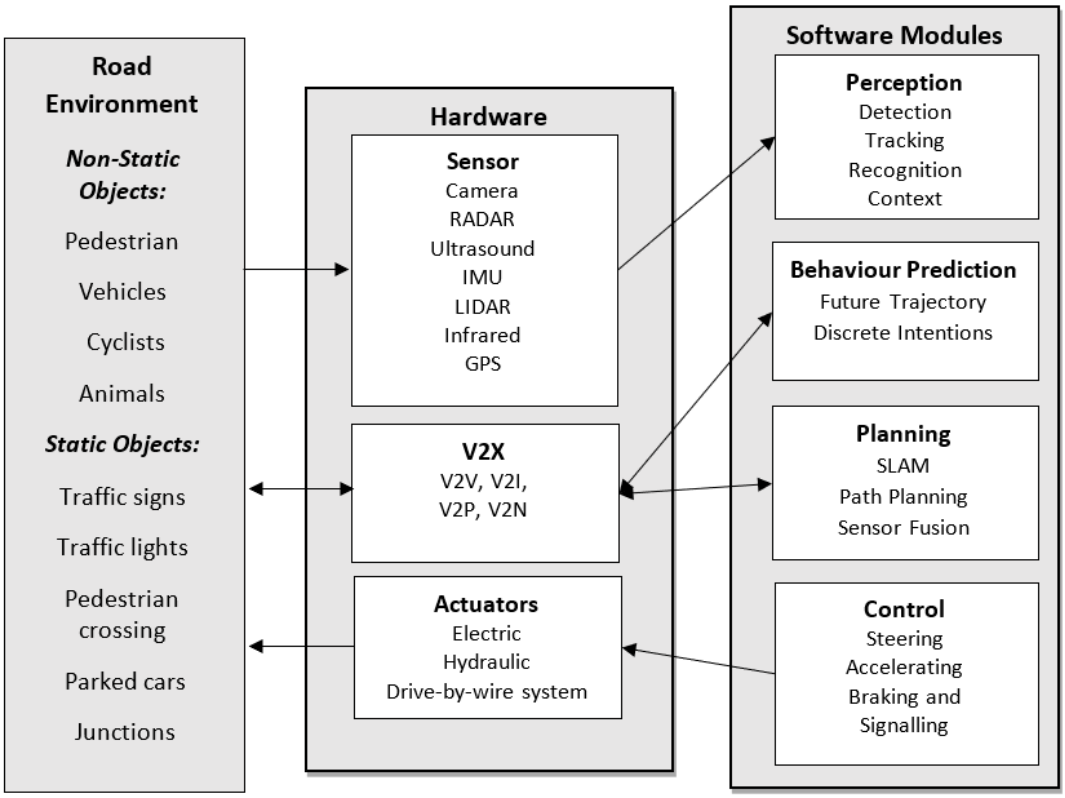


Figure 2.1: IV architecture: hardware requirements are sensors, V2X communication device and actuators. Software modules include perception, behaviour prediction, planning, and control. Modified version from [98].

According to [24], the listed functional requirements above should answer the following questions:

- "Where is the IV?"
- "What is around the IV ?"
- "What will happen next ?"
- "What should the IV do ?"

Perception

The perception module should answer "Where is the IV?" and "What is around the IV ?". For example, detection and tracking information about statics (e.g., traffic lights, traffic signs, road works, etc.) and non-static (e.g., pedestrians, vehicles, etc.) objects, and traffic road contexts information (e.g., road lanes, edges, curbs, pedestrian crossing, etc.). Furthermore, the raw data enables the IV to execute the Simultaneous Localisation and Mapping (SLAM) task [96].

Passive (receive and measure existing energy) or active sensors (measure reflected signals transmitted by it) can be used to perceive the environment. Passive sensors generally used are Charge-Coupled Devices (CDD) or Complementary metal-oxide semiconductors (CMOS) cameras. Active sensors generally used are Light Detection and Ranging (LIDAR), long/medium/short Radio Detection and Ranging (RADAR), ultrasound/Sound Navigation System (SONAR), Inertial Measurement Unit (IMU) and Global Navigation Satellite Systems (GNSS).

The advantages and disadvantages of each sensor are described in Table 2.2. It is observed that each sensor has its strengths and weaknesses. For example, radar sensors work well in dark environments, are not affected by extreme weather, and can accurately detect speed but have low resolution. An approach to overcome the deficiency of each sensor is to use sensor fusion technology, where data from multiple sensors are combined to attain enhanced information.

Table 2.2: Advantages and disadvantages of various sensors used in AVs [96].

	Advantages	**Disadvantages**
CAMERA	• Low cost. • Technology is mature. • High Resolution. • Possible to generate a 3D stereoscopic view. • Detect RGB information. • Road markings and signs are designed for human eyes. • Less chance of being affected by interference from another vehicle. • Wider field of view.	• Short range. • Performance decreases in poor weather and low light conditions. • Does not provide accurate distance and position of objects. • Usually does not provide depth information. Depth information can be acquired but make the system more complex (e.g., stereo camera and disparity estimation algorithms).
THERMAL	• Distinguish between hot and cold targets. • Can be used during the day and night.	• More expensive than cameras. • The main target is pedestrians, but they can get confused with hot air from the exhaust pipe or other objects that generate heat. • Cannot detect heat through the glass, for example, detect drivers inside the car.
INFRARED	• It can be used during the day and night. • They are cheaper and small.	• It has a short range.
SONAR	• Cheap.	• Affected by poor weather conditions. • It is short-range and commonly used for automated parking and blind spot detection features.
RADAR	• Its performance is not affected in bad weather or low light conditions. • Long range (up to 300 m). • Provide accurate distance, position, and speed. • Technology is mature. • Low cost	• Limited resolution. • Limited to recognise objects.
LIDAR	• 360-degree view of the environment. • Wide field of view. • High accuracy.	• Low spatial resolution compared to cameras. • Performance decreases in poor weather. • Complex and requires high processing power. • Expensive. • Affected by interference and external light. • Does not provide colour information. • Acquired information is sparse.

Recently, IV systems have been implemented using two main methods: a pure vision-based approach where only cameras and computer vision techniques are employed, and a sensor fusion approach where information from multiple sensors and computer vision techniques are used (e.g., cameras, LIDAR, RADAR, etc.) [102]. For example, Tesla uses a pure vision-based technique to acquire information from the traffic scene, whereas Waymo uses computer vision and fusion of advanced sensors. The primary advantages of a pure vision-based system are manifold. Firstly, many modern vehicles come equipped with cameras, facilitating easy integration.

Once an AI computer vision system is developed, it can be readily deployed. Additionally, cameras are more cost-effective than LIDAR systems and offer higher resolution. Unlike LIDAR, which requires pre-mapping the environment, a pure vision-based system operates in real-time, providing immediate processing. For these reasons, the subsequent sections of this paper predominantly concentrate on a pure vision-based approach.

Behaviour Prediction

The behaviour prediction module is responsible for answering "What will happen next ?". For example, predict future traffic agents' behaviour (e.g., trajectories and intentions). This module is the focus of this literature review, and an in-depth discussion will be done in future sections of this chapter.

Planning

The plan module takes the perceived and predicted information to decide what actions and paths the IV should make to achieve its final goal. It should answer "What should the IV do?" questions. The planning stage is subdivided into three tasks: mission, behaviour, and motion planner [98]. The mission planner is responsible for assigning a goal (e.g., pickup/drop-off task) to the IV and choosing the best routine to complete the assigned goal. The behaviour planner considers the interaction between other traffic agents and the available traffic rules to decide what behaviour the IV should perform, for example, should the IV change lane, stop, or turn. Ultimately, the motion planner is tasked with generating collision-free paths to execute the behaviours predetermined by the behaviour planner. The planning stage has been implemented using traditional techniques such as the Voronoi diagram, occupancy grid algorithm, or driving corridors diagram. However, these approaches are unsuitable for complex urban scenarios where the interaction between traffic agents and traffic rules needs to be considered. Many researchers have used machine learning (ML) such as CNN, Deep Reinforcement Learning, or hybrid systems where ML and traditional techniques are jointly used [96].

Act

The control stage uses the information acquired from the planning stage to perform the actual movements of the IV, which are conducted by sending steering, acceleration, braking, and signalling commands to the actuators. The drive-by-wire system is the most appropriate and advanced way to transfer the commands to the actuator. The control system generates and tracks trajectories and uses controllers to perform the desired trajectories. Trajectories generation is usually achieved either sensor-based or dynamics-based. Sensor-based approaches are more suitable for robotics, while dynamic-based approaches are suitable for vehicles. The most used methods to track trajectories are geometric or model-based. A feedback controller such as Proportional-Integral-Derivative (PID) is usually utilised to ensure the IV is not deviating from the target trajectories. However, feedback controllers have their limitations, for example, the system will only respond to errors when they occur [98]. Two degrees of freedom controllers, a combination of feedback and feedforward controllers, have been proposed to overcome the limitations of the feedback

controller. In this type of controller, a model reference of the system is also used, which helps the system to predict the IV motion with more details.

2.2 Behaviour Recognition and Prediction

There are several literature reviews covering both traditional and DL techniques to predict the behaviour of vehicles, for example, [103, 104, 105, 64]. Sivaraman and Trivedi [106] briefly reviewed the behaviour prediction of vehicles, but at that time, this topic was fairly new, and only traditional techniques were reviewed. Lefèvre, Vasquez, and Laugier [103] presents a survey and classifies vehicle prediction behaviour algorithms into physics-based, manoeuvre-based, and interaction-aware-based algorithms. They concluded that a behaviour prediction algorithm needs to consider the interaction between vehicles and the scene context to have a longer prediction horizon. In addition, they reviewed the existing risk assessment methods for IVs and concluded that a risk assessment module is highly dependent on the behaviour prediction algorithm. In this review, the authors only covered traditional techniques since DL techniques for vehicle behaviour prediction were still emerging. Shirazi and Morris [104] reviewed techniques used to analyse vehicles, drivers, and pedestrians' behaviour at road intersections. However, only traditional techniques were analysed, and the focus was not on the prediction behaviour of vehicles. Leon and Gavrilescu [105] reviewed methods used for vehicle tracking, behaviour prediction, and decision-making. Both traditional and DL techniques have been covered. The authors concluded that DL techniques have better results since they are more robust, flexible and have better generalisation ability. Mozaffari et al. [64] performed a systematic and comparative review of the different DL methods used to predict vehicle trajectories and intention behaviour. They presented a more detailed taxonomy of the prediction behaviour algorithms compared to [103]. They classified the algorithms by what input information was used, the output type produced, and the prediction method used. Although the review was extensive and very informative, the authors do not detail what intention behaviour the works were trying to predict, for example, lane change, overtaking, or making a turn, and do not provide specific information on what dataset was used.

The following works have performed pedestrian behaviour prediction reviews, [107, 108, 109, 110, 111, 112]. Kong and Fu [107] presented traditional and DL techniques used to recognise and predict human action. Ahmed et al. [112] presented a survey on the detection and intention prediction of pedestrians and cyclists. A review on pedestrian behaviour was given by [108], where they briefly described the traditional and DL techniques that have been used. Chen et al. [113] discussed the required architecture, the traditional and DL techniques to detect and predict pedestrian actions. Although these works have reviewed DL techniques, only a limited amount of work was considered. A detailed human trajectory prediction survey was done by [109], where they reviewed a substantial amount of published works to propose a taxonomy, identify the available datasets and evaluation metrics, and the limitations of the current methods. However, the authors did not review methods used to predict pedestrian intentions. A comprehensive survey was done by [111] on pedestrian intention prediction for IV systems.

To the author's knowledge, [74] is the only work that has reviewed behaviour prediction for both pedestrian and vehicle agents. While the authors propose a

novel taxonomy that unifies the behaviour prediction challenges for pedestrians and vehicles, they do not delve into the evaluation metrics, datasets, features, or results achieved by the reviewed studies.

Differing from all the previously cited review works, both for pedestrian and vehicle behaviour prediction, this paper:

- Presents a behaviour prediction general problem formulation.
- Presents the most used terminologies in the pedestrian and vehicle behaviour prediction domain.
- Reviews not only pedestrian or vehicle behaviour prediction algorithms but both of them.
- Briefly presents the most important traditional techniques and focuses more on the DL techniques for pedestrian and vehicle prediction algorithms.
- Summarises the main information acquired from the reviewed pedestrian and vehicle behaviour prediction works in tables. The tables report the methods, the problems the algorithms are trying to solve, the datasets used, and the results acquired.
- Reviews works that have performed prediction behaviour of heterogeneous agent traffic.
- Reviews the works to identify the requirements and challenges to design a pedestrian and vehicle behaviour prediction system for IV. The techniques that have been proposed and whether or not they meet the previously identified requirements, and if not, propose future works.

2.2.1 Behaviour Prediction General Problem Formulation

A complete pipeline of an object behaviour prediction system, as depicted in Figure 2.2, comprises detection and classification, tracking, and prediction stages. This paper only reviews works that studied the behaviour prediction stage. Literature reviews on the detection and tracking can be found in the following works [101, 114, 115, 116, 117, 118, 119].

Based on the vehicle and pedestrian intention prediction problem formulation proposed by [69, 120, 121, 122, 123, 124, 125, 126, 127, 128, 129, 130, 131, 132, 133, 134], and [135], a general intention prediction problem formulation is as follows: a sequence of feature vector $\{F_{t-OH}, ..., F_t\}$ extracted from a given sequence of video frames $\{t-OH, ..., t\}$ acquired from an image sensor is used by a model to determine the probability of the target agent intention $I_a^{t+n} \epsilon \{0, 1\}$, where t is the specific time of the last observed frame and n is the number of frames from the last observed frame to the final frame of the event, also known as time-to-event (TTE). The prediction intention estimation can be described by the equation

$$p(I_a|F_{t-T_{obs}:t}). \tag{2.1}$$

Based on the vehicle and pedestrian trajectory prediction problem formulation proposed by [136, 137, 138, 139, 140, 141, 142, 73, 143, 144, 145, 146, 147, 148, 149,

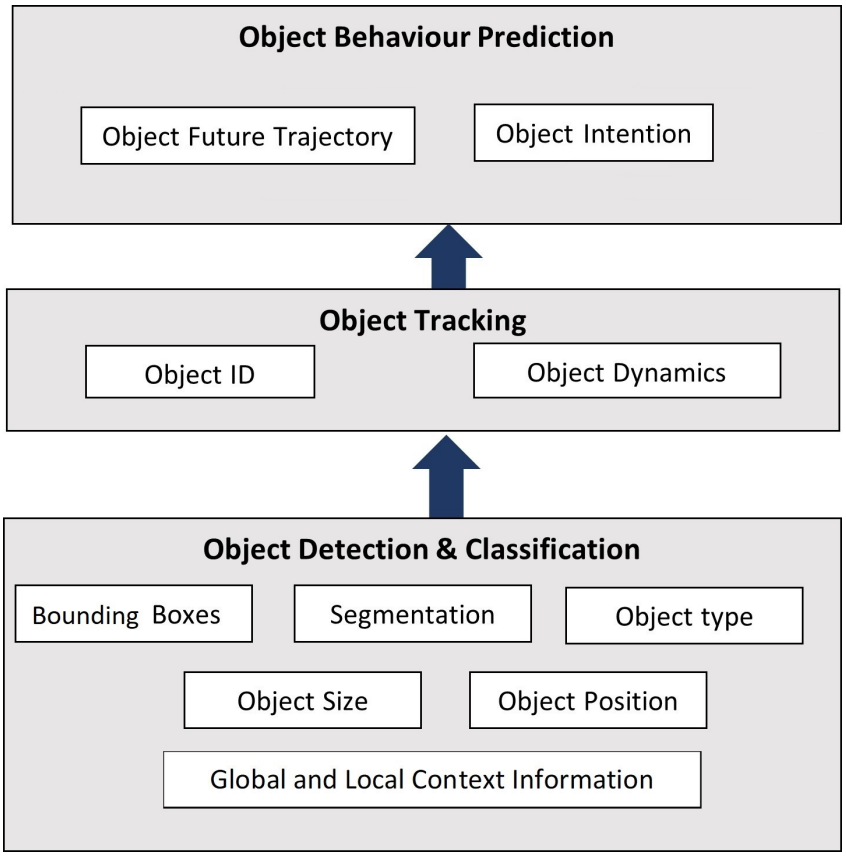


Figure 2.2: Object behaviour prediction complete pipeline process. The detection and classification stage outputs the object position, size, type, bounding box, segmentation, and global and local context information. The object tracking stage outputs the ID for each detected object and its dynamics (e.g., speed). The output of the object behaviour prediction module can be the object's intention and its future trajectory.

150, 151, 152], and [153], a general trajectory prediction problem could use the same sequence of feature vector used by intention prediction algorithm. However, in this case, the sequence's purpose is to predict the future path of the target agent, spanning up to the specified PH. The trajectory prediction estimation can be described by the equation

$$p(FuturePath_{t:PH}|F_{t-T_{obs}:t}). \tag{2.2}$$

Figure 2.3 depicts an example of predicting a vehicle's lane change manoeuvre. Here, image sequences from $t - N$ to t are used to extract a sequence of feature vectors, which are subsequently used to make predictions. In this case, a successful lane change manoeuvre prediction occurs when the vehicle's intention is correctly recognised before reaching the F1 stage.

The differences among the reviewed problem formulations of vehicle and pedestrian behaviour prediction are:

- Some problem formulations are for trajectories and others for intention.
- The input features used may be different, for example, some authors have used only position and speed, while others have used local and global context vectors.

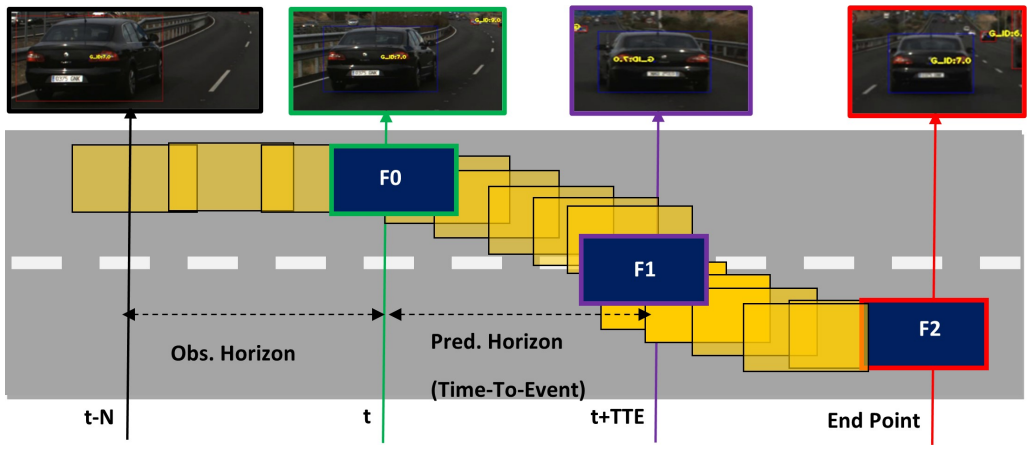


Figure 2.3: An example of lane change prediction problem: F0 is where the vehicle manoeuvre starts, F1 is where the actual manoeuvre happens, and F2 is the end of the manoeuvre.

- Some considered top-view, and others considered on-board view datasets.
- Authors have used different predictive models.

The listed items above are further discussed in the following sections 2.2.2, 2.2.3, and 2.2.4.

In the literature, some works use predicted intentions to improve the accuracy of the future trajectories, and other works use predicted trajectories to improve the accuracy of the predicted intention [64, 69]. These works will be discussed in the upcoming sections.

Before discussing the behaviour prediction of pedestrians and vehicles, it is important to understand their potential interactions. As depicted in Figure 2.4, interactions among different traffic agents can cascade and get very challenging, for example, to predict the actions of object 1, it might be required to consider the actions of the:

- Object 2, since it can change direction and velocity.
- Object 3, since its action will affect the action of object 2.
- Object 17, since it will affect the action of object 3.
- Object 9, since it will affect the action of object 2.
- Object 13, since it can make a right turn, which will affect the action of object 2.
- Object 11 or 13, since they may break the law by not obeying the red traffic light.

2.2.2 Vehicle Behaviour Prediction

In the vehicle behaviour prediction domain, the literature often uses the terms prediction behaviour of drivers/vehicles or prediction behaviour of target/surrounding

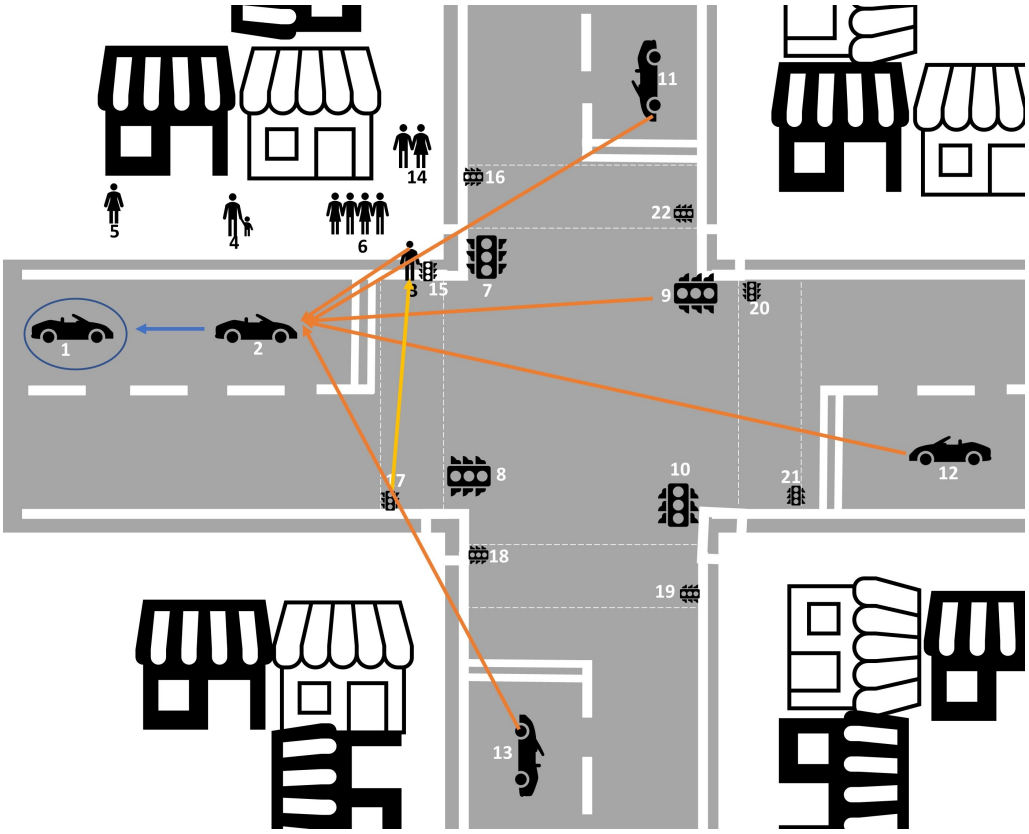


Figure 2.4: General interactions among traffic agents and their environments. Object 1 is the target object (blue circled), the blue arrow shows the direct interaction between the target object and object 2; the orange arrows show the interaction between object 2 and objects 3, 9, 11, 12, and 13; the yellow arrow shows the interaction between object 17 and object 3.

vehicles. The former usually means to predict the behaviour of the EV using its internal data, such as the steering angle, brake pedal position, velocity, speed, indicators status, etc [154, 155, 156, 157]. This approach is suitable for IV systems when considering vehicle-to-vehicle communication. The latter approach involves the EV using on-board sensors to gather information from the surrounding vehicles to predict their behaviour. Only the latter approach is reviewed in this review, as vehicle-to-vehicle communication is not yet available, and AVs would still share roads with conventional human drivers.

Vehicle behaviour prediction is a crucial component of the IV behaviour prediction system as it would enable the IV to perform risk assessment, plan future movements, and make appropriate decisions to avoid/mitigate the impact of collisions. Ideally, a vehicle behaviour prediction algorithm should be fast, cost-effective, accurate, generalise well in different traffic scenes, consider the interdependence between agents, and have a long prediction horizon. A long prediction horizon allows more time for the IV to make decisions and take appropriate actions. A typical vehicle behaviour prediction pipeline consists of multiple steps, including detecting the target and surrounding vehicles. This detection information is used to obtain tracking information. Subsequently, this tracking information is used as an observation feature to predict future trajectories. In order to enhance the quality and duration of predictions, context information of the traffic scene and the intention manoeuvre

Table 2.3: Motion, context and intention cues that can be used to predict vehicle behaviour.

Information	Features
MOTION	**Target Vehicle (TV):** Lateral/longitudinal position, velocity, acceleration, yaw, yaw rate, and relative speed. **TV-to-lane**: lateral offset, and lateral speed. **TV-to-Surrounding Vehicle (SV)**: distance from surrounding vehicles.
CONTEXT	**Road**: Lane marking, number of lanes, lane width, lane curvature, type of lines, entries, exits, left/right/forward arrows, crosswalks, traffic light, traffic signs, type of roads (urban, country, highway-motorway), bumps, road holes, road works, left/right-hand side traffic, and junctions. **Vehicle**: indicators, brake lights, warning lights, type of the vehicle, and sirens' light status. **Other road agents**: pedestrians, animals, cyclists, and trams. **Environment**: sunny, snowing, rainy, foggy, and dark.
INTENTION	Braking, turning left/right, lane keeping, left/right lane change, speeding, normal driving, aggressive driving, abnormal driving, merging, exiting, cutting in/out, and yielding.

of other vehicles can be considered. Table 2.3 provides the type of motion, context, and intention information that has been and could be used by the researchers to predict vehicle behaviour.

Vehicles possess specific characteristics that can simplify behaviour prediction, such as limited movement due to inertia, the requirement to obey traffic rules, and the necessity to navigate within road boundaries. However, predicting vehicle behaviour remains challenging because it depends on various factors, including the actions of other vehicles, traffic regulations, road geometry, and varying driving environments [103, 64]. Additionally, vehicles exhibit multimodal behaviour, with different types of vehicles potentially providing distinct motion information. Prediction can also be complicated if surrounding vehicles are occluded.

Top-view and on-board sensors are the primary data sources to predict vehicles' behaviour. Top-view data are captured from static sensors usually installed on tall buildings, while on-board sensors are captured from sensors installed on the EV. Top-view data have the advantage of providing more precise information since the acquired data have better quality, the vehicles surrounding the TV are captured, and vehicles are not easily occluded. However, it only covers a specific and fixed portion of the traffic scene, limiting the algorithm from generalising to other traffic scenarios. Top-view sensors are typically used in two types of traffic environments: highways-motorways and complex traffic scenes, such as busy urban areas and junctions. Highway-Motorway datasets can suffer imbalanced samples, with more instances of constant velocity behaviour than the specific manoeuvres of interest [142].

On-board sensor data can capture different traffic scenarios, however, its data quality can be affected by noises, surrounding vehicles can be occluded, and to detect all the vehicles surrounding the EV and the TV, more than one sensor might be required (e.g., front, rear, and sides cameras.) [121]. On-board sensor data is particularly advantageous for IV applications because the algorithms that use them, could be directly integrated into AVs, which are already equipped with on-board sensors. Several sensors, such as cameras, radar, and LIDAR could be used to acquire both top-view and on-board data [158, 159, 71, 160]. However, this research mainly focuses on works that have used camera sensors. For more information about the available datasets for vehicle behaviour prediction, please refer to [121]. Table

2.4 summarises the most relevant vehicle trajectory and intention prediction works from 2009 to 2022. From the table, it is observed the following:

- **Shift to Deep Learning and NGSIM Dataset**: Up to 2016, the majority of the works used traditional techniques and their OWN datasets, however, after 2016, most of the works adopted DL techniques and used the NGSIM dataset.
- **Expanding Information Sources**: Vehicle behaviour prediction algorithms have evolved from using only motion information to incorporating additional sources, including manoeuvre, interaction, and driver-style information.
- **Limited Use of Other Datasets**: While the NGSIM dataset gained popularity, other datasets such as Apollo, KITTI, LISA, INTERACTION, HighD, and PREVENTION were rarely used.
- **Trajectory Prediction Dominance**: Most research efforts were to predict trajectories. It was not until 2020 that more research addressed the prediction and recognition of vehicle intentions.
- **Focus on Lane Changing and Turning Maneuvers**: Most research focused on predicting the trajectories and intentions related to lane changing and turning manoeuvres. Other types of manoeuvres, such as reversing, braking, and U-turns, were seldom used.
- **Evaluation Metrics**: The most common evaluation metric for trajectory prediction was the Root Mean Square Error (RMSE), while for intention prediction, accuracy was the predominant evaluation metric.

The following two subsections discuss the algorithms used to predict vehicle behaviour. The first covers the algorithms used to predict trajectories, and the latter, the algorithms used to detect and predict discrete vehicle intention.

Trajectory Prediction

As reported in Table 2.4, vehicle trajectory prediction has been achieved using one or more of the following approaches: physics-based, manoeuvre-based, or interaction-aware motion models [103]. Physics-based motion models were one of the first approaches to be proposed, and they used physics principles to predict vehicle motions. This approach is computationally efficient, meets real-time requirements, and does not require the dataset to be human-labelled. However, they are less suitable for complex scenarios like busy urban scenarios and junctions. This is because they do not take into account the TV intentions, the contextual information of the scene, or the interaction between the TV and the SVs. This lack of information limits the prediction horizon for the EV to less than one-second [103]. In order to overcome the limitation of a short prediction horizon associated with the physics-based approach, manoeuvre-based approaches were introduced. In the manoeuvre-based approach, the EV uses the TV's predicted intention to predict future trajectories. This increases the trajectory prediction horizon and accuracy, as the predicted trajectory would match the predicted intention. However, if the predicted manoeuvre is incorrect, the whole predicted trajectory may also be inaccurate. The interaction-aware

approach uses the trajectories and the intentions of both the TV and the SVs to predict the TV trajectory. This approach further extends the prediction horizon and improves the accuracy of the predicted trajectories. On the other hand, it comes with complexities in implementation, demands greater computational power, and raises questions about determining which vehicles should be considered as SVs. Not all SV might be reliably detected by the EV.

Table 2.4: Relevant works for **vehicle** trajectory and intention prediction.

Work	Methods	Algorithm Objectives	Dataset-Results
PF+RBF [161]	Trajectory prototype. Particle Filter (PF) to track and generate motion hypothesis. RBF to classify trajectories. QRLCS to measure similarity between trajectories. **Evaluation**: RMSE.	Predict future trajectories of the ego and surrounding vehicles.	**OWN** See Table 2.5.
[162]	Extended Kalman Filter **Evaluation**: Mean Distance Error.	Estimate Position and Velocity.	**OWN** Graphs.
[163]	Bayesian Networks. Occupancy Grid Map (OGM). **Evaluation**: FP, FN, and Accuracy.	Detection of lane change manoeuvre.	**OWN** Accuracy: 83.8%.
[164]	SVM. Bayesian Filter. **Evaluation**: Recall, Precision, and F1-score.	Predict lane change manoeuvres of the EV.	**OWN** Recall: 1 Precision: 0.8 F1-score: 0.9 APT: 0.97 s
[165]	Target lane model to predict in which lane the TV will go. 3rd Order Linear System to model trajectory. Auto encoder to cluster the available trajectories into three prototype trajectories. Multi-layer Perceptron (MLP) network to predict the target lane and the probability for each one of the prototype trajectories. OH/PH: (1 s, 2 s, 3 s, 4 s,5 s)/5 s. **Evaluation**: Prediction time and absolute error of lateral position.	Predict lane change of surrounding vehicles.	**NGSIM** Absolute error: 0.7 m.
[166]	Features: linear changes, angular changes, and angular changes histogram. Multi-layer LSTM. **Evaluation**: Accuracy.	Classify manoeuvre intention at intersections.	**KITTI** 2 classes: 85%. 3 classes: 75%. 8 classes: 65%. 12 classes: 40%.
[167]	Detection: DMP + Feature Pyramid + HOG Tracking: MDP + TLD. Trajectory: KF. **Evaluation**: Recall.	Predict future trajectories of the surrounding vehicles.	**OWN** Recall: 92%
[143]	LSMT-RNN. OGM. Data-driven approach. PH: 0.5 s, 1 s, and 2 s. Information: Position, the velocity of surrounding vehicles, and velocity and yaw rate of EV. **Evaluation**: Mean Absolute Error(MAE).	Predict the future position of the surrounding vehicle using OGM.	**OWN** MAE:1.51 for 2 s; 0.88 for 1 s; and 0.59 for 0.5 s.
[168]	CNN. **Evaluation**: Accuracy.	Predict lane change manoeuvre.	**OWN** Accuracy: 89.87%

Continued on next page

Table 2.4 - Relevant works for **vehicle** trajectory and intention prediction (continued from the previous page).

Work	Methods	Algorithm Objectives	Dataset-Results
DESIRE [73]	Observation, sample generation, and rank refinement. CVAE + RNN (GRU) to predict multi-modal trajectories considering latent variables. IOC (based on Reinforcement Learning) to rank and refine the predicted trajectories. Spatial Grid-Based Pooling Layer to extract interaction feature. SCF to combine agents' interactions and scene context. OH/PH: 2 s/4 s. **Evaluation**: L2 distance error and miss rate.	Predict the future position of the surrounding vehicles considering static and dynamic scene context and interaction between agents.	**SDD KITTI** See Table 2.5.
[142]	LSTM encoder-decoder. **Evaluation**: average RMSE.	Predict the TV's future position by considering surrounding vehicles.	**NGSIM** See Table 2.5.
[156]	Two LSTM networks, one to encode past trajectories and predict intention manoeuvre, the other to encode past trajectories, and the predicted manoeuvre to decode future trajectories. **Evaluation**: lateral and longitudinal RMSE.	Predict vehicle trajectory using past trajectories and predicted manoeuvre intention.	**NGSIM** See Table 2.5.
[169]	LSTM encoder-decoder. OGM. Beam search algorithm. OH/PH: 3 s/2 s. **Evaluation**: MAE.	Predict the future position of the target and the surrounding vehicles.	**OWN** MAE (Grid): 1.27 for 2 s; 1.14 for 1.6 s; 0.99 for 1.2 s; 0.84 for 0.8 s; and 0.64 for 0.4 s.
M-LSTM [140]	Tracking history and Manoeuvres classification (Lane change, brake, and normal driving) to allow multi-modal prediction. LSTM encoder-decoder to encode tracked history motions and to decode multi-modal future motions. OH/PH: 3 s/5 s. **Evaluation**: RMSE.	Trajectory prediction of surrounding vehicles considering the interaction between traffic agents.	**NGSIM** See Table 2.5.
C-VGMM+VIM [170]	HMM for manoeuvre recognition. IMM + VGMM to predict trajectories. Markov Random Field for vehicle interaction. PH: 5 s **Evaluation**: Manoeuvre classification accuracy, mean and median error for the trajectory prediction.	Manoeuvre Intention (lane change, overtaking, cutting-in, drift into ego lane) and Trajectory Prediction.	**LISA-A** MAE overtakes and cut-ins: 2.49 for 5 s; 1.94 for 4 s; 1.39 for 3 s; 0.82 for 2 s; and 0.29 for 1 s. MAE stop-and-go: 2.17 for 5 s; 1.65 for 4 s; 1.14 for 3 s; 0.64 for 2 s; 0.20 for 1 s. Accuracy for overtakes and cut-ins: 55.89% Accuracy stop-and-go: 87.19% Time: 6FPS.
CS-LSTM [139]	LSTM encoder-decoder to encode previous motion information and to decode future motion. Convolutional Social Pooling to learn agent's interdependence motions. Multi-modal prediction (6 classes: RLC, LLC, NLC, brake, and normal). OH/PH: 3 s / 5 s. **Evaluation**: RMSE and Negative log-likelihood (NLL).	Predict future motions of surrounding vehicles taking into consideration motion, spatial configuration, and interdependence between agents.	**NGSIM** See Table 2.5. Computation time: 0.29 s (reported by [144]).
SA-LSTM [171]	Surrounding-Aware LSTM. OH: 6, 9, and 12 frames. **Evaluation**: Accuracy.	Predict lane change manoeuvre and future trajectories.	**NGSIM** Avg. Accuracy: 86.19%.

Continued on next page

Table 2.4 - Relevant works for **vehicle** trajectory and intention prediction (continued from the previous page).

Work	Methods	Algorithm Objectives	Dataset-Results
MATF [172]	Hybrid Model (LSTM + CNN) LSTM to encode past trajectories for multiple agents. CNN to encode context information. MATF to fuse interaction, spatial structure, and context information. Conditional generative adversarial training to detect uncertainty in predicting manoeuvres. Environment: Highway-Motorway and pedestrian crowd scenes. OH/PH: 3 s / 5 s. **Evaluation**: RMSE.	Trajectory prediction by considering social interaction and scene context.	**NGSIM** See Table 2.5.
[173]	Features: local position, velocity, acceleration, distance to lane markings, yaw angle and rate, lateral velocity, and acceleration. ANN and SVM. **Evaluation**: Recall, Accuracy, Precision, and F1-score.	Predict lane change manoeuvres of the surrounding vehicles.	**NGSIM** **ANN** Accuracy: 98.8%. Prediction: 2.4 s. **SVM** Accuracy: 97.1%. Prediction: 1.9 s.
ST-LSTM [136]	Spatio-temporal LSTM. Short-cut connections to avoid gradient vanishing. Weighted sum to integrate the outputs. Consider the 6 vehicles around the TV. OH/PH: 3 s/6 s. **Evaluation**: RMSE.	Trajectory prediction by considering spatial and temporal information.	**NGSIM I-80** See Table 2.5.
GRIP [144]	Fixed Graph Convolutional (10 blocks) Model to represent interactions between agents. Single LSTM encoder-decoder to make trajectory predictions. OH/PH: 3 s/5 s. Hardware: 4.0GHz i7, 32GB memory, and NVIDIA Titan XP. **Evaluation**: RMSE.	Predict surrounding vehicle trajectories considering the interaction between them.	**NGSIM** See Table 2.5. Computation time: 0.05 s.
GRIP++ [138]	Dynamic Graph Convolutional (3 blocks) Model to represent interactions between agents. Three GRU-RNN encoder-decoder to make trajectory predictions. OH/PH: 3 s/5 s. Hardware: 4.0GHz i7, 32GB memory, and NVIDIA Titan XP. **Evaluation**: RMSE, WSADE, and WSFDE.	Predict surrounding vehicle trajectories considering the interaction between them.	**ApolloScape** WSADE: 1.2588. WSFDE: 2.3631. **NGSIM** See Table 2.5. Computation time: 0.02 s.
NLS-LSTM [141]	Local and non-local social pooling. LSTM encoder-decoder. **Evaluation**: RMSE.	Predict vehicle trajectory using local and non-local social pooling.	**HighD** See Table 2.5 **NGSIM** See Table 2.5
[174]	Hybrid Model ANN to classify manoeuvres. LSTM to predict trajectories. OH: 3 s, 5 s, and 6 s. PH: 1 s, 3 s, and 5 s. **Evaluation**: RMSE and classification accuracy.	Manoeuvre classification and trajectory prediction.	**NGSIM** See Table 2.5
[120]	Two stream CNN (Disjoint). Spatio-temporal Multiplier Networks (ST) (cross-stream connections). ResNet-50 to extract both temporal and contextual information. OH/PH: 2 s/(1-2 s). 4 Sizes of RoI are used x1, x2, x3 and x4. Dense optical flow to extract movement context. **Evaluation**: Classification accuracy and Prediction Accuracy.	Recognition and prediction of lane change/keep manoeuvre using stacked visual cues from videos.	**PREVENTION** *Disjoint* Classification Accuracy:89.46%. Prediction Accuracy:91.02%. *ST* Classification Accuracy: 90.30%. Prediction Accuracy: 91.94%.

Continued on next page

Table 2.4 - Relevant works for **vehicle** trajectory and intention prediction (continued from the previous page).

Work	Methods	Algorithm Objectives	Dataset-Results
ARIMA-Bi-LSTM [175]	Off-line Bi-LSTM. Online ARIMA + Bi-LSTM. PH: 5 s. **Evaluation**: RMSE and Accuracy.	Predict trajectories and turning manoeuvres at intersections.	**NGSIM-LP** GS: lateral 0.032; long. 0.1093. TL: lateral 0.2719; long. 0.1592. TR: lateral 0.1168 long. 0.3954 Accuracy: 94.2% at 1 s, 93.5% at 2 s, and 74.5% at 3 s.
[121]	TSM to differ between target and surrounding vehicles. TIM to extract motion pattern. Greyscale image to extract context information. Compared various CNN models to detect and predict manoeuvres. OH: 1 s. **Evaluation**: Accuracy, precision, recall, anticipation (s), and AUC.	Detection and prediction of lane change performed by surrounding vehicles. Present a baseline to compare human performance against automated systems. Briefly compared the available datasets.	**PREVENTION** **Manoeuvre Detection:** Accuracy: 82.7%. Anticipation:2.28 s. **Manoeuvre Prediction:** Accuracy: 83.4%. Prediction: 0.72 s.
[69]	4 action recognition models were evaluated: Two-stream CNN, Two-stream Inflated 3D CNN, STM network, and SlowFast Network. 4 Sizes of RoI. Dense optical flow to extract movement context. OH:PH: 2 s/(1-2 s). **Evaluation**: Accuracy (%).	Recognition and prediction of lane change/keep event using stacked visual cues from videos.	**PREVENTION** Accuracy for STM: 91.91% for 2 s; 86.51% for 1 s.
ST-Conv-LSTM [176]	Spatial-temporal Convolutional LSTM. OH/PH: 2.4 s/1 s. **Evaluation**: Accuracy.	Predict lateral (lane change) and longitudinal (holding, sharp acceleration, deceleration, and stopping) intention.	**BDD100K** Accuracy: 57.9%.
IPTM-LSTM [177]	Intention encoder-decoder LSTM. Trajectory encoder-decoder LSTM. IPTM. **Evaluation**: Accuracy and RMSE.		**NGSIM-LP** Avg. Intention Accuracy: 90.94% RMSE: See Table 2.5 **INTERACTION** Avg. Intention Accuracy: 86.92%.
LSTM-GAN [178]	LSTM + Generative Confrontation Network. **Evaluation**: Accuracy.	Predict vehicle turning intention.	**OWN** Accuracy: 90.9%.
[179]	Game theory model to predict the intention of the driver. Recognise the vehicle behaviour using past vehicle state. Nash-optimisation function. **Evaluation**: Lateral position error, yaw rate error, probability error.	Predict the trajectory of lane change maneuvers using driver style (aggressive or conservative) and behaviour recognition.	**NGSIM** Graphs.
AI-TP [180]	Approach: Data-driven. **Features**: Past trajectories. **Model(s)**: graph attention mechanism (AI-TP), ConvGRU, **Evaluation**: MSE.	Trajectory prediction.	**NGSIM** See Table 2.5

The previously cited approaches have been implemented using either traditional or DL techniques. Traditional techniques encompass Linear methods like KF and Switching Linear Dynamic Models, as well as Non-linear methods such as EKF, UKF, Switching Non-Linear Dynamic Models, Particle filters, Bayesian filtering, Monte Carlo simulation, Naive Bayes Classifiers, Dynamic Bayesian Networks, HMM, SVM, case-based reasoning, random decision Forest, Artificial Neural Network (ANN), SVM, and Gaussian Process NN [69]. Traditional techniques have the advantage of being fast to infer and not requiring an extensive dataset. However, they struggle to generalise well and have limited prediction horizons. Additionally, most traditional techniques do not inherently account for vehicle interactions and may require

additional features. The DL techniques used in the literature were based on ANNs, Convolutional Neural Networks (CNN), Fully Connect Networks (FCN), Recurrent Neural Networks (RNN), Graph Convolutional Neural Networks (GCNN), Gated Recurrent Unit (GRU), or Long Short-Term Memory (LSTM) [69]. The main advantage of DL techniques is their ability to extract the required features to predict vehicle behaviour implicitly. Some DL techniques even consider the interaction between vehicles by themselves, for instance, RNN and GCNNs. Yet, DL techniques may not address the multi-modal behaviour of vehicles as they tend to average the multiple possible modalities to minimise the regression error. They also require an extensive dataset to generalise well, take longer to train, may suffer from gradient vanishing, and may not provide accurate trajectory prediction for longer time horizons.

The following paragraphs will discuss the most relevant DL algorithms to predict vehicle trajectories.

Altché and La Fortelle [142] and Kim et al. [143], to the authors' knowledge, were one of the first ones to use LSTM-RNN to predict the future trajectories of the surrounding vehicles by using their past trajectories as input feature. Park et al. [169] predicted future trajectories using an encoder-decoder LSTM. The encoder encodes past trajectories of the surrounding vehicles, while the decoder decodes future trajectories in an Occupancy Grid Map (OGM). The authors also applied a beam search algorithm, to reduce the error propagation caused by the greedy strategy that the decoder LSTM uses to maximise the output probabilities.

Deo and Trivedi [140] presented a Manoeuvre-LSTM model that encodes motion and interaction of the surrounding vehicles to assign probabilities for each manoeuvre. The assigned probabilities enable multi-modal trajectory predictions. During that period, the algorithm achieved better RMSE results than the state-of-the-art algorithms, but the RMSE values for long PH were still high. Although the algorithm considered vehicle interaction, it did not consider their inter-dependencies. In order to overcome this limitation, [139] combined convolutional social pooling and encoder-decoder LSTM to predict manoeuvres and future trajectories. The convolution social pooling can learn the interaction and interdependence of the surrounding vehicles. The downside of the algorithm is that the social tensor of the convolutional social network was fixed to the defined spatial grid around the TV, and it did not consider visual context information. The disadvantage of the last two algorithms is that the predicted trajectories depend on the manoeuvre classification performance. For example, [139] compared their algorithm with and without considering manoeuvre intention, and they reported that the algorithm without manoeuvre had better performance.

Dai, Li, and Li [136] claimed that the existing LSTM models suffered from vanishing gradients and could not learn spatial interactions between traffic agents. Therefore, they modified the conventional LSTM model by adding shortcut connections and treated spatial interaction between traffic agents as time series. Their model performed better than the M-LSTM [140] model, which considered manoeuvre prediction information.

Li, Ying, and Chuah [144] presented the Graph-based Interaction-aware Trajectory Prediction (GRIP) algorithm to predict future trajectories of the TV considering the SV information. GRIP used a GCNN to learn interaction patterns between the TV and SVs. The learnt patterns were then fed to an encoder-decoder

LSTM model to predict future trajectories. GRIP became the state-of-the-art algorithm and was one of the few works to report inference times. The disadvantage of GRIP is that it uses a fixed graph structure to learn the interaction between agents, which may not be suitable for complex urban scenarios. In response, [138] proposed GRIP++, an enhanced version that used fixed and dynamic graphs to learn the interaction between agents. GRIP++ offered improved computational efficiency compared to the existing algorithms.

Benterki et al. [174] proposed a hybrid method combining ANN and LSTM. ANN was first used to classify the TV's manoeuvre (LLC, RLC, and NLC) using the following manually selected features: yaw, yaw rate, lateral velocity, and lateral acceleration. Subsequently, the LSTM used the vehicle's position and the predicted manoeuvre to predict future trajectories. While the authors tested their algorithm in a real vehicle scenario, only three tests were performed: two for right lane changes and one for left lane-change manoeuvre.

Luan et al. [179] used vehicle behaviour and driver style to predict future trajectories. History trajectories of the surrounding vehicles were used to determine the type of driver, whether aggressive or conservative. Then, the predicted type of driver was used by a game theory model to predict the driver's intention. Vehicle behaviour was recognised by using past vehicle state. A comprehensive trajectory was then predicted by feeding the predicted driver intention and the recognised vehicle behaviour into two Nash-optimisation functions. The authors claimed that with the inclusion of the type of driver information, the prediction of the vehicle trajectory was improved. However, their results could not be directly compared to state-of-the-art algorithms such as [144, 138, 139].

The previously cited works did not consider the scene's visual context, which is an important feature as it considers the constraints of the environment. Authors [73] presented a Deep Stochastic Inverse Optimal Control RNN encoder-decoder (DESIRE) network that considers scene context. The DESIRE uses an RNN encoder to encode past trajectories, a Conditional Variational Auto-Encoder (CVAE) to enable multi-modal predictions, an RNN decoder to decode future trajectories, and a CNN to extract scene context information. In order to refine the predicted results, DESIRE applies Inverse Optimal Control (IOC) to the predicted trajectories and the extracted context information. The authors concluded that the model achieving the best results was the one that considered both scene context and vehicle interactions. Although their algorithm performs better than linear methods, it can not be directly compared to other works in the literature, since they used different metrics and datasets. Zhao et al. [172] aimed to predict future trajectories using interaction information between agents and the scene context. An LSTM network was used to encode multi-agent past trajectories, and a CNN was used to extract feature vectors from the scene context. The outputs of the LSTM and CNN were fused using a multi-agent tensor fusion (MATF) network, and the output of the MATF was then fed into an FCN to predict future trajectories. While these last two cited works considered visual context and achieved good performance, they did not outperform algorithms that did not consider visual contexts, such as GRIP and ST-LSTM.

Table 2.5 and Figure 2.5 report the results for most algorithms reviewed in this paper. Note that the graph only contains the works that have used the same dataset, OH, PH, and evaluation metrics. The following observations can be made from the table and the graph:

Table 2.5: Results for the most relevant **vehicle** trajectory prediction works.

Work	**Dataset**	**Metrics**	**Axis**	**Obs. Hor.**	**1s**	**2s**	**3s**	**4s**	**5s**	**6s**
CV [139]	NGSIM	RMSE	Both	3 s	0.73	1.78	3.13	4.78	6.68	-
S-LSTM [181]	NGSIM	RMSE	Both	3 s	0.65	1.31	2.16	3.25	4.55	-
GAIL-GRU [182]	NGSIM	RMSE	Both	3 s	0.69	1.51	2.55	3.65	4.71	-
C-VGMM+VIM [170]	NGSIM	RMSE	Both	3 s	0.66	1.56	2.75	4.24	5.99	-
M-LSTM [140]	NGSIM	RMSE	Both	3 s	0.58	1.26	2.12	3.24	4.66	-
CS-LSTM(M) [139]	NGSIM	RMSE	Both	3 s	0.62	1.29	2.13	3.20	4.52	-
CS-LSTM [139]	NGSIM	RMSE	Both	3 s	0.61	1.27	2.09	3.10	4.37	-
MATF GAN [172]	NGSIM	RMSE	Both	3 s	0.66	1.34	2.08	2.97	4.13	-
ST-LSTM-1350 [136] avg.	NGSIM	RMSE	Both	3 s	0.56	1.19	1.93	2.78	3.76	4.84
GRIP [144]	NGSIM	RMSE	Both	3 s	**0.37**	**0.86**	**1.45**	2.21	3.16	-
GRIP++ [138]	NGSIM	RMSE	Both	3 s	0.38	0.89	1.45	2.14	2.94	-
AI-TP [180]	NGSIM	RMSE	Both	3 s	0.47	0.1.05	1.53	**1.93**	**2.31**	-
NLS-LSTM	NGSIM	RMSE	Both	3 s	0.56	1.22	2.02	3.03	4.30	-
[141]	HighD				0.20	0.57	1.14	1.90	2.91	-
OGM-LSTM	NGSIM	RMSE	Lateral		0.56	1.24	-	-	-	-
[143]			Longi.		3.05	6.70	-	-	-	-
Dual LSTM	NGSIM	RMSE	Lateral	5 s	0.15	0.26	0.38	0.45	0.49	-
[156]			Longi.		0.47	1.39	2.57	4.04	5.77	-
[142]	NGSIM	RMSE	Lateral		0.11	0.25	0.33	0.40	0.47	-
			Longi.		0.71	1.98	3.75	5.96	9.00	-
ANN-LSTM	NGSIM	RMSE	Lateral	3 s	0.043	-	0.125	-	0.235	-
[174]			Longi.		0.122	-	0.235	-	0.264	-
IPTM-LSTM [177]	NGSIM-LP	RMSE	Both	3 s	0.77	1.34	2.19	-	-	-
MATF GAN [172]	Massachusetts	RMSE	Both	3 s	0.75	1.4	2.0	2.7	-	-
PF+RBF [161]	OWN	RMSE	Both	-	0.7	1.4	5.0	-	-	-
CS-LSTM(M) [139]	NGSIM	NLL	Both	3 s	0.58	2.14	3.03	3.68	4.22	-
C-VGMM+VIM [170]	LISA-A	MAE	Both	3 s	0.24	0.69	1.18	1.66	2.18	-
DESIRE	KITTI	DE	Both	2 s	0.28	0.67	1.22	2.06	-	-
[73]	SDD	PE			1.29	2.35	3.47	5.33		

- Not all algorithms can be directly compared since they have used different datasets, metrics, OH, and/or PH. Additionally, some of the works combined the predicted lateral and longitudinal trajectories to calculate their metrics, while others calculated the metrics for lateral and longitudinal trajectories, separately.

- When comparing the algorithms that used the same dataset, metrics, OH, and PH, it is observed that GRIP has the best performance for PTHs of 1 s, 2 s, and 3 s; while AI-TP has the best performance for PTHs of 4 s and 5 s.

- With the exception of KITTI and LISA-A, all the other datasets are top-view cameras, and the most frequently used dataset is the NGSIM.

- The most used metric is RMSE.

- Most of the works adopted an OH of 3 s and PH of up to 5 s.
- It is noticed that the algorithms' performance worsens as the prediction horizon increases.

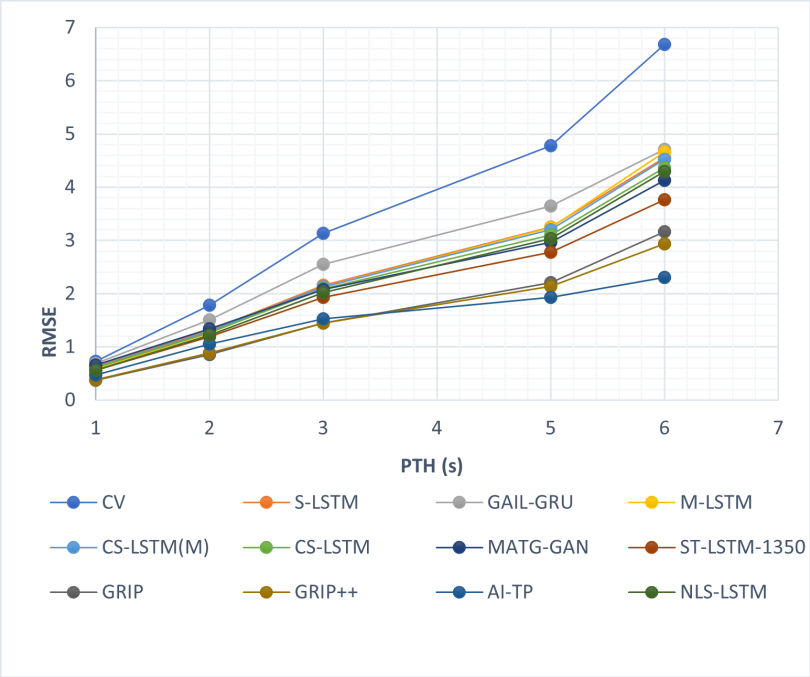


Figure 2.5: Vehicle Trajectory Prediction Performance using the NGSIM dataset, with an OH of 3 s, and PH ranging from 1-5 s (See Table 2.5).

Intention Recognition and Prediction

The difference between intention recognition and prediction is that for intention recognition, the manoeuvre can be recognised without any anticipation, while for intention prediction, the manoeuvre event must be recognised before it happens. Generally, the researcher specifies the desired anticipation time and then the accuracy of the manoeuvre detection is calculated. The intention of a vehicle's manoeuvre can be recognised by using either prototype trajectories or the manoeuvre intention estimation method.

The literature assumes that there is a motion pattern for the different types of vehicle manoeuvres. Consequently, previous trajectory samples can be used to define a set of prototype trajectories, which are then used to represent the different motion patterns. Vehicle manoeuvres are then predicted by using initially observed trajectories performed by the vehicle and matching them to the best available motion patterns. However, this approach is computationally expensive because it requires a substantial number of sample trajectories to determine the numerous possible motion patterns.

In contrast, the manoeuvre intention estimation methods use vehicle motion and road context features to classify the different types of manoeuvres, for instance, stopping/non-stopping, turning left/right, etc. Although this method is less complex

than calculating the numerous trajectory probabilities, a large training dataset is required to make the system robust to the different road scenarios. Another limitation is that the manoeuvre classes may not be sufficient to cover the complexity of the real vehicle intention. For instance, the system may predict a braking manoeuvre, but the braking can be normal or harsh. A proposed solution is to sub-categorise the manoeuvre, for example, normal/harsh stopping and normal/sharp right/left turn; however, this adds complexity to the dataset labelling [64].

Intention prediction algorithms can also use predicted trajectories and vehicle interaction to achieve better accuracy. Traditional methods used to predict vehicle intentions are Heuristics, Bayesian Networks, HMM, and SVM. DL methods commonly used are RNN, LSTM, and action recognition models.

The following paragraphs will discuss the most relevant DL algorithms used to predict vehicle intention manoeuvre.

Khosroshahi, Ohn-Bar, and Trivedi [166] implemented a multi-layer LSTM network to classify manoeuvre intentions at complex intersections. They extracted samples representing manoeuvre intentions from the KITTI dataset to train and test the algorithm. The input features included linear and angular changes, as well as a histogram of angular changes in the vehicle trajectories. The authors performed experiments with different numbers of manoeuvre classes: 2 (straight or turning), 3 (straight, turning left/right), 8 and 12 classes. The algorithm performed well with 2 and 3 classes, but the accuracy significantly decreased with 8 and 12 classes.

Lee et al. [168] transformed real-world images into a simplified version of Bird's Eye View (BEV) and fed them into a CNN to predict lane change behaviour. Zhang and Fu [175] used an offline Bidirectional LSTM to learn driving behaviour and an online Auto-Regressive Integrated Moving Average (ARIMA) to learn past trajectories and predict future ones. The outputs of the offline Bi-LSTM and ARIMA were then fed into another Bi-LSTM to recognise turning behaviour as left-turn, right-turn, or going straight. The algorithm went through evaluation using the NGSIM Lankershim and Peachtree Street dataset, and was able to meet real-time requirements while achieving good accuracy recognition for the PH of 1 s and 2 s. However, accuracy dropped when considering PH of 3 s, and it only considered turning left/right and going straight manoeuvres. Whereas vehicles at intersections can perform more complex manoeuvres as reported by [166]. In addition, the dataset used was acquired from top-view sensors while AVs are equipped with on-board camera sensors. Benterki et al. [173] compared two conventional methods to predict lane-change manoeuvre, ANN and SVM. They concluded that ANN and SVM have almost the same performance; however, ANN showed the best results.

Izquierdo et al. [121] used CNN, action recognition, and prediction methods to recognise and predict lane-keeping/changing manoeuvres. Instead of using a sequence of images, they encoded context, interaction, and dynamic state information in a unique enriched image. The enriched image was created by extracting the red channel from a grey-scale version of the original image, using a target selection method (TSM), and a temporal integration method (TIM). The authors also investigated human performance in recognising and predicting lane changes. Their findings indicated that humans can detect 83.9% of the lane change events with an average anticipation of 1.66 s before the manoeuvre is completed. Only 3 out of 72 users were able to predict the lane change events before they started, with an average prediction horizon of 1.08 s. On the other hand, their best algorithm, which

considers the trade-off between accuracy and anticipation, achieved 86.4% accuracy with average anticipation of 2.09 s when considering TTE equal to 0. When TTE was set to 1 s, their algorithm achieved an anticipation of 2.69 s, a prediction of 0.72 s, and an average accuracy of 83.4%.

Fernández-Llorca et al. [120] and Biparva et al. [69] recognised and predicted lane-keeping/changing manoeuvres using video action recognition approaches. Biparva et al. [69] used four types of video action recognition approaches: Two-stream CNN, Two-stream Inflated 3D CNN, spatio-temporal Multiplier Networks, and SlowFast Networks. All of the aforementioned networks used spatial and temporal information from a single image, a sequence of images, or a sequence of optical flow images for recognition and prediction tasks. Moreover, four sizes of RoI were used, denoted as x1, x2, x3 and x4, to consider the interaction between agents, and to extract contextual information around the TV. The network with the best recognition performance was the SlowFast CNN achieving an accuracy of 90.98% with an OH of 2 s before the TTE. Meanwhile, the network with the best prediction performance was the spatiotemporal multiplier, achieving an accuracy of 91.94% with an OH of 2 s. The limitations of the previously cited works are as follows: the distribution of the manoeuvre classes was imbalanced, with more lane-keeping samples than lane-changing ones; the time required to recognise and predict a single instance was not provided; and some of the algorithms, such as the SlowFast network, was not able to complete its training due to the GPU memory limitation.

Furthermore, it was observed from the previous vehicle intention prediction works that the authors have selected a fixed PH to predict the vehicle's intentions. The drawback of using a fixed PH is that manoeuvre samples may vary in length. For instance, the lane-change manoeuvre performed by an aggressive driver will be shorter than a lane-changing manoeuvre performed by a normal driver.

2.2.3 Pedestrian Behaviour Prediction

At present, IV systems can effectively detect and track pedestrians, however, this alone is not enough to prevent potential collisions. In order to avoid a collision, IV systems must predict pedestrian behaviours. This section aims to provide a literature review of the challenges and techniques used over the years for addressing pedestrian behaviour prediction.

Pedestrian behaviour prediction has been applied in three main types of datasets: datasets that are recorded using drones, for example, ETH and UCY; datasets recorded from static cameras; and datasets recorded from car dash cameras, for example, Daimler, JAAD, and PIE or KITTI. Datasets from on-board cameras are more appropriate for training models for IV because they provide a more realistic representation. However, when the car is in motion, it may affect the position of the pedestrian bounding box, and pedestrians can be easily occluded. On-board datasets can be categorised as either naturalistic or non-naturalistic, as discussed by [183]. In non-naturalistic datasets, the pedestrian behaviours and intentions are performed by actors, whereas, in naturalistic datasets, behaviours and intentions are recorded from actual road traffic scenarios.

Pedestrian behaviour prediction has been heavily investigated in the past years, and it has many challenges. For instance, pedestrians are highly dynamic, they can move in many directions and change them very quickly, and can be easily occluded

Table 2.6: Features and information used to predict pedestrian intention.

Feature	Information
Bbox coordinates	Position, speed, height and width.
Bbox cropped image	Pedestrian appearance, local and surrounding context.
Full image	Global context and some interaction between different traffic objects.
Body Pose	Displacement, action, skeleton, and landmarks.
EV position/speed	Interaction between pedestrian and EV. Pedestrian behaviour is affected by EV speed.

by other objects. They can also become distracted by their own objects or external environments, their movements may be influenced by other traffic agents, and they can be difficult to detect in poor visibility conditions. As reported in Tables 2.6, 2.7 and 2.9, researchers have proposed various methods and features to address these challenges over the years. From these tables, the following observations can be made:

- Until 2018, most of the works used traditional methods and their OWN dataset. Thereafter, most authors adopted DL techniques and used the ETH and UCY datasets for trajectory prediction, as well as JAAD and PIE datasets for intention prediction.
- Pedestrian behaviour prediction algorithms have evolved, from solely using motion information to using pedestrian appearance, body pose landmarks, local/global context, interactions between agents, and EV dynamics.
- Prior to 2018, the focus was predominantly on trajectory prediction, thereafter substantial research efforts have been dedicated to predicting pedestrian intentions.
- Most of the intention prediction works were to predict the crossing intention.
- The most used evaluation metrics for intention prediction were accuracy, F1-score, precision, recall, Area Under the Curve (AUC), and Receiver Operating Characteristic Curve (ROC-AUC).
- The most used evaluation metrics for trajectory prediction were Average Displacement Error (ADE), Average Final Displacement Error (FDE), and MSE. Other metrics are Average Non-linear Displacement Error (ANDE), Mean Average Displacement (MAD), and Final Average Displacement (FAD).

The following subsections discuss some of the algorithms reported in Tables 2.7 and 2.9. The first subsection provides an in-depth exploration of trajectory prediction algorithms, while the subsequent subsection explores intention prediction algorithms.

Table 2.7: Relevant works for **pedestrian** trajectory prediction.

Work	Methods	Dataset/Results
[184]	**Approach**: Dynamic. **Features**: constant velocity, acceleration, turn. **Models**: Recursive Bayesian filters – Compared EKF and IMM filters. **PH**: < 2 s. **Evaluation**: MLPE.	**Daimler** IMM has not shown significant performance over simpler models.
[185]	**Approach**: Dynamic. **Features**: optical flow. Compared the performance between GDPMs, PHTM, KF and IMMKF. Provided human performance on classifying pedestrian behaviour prediction. **Evaluation**: Mean Combined Longitudinal and Lateral RMSE.	**OWN** (on-board) GDPM and PHTM showed better accuracy, however, they are more computationally expensive. 10-50 cm Time Horizon 0.77 s.
[68]	**Approach**: Dynamic + Context. **Features**: Head orientation, distance between vehicle and pedestrian, distance between pedestrian and curb. **Models**: Dynamic Bayesian Filters (SLDS). **Evaluation**: Predictive log likelihood.	**OWN** (on-board) Outperforms state-of-art algorithm PHTM. Best result of -0.33 was achieved in the critical, vehicle-seen and stopping scenario using the full context information.
Social-LSTM [181]	**Approach**: Data driven. **Features**: Past trajectories. **Models**: Social pooling layer, and LSTM. **OH/PH**: 8 (3.2 s)/12 (4.8 s) frames. **Evaluation**: ADE, FDE, and AND.	**ETH and UCY** ADE/FDE/AND: 0.27/0.61/0.15.
[186]	**Approach**: Dynamic + Context. **Features**: pedestrian state (position, orientation, and speed), predicted goals, environment context (building, sidewalk, crosswalk, road and grass), dynamic environments such as traffic lights, and assumed rational behaviour for the agent. **Models**: Jump-Markov Process, and Rao-Blackwellized filter. **Evaluation**: L2 error, and Average prediction error.	**OWN** (on-board) for training and **KITTI** for evaluation. Displayed in a graph.
[187]	**Approach**: Data driven + Goal-directed. **Features**: visual cues, predicted pedestrian destinations, and trajectories. **Models**: RMDN, LSTM, topology network, and Markov Decision Process. **Evaluation**: Predicted probability distribution, Average accuracy of predicted destination, and prediction accuracy over time.	**OWN** (on-board) Outperformed IMM. Results were not clear, but from graph Prediction accuracy $10^{(-1)}$ for 1.5 s. Destination plays an important role when trying to predict pedestrian intention.
SR-LSTM [188]	**Approach**: Data driven and social behaviour. **Features**: trajectories and current state of the neighbours. **Model(s)**: SR-LSTM and attention mechanism. **Evaluation**: MAD, and FAD.	**ETH** and **UCY** MAD: 0.45; FAD: 0.94.
Social-GAN [153]	**Approach**: Data driven. **Features**: Past trajectories. **Model(s)**: GAN, Pooling Module, and LSTM. **PH**: 8 and 12 meters. **Evaluation**: ADE and FDE.	**ETH, UCY** ADE: 0.39/0.58. FDE: 0.78/1.18.
Social attention [150]	**Approach**: Data driven. **Features**: Past trajectories. **Model(s)**: ST-Graph, LSTM, and Attention. **OH/PH**: 8 (3.2 s)/12 (4.8 s) time steps. **Evaluation**: ADE and FDE.	**ETH and UCY** ADE: 0.30 m. FDE: 2.59 m.
SS-LSTM [135]	**Approach**: Data driven. **Features**: Past trajectories, neighbour features (occupancy maps: grip, circle and log), and individual information. **Model(s)**: CNN, and Hierarchical-LSTM. **OH/PH**: 8/12 frames. **Evaluation**: ADE and FDE.	**ETH and UCY** ADE: 0.070 pixels. FDE: 0.133 pixels.

Continued on next page

Table 2.7 – continued from the previous page.

Work	Methods	Dataset/Results
CIDNN [151]	**Approach**: Data driven. **Features**: Past trajectories, and interactions. **Model(s)**: stacked-LSTM, and MLP. **OH/PH**: 5/5 frames. **Hardware**: Intel Xeon CPU E52643 4.40 and TITAN GPU. **Evaluation**: ADE.	**GC/ETH/UCY/CUHK/Subway** ADE: 0.012/0.09/0.12/0.008/0.016. Inference: 0.43 ms
LSTM-Baysian [189]	**Approach**: Data driven. **Features**: Bbox coordinates past trajectories and EV odometry. **Model(s)**: Two stream architecture, Bayesian RNN (LSTM), and CNN. **OH/PH**: 0.5/1 s. **Evaluation**: MSE in pixels and NLL.	**CityScapes(on-board)** MSE/NLL: 505/3.92.
DBN-SLDS [190]	**Approach**: Data driven. **Features**: context cues (VRU actions, and its static and dynamic environment). **Model(s)**: DBN and SLDS. **TTE** = [-15, 0] **PH**:1 s. **Evaluation**: Prediction error.	**OWN (on-board, non-naturalistic** Graphs.
MX-LSTM [191]	**Approach**: Data driven. **Features**: Past trajectories, and head pose estimation. **Model(s)**: tracklets, vislets, VFO social pooling, and LSTM. **OH/PH**: 8/12 frames. **Evaluation**: MAD and FAD in meters.	**UCY** MAD/FAD: 0.49/1.12 m. **Towncentre** MAD/FAD: 1.15/2.30 m.
Scene-LSTM [192]	**Approach**: Data driven. **Features**: Past trajectories and scenes are divided into grid cells. **Model(s)**: Scene Data Filter, and Coupled-LSTM. **OH/PH**: 3.2/4.8 s. **Evaluation**: ADE, FDE and NDE.	**UCY and ETH** ADE/FDE/NDE: 0.7/0.7/0.9.
SoPhie [152]	**Approach**: Data driven. **Features**: Past trajectories, social interactions, and images of the scene. **Model(s)**: CNN, LSTM, GAN, Social and physical attention mechanism. **PH**: 12 future timesteps. **Evaluation**: ADE and FDE.	**ETH, UCY** ADE: 0.54 m. FDE: 1.15 m. **SDD** ADE: 16.24 pixels. FDE: 29.38 pixels.
StarNet-DNN [149]	**Approach**: Data driven. **Features**: Past trajectories. **Model(s)**: StarNet DNN (Host and hub networks), and LSTM. **PH**: 8 frames. **Hardware**: Tesla V100 GPU. **Evaluation**: ADE and FDE.	**ETH and UCY** ADE/FDE: 0.30/0.57. Inference: 0.073 s.
PECNet [146]	**Approach**: Data-driven and goal-directed. **Features**: Past trajectories and estimated end point destination. **Model(s)**: CVAE, attention mechanism, and social pooling. **OH/PH**: 3.2/4.8 s. **Evaluation**: ADE and FDE.	**ETH and UCY** ADE/FDE: 0.29/0.48 m. **SDD** ADE/FDE: 9.96/15.88 p.
ST-GCNN [147]	**Approach**: Data driven. **Features**: Past trajectories and sequence of images. **Model(s)**: GCN, and TXP-CNN. **OH/PH**: 3.2/4.8 s. **Evaluation**: ADE and FDE.	**ETH and UCY** ADE/FDE: 0.44/0.75 m.
RSBG [148]	**Approach**: Data driven. **Features**: Past trajectories and local context. **Model(s)**: GCN, CNN, and LSTM. **OH/PH**: 3.2/4.8 s. **Evaluation**: ADE and FDE.	**ETH and UCY** ADE/FDE: 0.48/0.99 m.

Continued on next page

Table 2.7 – continued from the previous page.

Work	Methods	Dataset/Results
LVTA [193]	**Approach**: Data driven. **Features**: Past trajectories and velocities. **Model(s)**: attention mechanism, and LSTM. **OH/PH**: 3.2/4.8 s. **Evaluation**: ADE and FDE.	**ETH and UCY** ADE/FDE: 0.46/0.92 m.
Holistic-LSTM [194]	**Approach**: Data driven. **Features**: bbox past trajectories, crossing intention, pedestrian scale, depth estimation, and global scene dynamics (depth and optical flow). **Model(s)**: ConvLSTM, modified LSTM with more inputs, and attention mechanism. **OH/PH**: 0.5/1 s. **Evaluation**: MSE, CMSE, and CFMSE of the bbox coordinates.	**JAAD** MSE: 389. **PIE** MSE: 167. **S-KITTI** MSE: 525/1.5 s.
Bi-TraP [195]	**Approach**: Data driven and Multi-modal goal estimation. **Features**: bbox past trajectories. **Model(s)**: CVAE, Gaussian distribution, GMM, and Bi-directional GRU. **OH/PH (JAAD/PIE)**: 0.5/1.5 s. **OH/PH (ETH/UCY)**: 3.2/4.8 s. **Evaluation**: ADE and FDE.	**JAAD** ADE: 1206. **PIE** ADE: 511. **ETH-UCY** ADE/FDE: 0.18/0.35.
BA-PTP [67]	**Approach**: Data driven. **Features**: vehicle odometry, bbox, body, head orientation, and pose. **Model(s)**: attention mechanism and Bi-GRU, **OH/PH (PIE)**: 0.5/1.5 s. **OH/PH (ECP)**: 0.6/1.6 s. **Evaluation**: MSE, CMSE, and CFMSE.	**PIE** MSE/CMSE/CFMSE: 420/383/1513. **ECP-Intention** MSE/CMSE/CFMSE: 768/680/1966
SGNet [196]	**Approach**: Data-driven, and goal-directed. **Features**: Past trajectories. **Model(s)**: Stepwise goal estimator, attention mechanism, GRU, and CVAE. **OH/PH (JAAD, PIE, HEV-I)**: 1.6/0.5,1.0,1.5 s. **OH/PH (ETH & UCY)**: 3.2/4.8 s. **OH/PH (NuScenes)**: 2/6 s. **Evaluation**: MSE, CMSE, CFMSE, ADE and FDE.	**JAAD** MSE/CMSE/CFMSE: 1049/996/4076 p (1.5 s). **PIE** MSE/CMSE/CFMSE: 442/413/1761 p (1.5 s). **ETH and UCY** ADE/FDE: 0.35/0.83 Euclidean space. **NuScenes** ADE/FDE: 1.32/2.50.
PTPGC [197]	**Approach**: Data driven. **Features**: Past trajectories, length of attributes, and number of pedestrians. **Model(s)**: Graph attention, convLSTM, and Temporal CNN. **OH/PH**: 3.2/4.8 s. **Evaluation**: ADE and FDE.	**ETH and UCY** ADE/FDE: 0.67/1.29.

Trajectory Prediction

Both traditional and DL techniques have been used to predict pedestrian trajectories. Traditional techniques rely on hand-crafted functions, such as EKF, IMM, and social forces, to predict pedestrians' future trajectories. However, these functions have limitations in handling complex scenarios. To address this, several researchers adopted DL techniques such as CNN, Generative Adversarial Network (GAN), GCNN, LSTM, GRU, CVAE, attention mechanism, and/or MLP.

Although LSTM networks have many advantages, they struggle to learn dependencies between multiple correlated sequences. For this reason, [181] proposed a Social LSTM network to predict pedestrian trajectories. Social pooling layers were introduced to enable LSTM networks to share their hidden state. This enables the algorithm to learn interactions among pedestrians. Social-LSTM only considers motion features to model human interactions, however, [151] argues that spatial po-

sition should also be considered. For this reason, they presented a model where MLP layers were used to encode location, and LSTM was used to encode motion for each neighbour. Both sets of encoded information were then used as input to a crowd interaction module to predict pedestrian displacement. In a different approach, [193] used two LSTM layers to encode the pedestrian's location and velocity, along with a temporal attention (TA) mechanism to extract the most relevant features from the velocity and location inputs.

Humans are highly dynamic, which makes the task of predicting their trajectories more challenging. In response to this, [187] implemented a DNN that would first predict the future destinations of the pedestrians, and then predict their future trajectories. They have used CNN, LSTM and Mixture Density Network to predict potential destinations, and another CNN to plan and predict future trajectories based on these potential destinations. CVAE was used by [146] to predict future endpoints, these then were subsequently used to predict multi-modal longer-term trajectories. They also presented a novel self-attention-based social pooling layers that extract relevant features from the neighbours using non-local attention. Yao et al. [195] also proposed a goal-direct method, where they combine CVAE and bi-directional GRU to encode past trajectories and decode multi-modal future trajectories. Goal-directed models have the disadvantage that only one goal is estimated over a long-term prediction. For this reason, if a pedestrian changes direction, the estimated goal may be incorrect, consequently affecting the estimated predicted trajectories. Wang et al. [196] proposed a method where they model and estimate goals continuously by using RNNs.

While many studies relied on historical trajectories to predict future ones, they often overlooked the current state of the pedestrian. In order to overcome this issue, [145] introduced a state refinement LSTM that considered both the current and previous state of the target pedestrian and the surrounding pedestrians. This state refinement module enables the network to incorporate interactions through a message-passing mechanism. It also uses a motion gate as an attention mechanism to focus on the most relevant features of the neighbours.

Previous research, when considering human-to-human interactions, would often take into account only nearby neighbours, even though more distant neighbours might also influence the behaviour of the target pedestrian. A GAN was presented by [153] that not only considers local neighbours but all neighbours in the scene. The GAN network comprises an LSTM generator to generate multi-potential trajectories, a pooling module to learn human-to-human interactions, and an LSTM discriminator to select acceptable trajectories from the generated ones. Similarly, [150] considered all the pedestrians in the scene using a spatio-temporal graph and LSTM. Additionally, they adopted an attention mechanism to learn the relevance of each agent, regardless of how far they are from each other. A star-like network was introduced by [149] to account for all agents in the scene. The network has a centralised hub network, which gathers motion information from all pedestrians in the scene, and a host network for each pedestrian. The host networks query the hub network for social information to predict trajectories. Graph attention and convolutional LSTM were also proposed by [197] to consider the surrounding neighbours.

Xue, Huynh, and Reynolds [135] emphasised the importance of considering scene layout when predicting pedestrian trajectories. As a result, they used three different LSTMs to learn information about individuals, social interactions, and scene

layout. One LSTM used the trajectory of the target pedestrian as its input, another used an occupancy map as its input, and the final one used feature vectors extracted from the original image by a CNN as its input. Likewise, [192] took scene layout into account, where they used a two-level grid structure of the original image and trajectory information as inputs to a two-stream LSTM for predicting future trajectories. CNN, LSTM, attention mechanism, and GAN were used by [152] to predict trajectories using both past trajectories and scene context as inputs. The CNN extracted scene-related features, the LSTM extracted motion-related features, the attention mechanism extracted both the physical and position relevant features, and the GAN generated multiple trajectories and then selected the most suitable ones.

Mohamed et al. [147] classified methods such as social pooling or the combination of hidden state features, used to model human interactions, as "aggregation methods". They claimed that these types of methods have limitations in accurately modelling human interactions because the aggregation occurs within the feature space and does not directly model physical interactions. Furthermore, some of these aggregation methods, such as pooling layers, may overlook to capture important information. Given these considerations, the authors proposed a social spatio-temporal GCN (ST-GCN) to model interactions among pedestrians. The ST-GCN model's output is subsequently used as input for a time extrapolate CNN to predict future trajectories.

The above works have not considered group-based interactions, which involve two or more individuals exhibiting similar movements, behaviours, or goals. A recursive social behaviour graph and GCN were implemented by [148] to explore and learn group-based interactions. The authors also used CNN and LSTM to obtain an individual representation of each pedestrian in the scene. The individual representations, along with the learned group-based features, were combined and used by a decoder LSTM to predict future trajectories.

Bhattacharyya, Fritz, and Schiele [189] claimed that they were the pioneers in using an on-board dataset to predict pedestrian behaviour. The authors used a two-stream LSTM architecture to encode bounding box coordinates, ego-vehicle odometry information, and feature vectors extracted from the original image by a CNN. Another work that used an on-board dataset is [67], in which the authors used a multi-stream RNN to individually encode bounding box coordinates, head orientation, body orientation, pose skeleton, and past trajectories. The encoded information from each stream is fused through an attention mechanism and subsequently input to an RNN decoder to predict future bounding boxes. The drawback of the latter two algorithms is that they did not consider social interaction among the agents.

Hasan et al. [191] argues that head orientation and movement are correlated. Consequently, they proposed a two-stream LSTM to encode both trajectory and head orientation information. The two encoded information, were then merged using a View Frustum social pooling layer. The disadvantage of this method is that it is only suitable for top-view and BEV datasets.

Usually, when a system adopts LSTM networks and requires the use of multiple types of inputs, these inputs are first combined before being fed to LSTM cells. This practice is required because LSTM cells are designed to accept only a single input sequence, which can constrain their ability to capture relevant information

Table 2.8: Results for the most relevant **pedestrian** trajectory prediction works.

Work	Dataset	OH	PH	ADE	FDE	AND	MAD	FAD	MSE
Social-LSTM [181]	ETH & UCY	3.2 s	4.8 s.	0.27 m	0.61 m	0.15 m	-	-	-
Scene-LSTM [192]	ETH & UCY	3.2 s	4.8 s	0.7 m	0.7 m	0.9 m	-	-	-
Social-GAN [153]	ETH & UCY	3.2 s	4.8 s	0.48 m	0.98 m	-	-	-	-
Social-attention [150]	ETH & UCY	3.2 s	4.8 s	0.30 m	2.59 m	-	-	-	-
Sophie [152]	ETH & UCY SDD	3.2 s	4.8 s	0.54 m 16.24 pi	1.15 m 29.38 pi	-	- -	- -	- -
StarNet-DNN [149]	ETH & UCY	3.2 s	4.8 s	0.30 m	0.57 m	-	-	-	-
PECNet [146]	ETH & UCY SDD	3.2 s	4.8 s	0.29 m 9.96 pi	0.48 m 15.88 pi	-	- -	- -	- -
ST-GCNN [147]	ETH & UCY	3.2 s	4.8 s	0.44 m	0.75 m	-	-	-	-
RSBG [148]	ETH & UCY	3.2 s	4.8 s	0.48 m	0.99 m	-	-	-	-
LVTA [193]	ETH & UCY	3.2 s	4.8 s	0.46 m	0.92 m	-	-	-	-
Bi-TraP [195]	ETH & UCY JAAD PIE	3.2 s 0.5 s 0.5 s	4.8 s 1.5 s 1.5 s	**0.18 m** 1206 511	**0.35 m** - -	- -	- -	- -	- - -
SGNet [196]	ETH & UCY JAAD PIE NuScenes	3.2 s 1.6 s 1.6 s 2 s	4.8 s 1.5 s 1.5 s 6 s	0.35 m - - 1.32	0.83 - - 2.5	- -	- -	- -	- 1049 442 -
SGNet [196]	ETH & UCY	3.2 s	4.8 s	0.35 m	0.83 m	-	-	-	-
PTPGC [197]	ETH & UCY	3.2 s	4.8 s	0.67 m	1.29 m	-	-	-	-
SS-LSTM [135]	ETH & UCY	3.2 s	4.8 s	0.070 npu	0.133 npu	-	-	-	-
SR-LSTM [188]	ETH & UCY	3.2 s	4.8 s	-	-	-	0.45	0.94	-
CIDNN [151]	ETH & UCY	4 s	4 s	0.11	-	-	-	-	-
MX-LSTM [191]	UCY Towncentre	3.2 s	4.8 s	-	- -	- -	0.49 m 1.15 m	1.12 m 2.30 m	- -
Holistic-LSTM [194]	JAAD PIE S-KITTI	0.5 s	1 s 1 s 1.5 s			-	- -	- -	389 167 525
BA-PTP [67]	PIE ECP	0.5 s 0.6 s	1.5 s 1.6 s			-	- -	- -	420 768

from various input sources. Quan et al. [194] adapted the conventional LSTM cell to accept four additional input sequences: vehicle speed, pedestrian intention, correlation among frames, and bounding box location. The vehicle speed was estimated by using optical flow and depth information; the pedestrian intention was estimated using convLSTM; and the correlation among frames was derived from optical flow images.

Table 2.8 and Figure 2.6 report the results for the most relevant studies in pedestrian trajectory prediction. It is not possible to directly compare all of them since some of them have used different datasets, metrics, OH, and PH. However, when examining the results of the algorithms that used the same dataset, metrics, OH, and PH, the Bi-Trap [195] algorithm outperformed others. Bi-Trap achieved

ADE and FDE values of 0.18 m and 0.35 m, respectively.

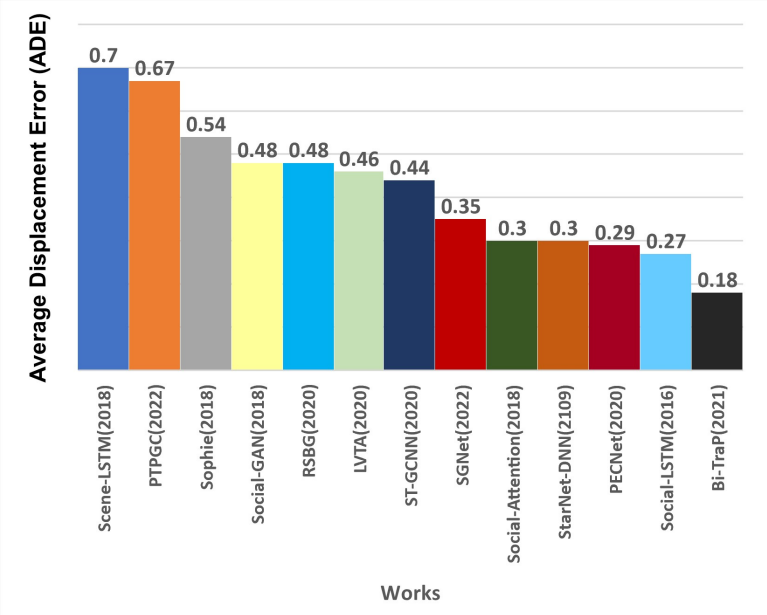


Figure 2.6: **Pedestrian** Trajectory Prediction Performance using the ETH and UCY datasets, with an OH of 3.2 s, a PH of 4.8 s, and Average Displacement Error (ADE) in metres (See Table 2.8).

Intention Recognition and Prediction

The difference between pedestrian intention recognition and prediction aligns with what was explained in Section 2.2.2. Recognition does not require anticipation, while prediction does. The main methods used to predict pedestrian intentions include CNN, GCNN, GRU, LSTM, attention mechanism, multi-tasking, and transformer networks.

CNN: Fang, Vázquez, and López [198] and Fang and López [183] used CNNs to extract human skeleton features and used SVM/RF classifier to predict if the pedestrian is crossing the road. Abdulrahim and Salam [199] also used CNNs, along with depth information, to learn 3D human body landmarks, including additional information such as the pedestrian shoulders, neck, and face. While CNNs can extract spatial features, their capability to capture temporal dependencies is limited. To overcome this limitation, [200] implemented a 3D-CNN to extract spatio-temporal information. Additionally, [133] proposed an alternative model called FuSSI-Net, designed to extract both spatio-temporal information. FuSSI-Net is a spatio-temporal Dense-net that takes a sequence of bounding boxes and skeleton features as inputs to predict crossing intention. Although these last two models can extract spatial and temporal information, they are limited to short-time horizon prediction and become computationally expensive as the input sequence length increases.

LSTM: Rasouli et al. [128] used LSTM to encode local context, trajectories, and EV information. Subsequently, the encoded information was decoded to estimate the probability of a pedestrian crossing the road. Bouhsain, Saadatnejad, and Alahi [131] used bounding box coordinates and velocities features as inputs for a sequence-to-sequence LSTM, which was used to predict both the pedestrian intentions and

the future position of the pedestrians' bounding boxes. In a different approach, [201], introduced a stacked-LSTM model, where appearance, context, and dynamic features of the pedestrian were used to predict crossing intentions. LSTM networks have the ability to learn and memorise features over the long term, as they capture long-distance dependencies [202]. Nevertheless, they have limitations in extracting spatial features, managing dependencies among the extracted features, exhibiting longer training times, and assigning uniform attention to all inputs, even though some inputs can be more relevant than others [111]. Ahmed et al. [203] used a 2D pose estimator in conjunction with LSTM to predict the crossing behaviour of the pedestrian.

Table 2.9: Relevant works for **pedestrian** intention prediction.

Work	Methods	Problem	Dataset/Results
[184]	Approach: Dynamic. **Features**: Recursive Bayesian filters – Compared EKF and IMM filters (constant velocity/acceleration/turn). PH: < 2 s. **Evaluation**: MLPE.	Trajectory and intention prediction.	**Daimler** IMM has not shown significant performance over simpler models.
[185]	Approach: Dynamic. **Features**: Compared the performance between GDPMs using optical flow information, PHTM, KF and IMMKF. Provided human performance on classifying pedestrian behaviour prediction. **Evaluation**: Mean Combined Longitudinal and Lateral RMSE.	Trajectory and intention prediction.	**OWN** (on-board) GDPM and PHTM showed better accuracy, however, they are more computationally expensive. 10-50 cm Time Horizon 0.77 s.
[204]	Approach: Dynamic + Context. **Features**: distance and time to curb, distance and time to the ego lane, distance and time to the zebra crossing, distance and time to the collision point, a difference of time to the collision point, face, global and relative orientation. A single neural network is used as a classifier to learn the different features. Inner-city and zebra model. PH: 1 s. **Evaluation**: TPR and FPR.	Intention prediction (crossing).	**OWN** (on-board) Inner-city dataset, zebra dataset and combination of both ICZ. Inner-city model: 31% TPR, 0.0 FPR, PH 0.72 s for the zebra dataset. TPR 29%, PH 0.67 s for the inner-city dataset. TPR 31%, PH 0.72 s for the ICZ dataset. Zebra crossing model: 100% TPR, 3.23 s PH for the zebra dataset. 86% TPR, 28% FPR and 1.73 s PH for the inner-city dataset. CMT model: 62% TPR, 2.59 s PH for the ICZ dataset.
[205]	Approach: Dynamic + Context. FLDCRF. **Features**: pedestrian position (distance to curb, and left or right side of the road), pedestrian-vehicle interaction, optical flow. **Evaluation**: average probability, time to stop and time to cross.	Intention prediction.	**NTUC** (OWN, on-board and actors) Average probability $>$ 0.7 predicting 1.2 s before the action.

Continued on next page

Table 2.9 – continued from the previous page.

Work	Methods	Problem	Dataset/Results
[206]	Approach: Dynamic. Balanced-GDPMs to reduce 3-D time relevant information into low dimensional information and to assume future latent positions. **Features**: Skeleton motion analysis. Four models to predict start, stop, walk and stand actions. HMM is used to select which model to use to predict future pedestrian paths and poses. **Evaluation**: MED against TTE.	Predict pedestrian actions.	**CMU-UAH** Achieved MED of 41.24 mm for TTE of 1 s, for starting activity; and MED of 238.01 mm for TTE of 1 s for stopping activity.
[198]	Approach: Data Driven. **Features**: Skeleton. CNN for pose estimation. Deep association for tracking. **Evaluation**: Intention probability vs TTE.	Intention prediction (crossing/not crossing).	**Daimler** 0.8 predictability with TTE=12 (750 ms).
CV [183]	Approach: Data Driven. **Features**: Skeleton. CNN for pose estimation. Deep association for tracking. **Evaluation**: Accuracy.	Intention prediction (crossing/not crossing).	See Table 2.10
PIE (int) [128]	Approach: Data driven. **Features**: bbox coord, image context, and image bbox. RNN (LSTM). **Evaluation**: Accuracy, and F1-score.	Intention prediction (crossing).	See Table 2.10
[131]	Approach: Data Driven. **Features**: bboxes coordiantes adn velocities. PV-LSTM Multi-task sequence to sequence learning **Evaluation**: ADE, FDE, Accuracy.	Pedestrian intention and pedestrian bbox predictions (crossing).	See Table 2.10.
[207]	Approach: Context, Temporal, and Data driven. **Features**: Graph Convolution and GRU to learn spatio-temporal relationships. **Evaluation**: Accuracy.	Intention prediction (crossing).	**Stanford-TIR** A: 79.10%. **JAAD** A: 79.28%.
[208]	Approach: Data driven. **Features**: pedestrian body landmarks considering depth information. CNN. **Evaluation**: Accuracy.	Intention prediction (walking and crossing).	**OWN** (on-board) A: 89%.
FUSSI-net [133]	Approach: Data driven, target-agent context. **Features**: Skeleton and bbox. DenseNet. **Evaluation**: Accuracy.	Intention prediction (crossing).	See Table 2.10
SFR-GRU [132]	Approach: Data driven. **Features**: pose, 2D bbox, appearance, global context, and ego speed. Stacked-RNN (GRU). **Evaluation**: Accuracy, Precision, recall, F1-score, and AUC.	Intention prediction (crossing).	See Table 2.10
C+B+S+Int [130]	Approach: Data driven. **Features**: surrounding, appearance, context, bbox, and EV speed. single GRU. PH: 2 s. **Evaluation**: Accuracy, AUC, F1, Precision, and recall.	Intention prediction (crossing). Studied human performance.	See Table 2.10
[209]	Approach: Data driven and key body landmarks. **Features**: PAF and PIF. Uses only one RGB image. Multitask learning. CNN (ResNet). **Evaluation**: Precision for different prediction horizons.	Recognition and Intention prediction (crossing) in real-time.	**JAAD** Recognition: -0 s: 81.7%; -1 s: 83.6%; -2 s: 83.5%; -3 s: 83%; -4 s: 82.7%. Prediction: -1 s: 42.6%; -2 s: 46.1%; -3 s: 46.3%; -4 s: 46.0%. FPS: 5.

Continued on next page

Table 2.9 – continued from the previous page.

Work	Methods	Problem	Dataset/Results
[210]	Approach: Data Driven. **Features**:: pose-key-points. Compared SVM, RF, GBM, and XGBoost models. **Evaluation**: Accuracy.	Intention prediction (crossing at a red light).	**CCTV** A: 92%: 1 s**;** 92%: 2 s**;** 88.9%: 3 s**;** 92.5%: 4 s.
PCIR [200]	Approach: Data driven, context, and behavioural. **Features**: pedestrians, EV, and environment. 3D-CNN. **Evaluation**: AP.	Intention detection (crossing).	See Table 2.10
[211]	Approach: Data driven. **Features**: bbox, body pose, road objects. Graph encoder, CNN, and LSTM. PH: 1.5 s. **Evaluation**: Balanced Accuracy and F1 score.	Intention prediction (crossing).	See Table 2.10
I+A+F+R [122]	Approach: Data driven, and multi-task. ARN Attentive Relation Network. CNN, MLP, and GRU. PH: 1-2 s. **Features**: bbox context and coordinates, relation, and visual. **Evaluation**: Accuracy, F1-score, ROC-AUC, precision.	Intention and action prediction (crossing).	See Table 2.10 Inference: $<$ 6 ms.
PCPA [76]	Approach: Data driven. **Features**: bbox, pose, local context, and EV speed. 3D CNN + single-RNN (GRU) + attention mechanism. **Evaluation**: Accuracy, AUC, and F1.	Intention prediction (crossing).	See Table 2.10
[126]	Approach: Data driven. **Features**: local and global context, bbox, pose-key-points. Attention mechanism, 2D CNN, and RNN. **Evaluation**: Accuracy, F1, and recall.	Intention prediction (crossing).	See Table 2.10
Graph+ [212]	Approach: Data driven. **Features**: context, EV velocity, and key body landmarks. Graph Convolutional Network. **Evaluation**: Accuracy.	Intention Prediction (crossing).	See Table 2.10 Inference: 6 ms.
ST-CrossingPose [124]	Approach: Data driven. **Features**: skeleton-based. Spatio-Temporal GCN. **Evaluation**: Accuracy, AUC, F1-score, Precision, and Recall.	Intention prediction (crossing).	**JAAD** Recognition: 63%. See Table 2.10
[123]	Approach: Data Driven. **Features**: bbox. Transformer Networks. PH: 1 s and 2 s. Test human ability for pedestrian action prediction. **Evaluation**: Accuracy and F1-Score.	Intention recognition and prediction (crossing).	**PIE** A:91%. F1:0.83. **CP2A** A:91%. F1:0.91.
Scene-STGCN [127]	Approach: Data Driven. **Features**: Scene Spatio-Temporal GCN. **Evaluation**: Accuracy, F1-score, AP, and ROC-AUC.	Intention recognition (crossing).	See Table 2.10
[125]	Approach: Data driven. **Features**: body land-marks. SqueezeNet and GRU. Hardware: AMD Ryzen 5 3600, G Force RTX 3070. **Evaluation**: Accuracy and ROC-AUC.	Intention prediction (crossing). Lightweight and inference speed.	See Table 2.10
CA-LSTM [201]	Approach: Data driven, context and dynamic. **Features**: appearance, velocity, and walking angle. Attention LSTM. **Evaluation**: Accuracy, F1-score, recall metrics.	Intention Prediction (crossing).	See Table 2.10

Continued on next page

Table 2.9 – continued from the previous page.

Work	Methods	Problem	Dataset/Results
[134]	Approach: Data driven. **Features**: pedestrian localisation and environment contest (lane lines). ML and DL. **Evaluation**: Accuracy.	Intention recognition in real-time.	See Table 2.10
[213]	Approach: Data driven. **Features**: pedestrian pose (skeleton), pedestrian to vehicle distance, and EV information. Multi-feature fusion. Random forest classifier. PH: 0.6 s. **Evaluation**: Accuracy and AUC.	Intention prediction (crossing).	See Table 2.10
[203]	**Approach**: Data driven. **Features**: Past trajectories, velocity, and 3D joint estimation. **Model(s)**: Position and Velocity LSTM. **PH**: 0.4 s. **Evaluation**: Accuracy.	Intention prediction (crossing).	**JAAD and PIE** Accuracy: 89%/91%.

GRU: GRUs are an alternative to LSTMs, as they also learn temporal information. Kotseruba, Rasouli, and Tsotsos [130] used pedestrian appearance features, which were extracted using a VGG network, and EV velocity information as inputs for a GRU network to predict pedestrian intentions. Rasouli, Kotseruba, and Tsotsos [132] used pedestrian appearance, global context, body pose, bounding boxes, and ego-vehicle speed features as inputs to a stacked GRU network to predict pedestrian crossing behaviour. These features were gradually integrated into the GRU network, starting with pedestrian appearance, followed by global context, body pose, bounding boxes, and concluding with the EV speed. GRUs offer the advantage of requiring less memory and being faster than LSTMs. However, they tend to be less accurate when handling long input sequences [202].

GCN: A spatio-temporal GCN was presented by [124], where they used a sequence of skeleton features to predict crossing intentions. The skeleton joints were connected by nodes and edges to learn both spatial and temporal features. Cadena et al. [212] used two GCNs, which took human body key points, local context, and ego speed information as inputs to predict crossing intentions. GCNs have the advantage of extracting interactions among the target pedestrian and its neighbours, considering both spatial and temporal dependencies [111]. In addition, GCNs can handle non-Euclidean data formats, such as scenarios where pedestrians are dispersed across a scene, which cannot be represented using a grid-like structure. However, they can only handle short-term sequences and perform poorly when applied to regression tasks.

Attention Mechanism: [201] also used a self-attention mechanism to extract the most relevant information from the pedestrian's appearance, the pedestrian's surroundings, and dynamic features. Rasouli et al. [128] combined different attention mechanism layers at different network locations to investigate their impact on the model performance. Attention mechanism approaches enable networks like LSTM to focus more on the most relevant features and less on redundant ones.

Transformers: Even though the attention mechanism has the ability to focus on the most relevant features, it was reported by [123] that its effectiveness might be reduced when coupled with LSTM networks. For this reason, [123] proposed a framework based on three types of transformer networks: encoder-only, encoder-pooling, and encoder-decoder architectures. The proposed framework used only the

pedestrian bounding box information as its input. The authors argued that their model outperformed other methods that used multiple input features. Transformer networks offer the advantage of parallel input processing, accelerating the training stage. On the other hand, the ability to process the input data in parallel restricts the model from taking advantage of the input's sequential nature.

Multiple Methods: many studies have used more than one method to predict pedestrian intention. Liu et al. [207] used GCN to generate a pedestrian-centring graph for each observation frame. These graphs connect the target pedestrian to its surroundings, allowing the algorithm to learn the relation between the pedestrian and the scene. In addition, edges were introduced between the pedestrian nodes in each pedestrian-centring graph to allow the algorithm to learn temporal information. The resulting interconnected graphs were then fed into a GRU network to predict crossing intention. Chen, Tian, and Ding [211] used a combination of methods, including a CNN to extract features from traffic objects and pedestrian appearance, a GCN to auto encode the extracted features, another framework to extract human skeleton, and an LSTM network to predict crossing intentions. CNN, ARN, MLP and GRU were used by [122] to predict crossing intentions. CNN was used to extract global features, ARN was used to extract relational features from detected traffic objects, MLP was used for intention classification, and LSTM was used for intention prediction. One major difference of this work is that the network also takes the predicted intention output as input. Kotseruba, Rasouli, and Tsotsos [76] used 3D-CNN, RNN, and attention mechanism. The 3D-CNN was used to encode local features from a sequence of cropped bounding boxes, and the RNN was used to encode the bounding-box coordinates, pose landmarks, and the ego-vehicle speed. Finally, an attention mechanism was used to combine the most relevant features. Yang et al. [126] used 2D-CNN, stacked-RNN, and attention mechanism. Spatio-temporal GCN was used by [127] to encode the input image, image class and location information tensors. Then, the output of the spatio-temporal GCN was fed into an LSTM network to generate long-term predictions. Zeng [125] used SqueezeNet to extract visual features and used GRU to extract temporal dependencies. They also used a multi-tasking approach to predict both pedestrians' intentions and poses. One primary advantage of using multiple models is that each model can compensate for the limitations of others. For example, CNN, GCN, and attention mechanisms can aid the limitations of an LSTM network in extracting spatial information, handling non-Euclidian data, and prioritising relevant features, respectively.

Complete Pipeline: Gazzeh and Douik [134] presented a complete pipeline model which includes detection, tracking, and crossing intention prediction. They used YOLOv4 for object detection, DeepSort for tracking, Canny Edge for lane line detection, and linear SVM for intention prediction. Another complete pipeline system was implemented by [133], using YOLOv3 for detection, DeepSort for tracking, and spatio-temporal Densenet for intention prediction. YOLOv5, DeepSort, and an LSTM network with an attention mechanism were used by [201] to detect, track, and predict pedestrian intention, respectively. Razali, Mordan, and Alahi [209] implemented a multi-task network designed to recognise pose states and predict pedestrian intentions. In this network, ResNet was responsible for feature extraction. Part-Intensity Fields (PIFs) and Part-Association Fields (PAFs) were used to generate channels and detect pose joints. A separate head network was then utilised to predict pedestrian intentions.

Table 2.10 presents the results achieved by the most relevant pedestrian intention prediction works in the literature. Unfortunately, direct comparisons between these studies are impossible due to variations in different problem formulations, OH, TTE, datasets, and metrics. For example, the work that achieved the best accuracy was [124]. However, the authors used their own dataset. The second best was [131], but they used an OH and TTE of 0.6 s.

Table 2.10: Results for the most relevant **pedestrian** intention prediction works.

Work	Dataset	Obs. Hor.	TTE	Acc (%)	AUC (%)	F1 (%)	Rec. (%)	Prec (%)	ROC-AUC(%)
[134]	JAAD	-	Recog.	92.88	-	-	-	-	-
[183]	JAAD	0.5 s	Next-Frame	88	-	-	-	-	-
STRR-Graph [207]	JAAD	0.5 s	Next-Frame	76.98	-	-	-	-	-
FUSSI-net [133]	JAAD	0.5 s	Next-Frame	76.6	-	-	-	-	-
PIEint [128]	PIE	0.5 s	Next-Frame	79	-	87	-	90	73
CA-LSTM [201]	JAAD	0.5 s	Next-Frame	89.68	-	75.38	**85.96**	-	-
PV-LSTM [131]	JAAD	0.6 s	0.6 s	91.48	-	-	-	-	-
[213]	BPI	-	0.6 s	89.5	**99.2**	-	-	-	-
SFR-GRU [132]	PIE	0.5 s	2 s	84.4	82.9	72.1	80	65.7	-
C+B+S+Int [130]	PIE	0.5 s	2 s	83	85	81	85	79	-
PCIR [200]	JAAD	-	-	89.6	-	-	-	-	-
[211]	PIE	0.5 s	1.5 s	79	-	78	-	-	-
I+A+F+R [122]	JAAD PIE	0.5 s -	1-2 s	87 84	92 88	70 **90**	- -	66 **96**	- -
[126]	JAAD PIE	0.5 s	1-2 s	83 89	82 86	63 80	81 81	51 79	- -
GRAPH+ [212]	JAAD PIE	0.5 s	1-2 s	86 89	88 90	65 81	75 79	58 83	- -
[123]	PIE	0.5 s	1-2 s	91	91	83	-	-	-
Scene-STGCN [127]	PIE	0.5 s	1-2 s	83	-	89	-	**96**	**85**
PCPA [76]	JAAD PIE	0.5 s -	0.5-1 s	85 87	86 86	68 77	- -	- -	- -
ST-CrossingPose [124]	OWN	0.5 s	1 s 2 s	**92** **92**	84.9 84.1	83.7 79.7	81.8 79.7	85.9 81.3	- -
[125]	JAAD	-s	-s	84	-	-	-	-	**85**

2.2.4 Heterogeneous Road Agents

All the previously mentioned works primarily focused on predicting the behaviour of either pedestrians or vehicles. However, in a real-world traffic scenario, complex interactions occur among various types of agents, each with different dimensions and dynamics. Consequently, it is crucial to consider the interaction between heterogeneous agents. Several works have addressed the detection and behaviour prediction of heterogeneous agents.

For example, authors [214] introduced the **TrafficPredict** algorithm, which was developed to learn motion patterns and predict the trajectories of different types of traffic agents, including pedestrians, bicycles and cars. They adopted the 4D Graph network in conjunction with an RCNN LSTM to learn the movements and traffic agents' interactions. The authors used an OH of 2 s to predict a horizon of 3 s. They achieved a state-of-the-art average displacement error of 0.085 and a final displacement error of 0.141. **DeepTAgent** is another heterogenous system presented by [215] in which they used Mask R-CNN to detect objects, a CNN to extract tracking features, and a Heterogenous Interaction Model (HTMI) that considered collision avoidance behaviour to predict the agents' position, velocity and subsequently their trajectory and interactions. The authors [216] presented a hybrid network for predicting the trajectory of road agents and modelling their interactions. They used a CNN to capture local information, such as the agent's shape and position, and an LSTM network for trajectory prediction. In dense, diverse traffic situations, the algorithm demonstrated a notable performance of 30% over state-of-the-art methods. However, it did not outperform the state-of-the-art algorithms in sparse and homogeneous traffic scenes. Li et al. [217] presented a framework called **EvolveGraph**. In this framework, they encoded an observation graph to infer an interaction graph and decoded both the observation and interaction graphs to predict future trajectories. Zhang et al. [180] implemented the Attention-based Interaction-aware Trajectory Prediction (AI-TP) model. This model used a Graph Attention Network (GAT) to represent interaction among heterogeneous traffic agents and used a Convolutional GRU (ConvGRU) to make predictions. A multi-agent trajectory prediction system was performed by [218] where a three-channel framework accounted for dynamics, interactions and road structure. Moreover, a novel Heterogeneous Edge-enhanced graph ATtention network (HEAT) was proposed to extract interaction features. Dynamic features were extracted from the agents' previous trajectories. Interaction patterns were represented through a directed edge-feature heterogeneous graph and extracted with the HEAT network. The road structure information was shared among all agents using a gate mechanism. Finally, all the information acquired from the previous process was combined to predict trajectories.

All the previously cited works have predicted the trajectories and interactions among the agents. However, they have not taken into consideration their intentions, such as crossing/not crossing or braking/non-braking. Also, they have not incorporated information from static road objects such as traffic lights and road signs. Static road traffic objects are crucial in directing, informing, and controlling road users' behaviour. Furthermore, there is limited research on using detection and prediction information to identify potential and developing hazards.

The authors [219] proposed a multi-task learning model that combines both object detection and distance prediction to identify dangerous traffic road objects. They used SSD CNN to detect cars, vans, and pedestrians. The input image was divided into a grid map with four vertical and three horizontal distances. Depending on the category of the TV and its location, the network assigned a danger level using blue, green, yellow, and red bounding boxes, where blue and red represented the least and the most dangerous levels, respectively. However, predicting the TV's velocity using a grid map limits the velocity resolution and might not give realistic measurements. Also, relying solely on the distance between the ego and the target vehicle is insufficient. For example, an EV might maintain a safe distance from the

TV, but the TV can suddenly brake and change its velocity. Therefore, it would be beneficial for the EV to predict and recognise instances when the TV is braking or experiencing a sudden change in velocity.

Authors [220] considered themselves pioneers in combining object detection and intention recognition to assess the risks in complex traffic scenarios. Their objective was to detect both non-static objects, such as vehicles and pedestrians, and static objects, such as traffic lights, and then use the gained information to evaluate potential hazards ahead. In order to detect the objects, they used the YOLOv4 and the BDD100K dataset and achieved an mAP of 52.7%. For recognising the pedestrian intention (crossing or not-crossing), they used VGG-19 CNN and Part Affinity fields, achieving an accuracy of 97.5%. To predict vehicle intentions, including braking and turning, they employed the EfficientNet CNN, achieving a recognition accuracy of 94%. Lastly, for recognising traffic lights' state (red, green, or amber), they used the MobileNet CNN, achieving an accuracy of 97.75%. Nevertheless, using only the brake and the turn signal lights information to predict vehicle behaviour and assess danger is not sufficient since braking behaviour can exhibit varying intensities. For example, normal braking, characterised by a gradual decrease in the vehicle's velocity, is typically considered a potential hazard. In contrast, harsh braking, involving a sudden and significant change in the vehicle's velocity, is considered a developing hazard. Furthermore, there are situations where the TVs abruptly change their direction without using their turn signal, posing a developing hazard. Therefore, the EV must be capable of detecting sudden changes in the vehicle's direction and velocity. Similarly, depending only on pedestrian crossing/not crossing intentions limits the system to make a long prediction horizon, as pedestrians can cross at different velocities, and may suddenly change their goal destination.

2.2.5 Traffic Behaviour Dataset

There are many publicly available datasets to deal with the different IV tasks such as processing raw sensor data, perceiving low-level features (stereo vision, 3D vision, optical flow), perceiving high-level features (object detection, object tracking, lane/road detection, semantic segmentation), and analysing the behaviour of other traffic road users (activity, attention, intent, and style). However, [60] performed an in-depth study to compare 45 traffic road datasets, and concluded that most of the available datasets are to solve low-level and high-level perception tasks. The reviewed datasets, including behavioural label information, are Brain4cars, DIPLECS, DR(eye)ve, EISATS, JAAD, PIE, PREVENTION, and UAH.

The Brain4cars dataset was developed to predict the behaviour of the EV's driver, including left lane change, right lane change, left turn, right turn, and driving straight. The ADAS system then uses the predicted behaviour information to alert the driver if their intended manoeuvre would pose a traffic hazard. This dataset comprises images capturing the road ahead and the driver's face inside the EV. The dataset consists of 1180 miles driven by 10 different drivers. The driven miles include motorways and urban environments [221].

The DIPLECS project comprises three datasets: one dataset was recorded on a Swedish road, another on an English countryside road, and the third from an indoor track. All the datasets were acquired using camera sensors and were designed to predict the turning and braking intention of the EV's driver [222].

The DR(eye)ve dataset was developed to predict where the EV's driver pays more attention in a traffic scene. The dataset was acquired using a front camera to record the road ahead and eye-tracking glasses to extract the EV's driver's eye-tracking information. It comprises 74 videos of 5 minutes each [223].

The EISATS dataset has been archived, and it is no longer available.

The JAAD and PIE datasets were developed to recognise and predict the crossing behaviour of pedestrians. It provides labels for crossing and not crossing, for pedestrian action behaviour such as looking, standing, and walking, and for scene contexts such as narrow or wide roads, pedestrian crossing signs, stop signs, traffic lights, parking lots signs, and zebra crossing. The JAAD dataset contains 346 video clips with a length of 5 to 10 seconds [224], and the PIE dataset contains 300K labelled video frames with 1842 pedestrian samples, and it is one of the largest datasets studying pedestrian behaviour [128].

The PREVENTION dataset was developed to study the surrounding vehicles' lane change behaviour. It provides annotations of vehicle trajectories, vehicle types, and lane information. The dataset was recorded in a motorway environment and contains approximately 356 minutes of recording. The sensors used to acquire the dataset were LIDAR, radar, and cameras.

The Honda Research Institute Driving Dataset (HDD) was created by [225] to enable studies on learning the behaviour of the EV's driver in real traffic scene environments. The dataset has 30 classes that belong to one of the following driving behaviours: goal-oriented action, stimulus-driven action, cause, and attention.

2.2.6 Discussion

This paper has surveyed several works that investigate the prediction of pedestrian and vehicle behaviour. Based on the findings, this section presents a general framework diagram, outlines risk assessment, discusses challenges, examines techniques, outlines requirements, and suggests potential future directions for pedestrian and vehicle behaviour prediction systems.

General Framework for a Behaviour Prediction System

A proposed general framework for a behaviour prediction system is depicted in Figure 2.7. The camera sensor outputs RGB images, which are used by the detection and image processing algorithms.

The detection algorithm detects both static and non-static road objects, including road lanes, vehicles, vulnerable road users, traffic lights, and road signs. The position information of the detected objects, represented by bounding boxes, is then used by a tracking algorithm to assign each object a unique ID. This ID assignment enables the system to track past trajectories of each detected object, which serves as input for subsequent processing.

The image processing algorithm uses the RGB images from the camera sensor and the past trajectories of the detected objects to generate optical flow, depth, appearance, global and local context images. An example of how image processing uses past trajectories is using the bounding box information to crop the RGB image at the specific location of the detected object. This cropping operation provides local context information for further analysis and decision-making within the system.

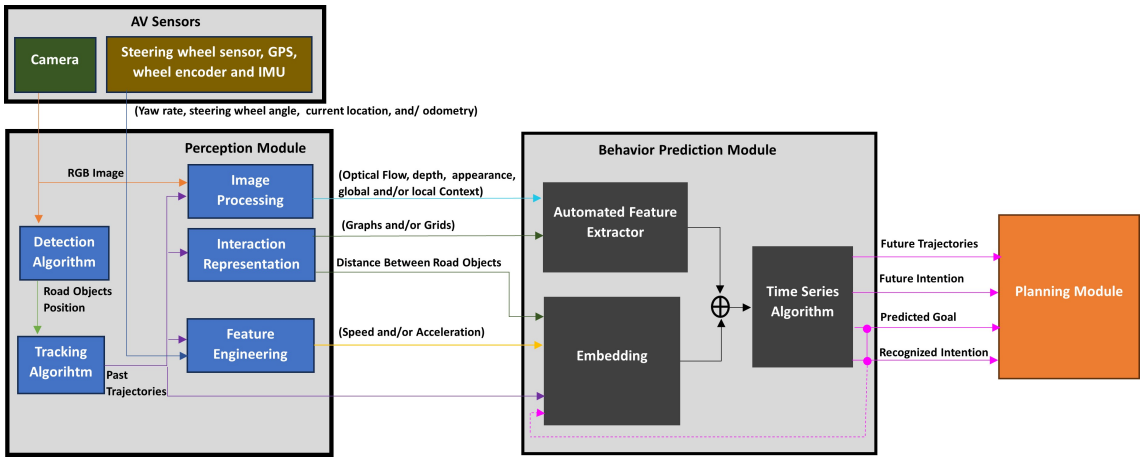


Figure 2.7: General behaviour prediction framework. The behaviour prediction module **consists** of an automated feature extractor (CNN, 3D-CNN, GCN, FCN, CVAE, GAN, etc.), an embedding layer (FCN and ANN), and a time series algorithm (RNN, GRU, and LSTM). It is **dependent** on the perception module (Detection, tracking, image processing, interaction representation, and feature engineering), which is dependent on the EV sensors (camera, GPS, and wheel encoder). Additionally, the outputs of the behaviour prediction modules are **sent** to the planning module.

The interaction representation algorithm uses the objects' past trajectories to calculate distances between the traffic agents, construct graph networks with vertices and edges, and generate grid maps that account for interactions between traffic agents.

The feature engineering algorithm uses objects' past trajectories and internal sensor data from the IV (e.g., steering wheel angle, yaw rate, wheel encoder, etc.) to derive additional features. For example, it uses the differences between the objects' positions between consecutive frames to calculate their velocities.

The outputs of the perception module are then fed into the automated feature extractor and the embedding algorithms within the behaviour prediction module. Automated feature extractors are deep learning algorithms designed to generate feature vectors representing spatial properties of the inputs. Embedding uses a linear transformation to transform the inputs into a desired output feature size. The time series algorithm uses the combined feature vectors generated by the automated feature extractor and the embedding layer to learn temporal information, enabling it to predict various aspects of object behaviour, including future trajectories, future intentions, goals, and current intentions. Note that the embedding layer and the time series algorithm can use the predicted goals and recognised intentions as extra information for predicting future trajectories.

Finally, the outputs of the behaviour prediction module are then used by the IV's Planning module, which in turn uses this information to plan the IV's actions to achieve its final goal.

Risk Assessment for Behaviour Prediction System

Authors [226] proposed a risk assessment for an IV. They mentioned that IV failures can arise from various aspects, including vehicular components such as hardware,

software, mechanical systems, communication infrastructure, and interactions between the passenger and the IV Human Machine Interface system. Based on their findings, this paper presents a risk assessment for an IV behaviour prediction system. This assessment identifies, analyses, and provides recommendations for mitigating and controlling these identified risks.

Risk Identification

Based on the general framework for a behaviour prediction system depicted in Figure 2.7, the following risks have been identified:

- Camera sensor failure: this includes hardware malfunctions, blocked field of view, and noise (electricity, heat, and illumination).
- Computing components failure: computer or GPU failure.
- Sensor Failure: Failure in the steering wheel, wheel encoder, GPS, and IMU sensors.
- Detection algorithm failure: missed detections, poor intersection over union, false-positive and false-negative classification.
- Tracking algorithm failure: missed tracking and incorrect association of objects between frames. For instance, an object might not be tracked in the next frame, or objects might swap their IDs due to overlap.
- Image processing failure: incorrect optical flow and depth estimation.
- Interaction representation failure: noisy and incorrect distance calculation and incorrect graph or grid representation of the object interactions.
- Feature Engineering failure: redundant features, noisy estimates speed and acceleration due to poor detection and tracking performance.
- Cybersecurity failure: remote hacking, vehicle spoofing, insider threat, and tampering with sensor data.

Risk Analysis

The authors [226] discussed several methods for analysing risks in automotive contexts, including situation-based analysis, ontology-based analysis, failure modes and effects analysis (FMEA), and fault tree analysis (FTA). Their investigation concluded that FTA is the most suitable method for conducting a risk assessment on IV features. For this reason, this paper also adopts FTA to perform a risk analysis on the behaviour prediction system. FTA methods have the following advantages: being event-orientated, enabling the diagnosis of the root cause of failures, facilitating an understanding of how subsystems can impact each other, having a straightforward and graphical nature for ease of comprehension, and aiding in decision-making regarding the control of identified risks. The proposed FTA is depicted in Figure 2.8. A qualitative analysis of the proposed FTA reveals that the system is highly vulnerable because any failure occurrence of the basic events (EVX) can lead to the

failure of the behaviour prediction system. For instance, if the detection algorithm fails, it can cascade failures throughout the tracking algorithm, image processing, interaction representation, and feature engineering, resulting in the failure of the behaviour prediction system.

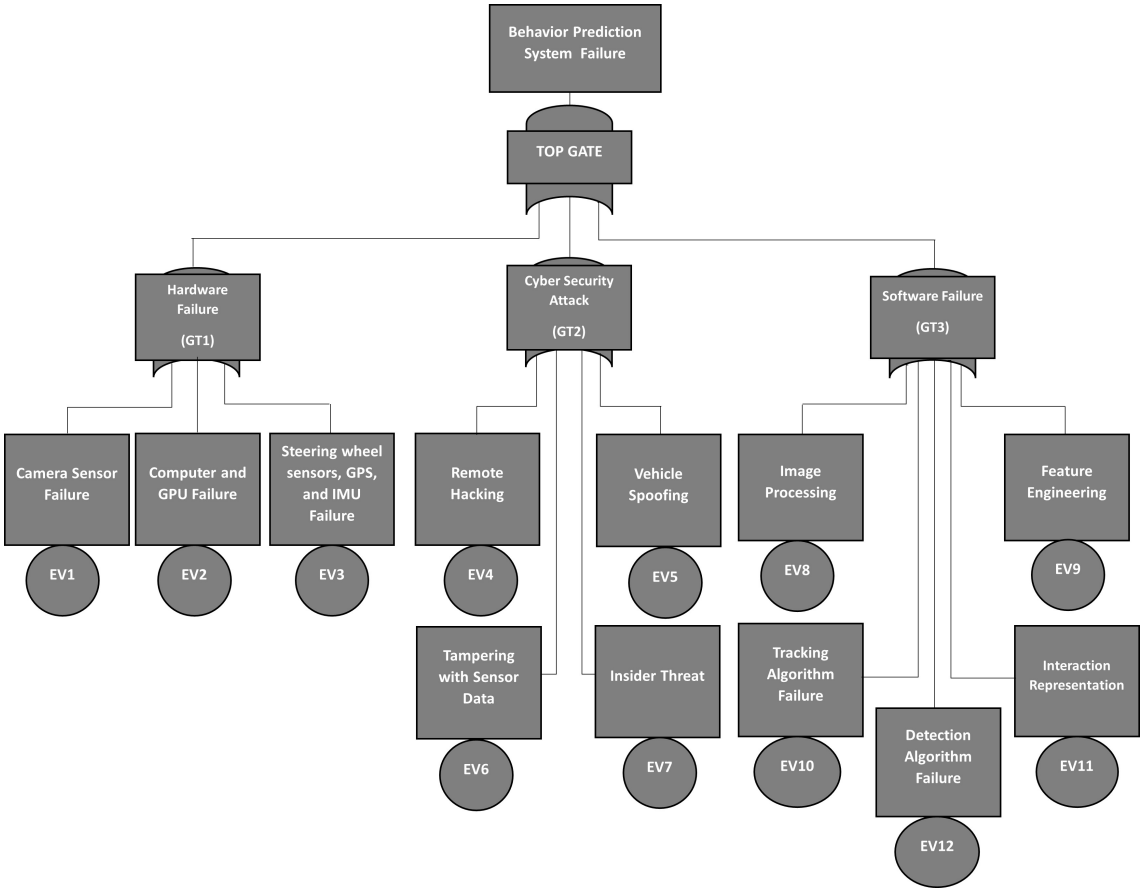


Figure 2.8: Fault tree analysis for a Behaviour Prediction System. The circle shapes with the square shapes are the basic events that may lead to failures on the top events. The square shape after the TOP GATE is the top event, which indicates the failure of the behaviour prediction system. The "OR" gates mean that if one of its input events occurs, it will output an event that is true.

In order to quantitatively analyse the behaviour prediction system, it is required to know the probability of failure for each event (EVX), which depends on the hardware, software, and cybersecurity in use. However, a general mathematical model to calculate the overall system failure from an FTA diagram depicted in Figure 2.8 is given by the following equation [227, 228],

$$Q_0(t) \leq (1 - \Pi_{j=1}^{k}[1 - \check{Q}_j(t)]) \tag{2.3}$$

where $Q_0(t)$ is the top event (failure of the behaviour prediction system), $\check{Q_j}(t)$ is the failure probability of a minimal cut-set. For instance, the probability that the TOP GATE in the proposed FTA diagram happens is given by,

$$Q_0(t) \leq (1 - [1 - P(GT1)] * [1 - P(GT2)] * [1 - P(GT3)]) \tag{2.4}$$

where

$$P(GT1) = (1 - [1 - P(EV1)] * [1 - P(EV2)] * [1 - P(EV3)]) \tag{2.5}$$

and,

$$P(GT2) = (1 - [1 - P(EV4)] * [1 - P(EV5)] * [1 - P(EV6)] * [1 - P(EV7)]) \tag{2.6}$$

and

$$P(GT3) = (1 - [1 - P(EV8)] * [1 - P(EV9)] * [1 - P(EV10)] * [1 - P(EV11)]). \tag{2.7}$$

Risk Control

Based on the identification and analysis of risks, it has been concluded that a behaviour prediction system is vulnerable. Below are some recommendations to mitigate these risks:

- Given that any hardware failure can cause a top event, it is recommended to have backups for hardware components with a high probability of failure, for example, to have an extra camera sensor. The disadvantage of this approach is that it is expensive and requires more space in the vehicle.

- For the general prediction behaviour system in question, it is observed that it relies on three types of information (RGB image, engineering feature, and interaction) for predictions. Therefore it is recommended to enable the system to function in a degraded mode by using one or two pieces of information if one of them fails.

- The detection and tracking algorithms are important for the system, as their outputs are used by the other algorithms. Thus, it is recommended to use sensor fusion since if one of the hardware or the algorithms responsible for detecting and tracking the object fails, the system can work in a degraded mode.

Behaviour Prediction System Challenges

Table 2.11 summarises the main challenges in the research of behaviour prediction of traffic agents. These challenges are categorised into target agents, systems, resources, and uncertainties. Target agents refer to the unique characteristics of these agents that make their behaviour challenging to predict. System challenges are related to the inherent characteristics of the system, considering its design and evaluation. Resource challenges are associated with the hardware and data required for training and operating the system. Uncertainties include events such as hardware malfunctions, cybersecurity vulnerabilities, and software failures.

Table 2.11: Behaviour Prediction Research Challenges.

Type of Challenge	Class	Challenges
Target Agents	*Pedestrian*	Highly dynamic, can move in many directions and change them very quickly, be easily occluded, be distracted by their own objects or external environments, their motion can be affected by other traffic agents, might be under the influence of drugs or alcoholic drinks, and they are hard to see in poor visibility condition.
	Vehicle	Dependent on other vehicles' actions, traffic rules, road geometry, different driving environments, vehicles have multi-modal behaviour, different types of vehicles have different motion properties, drivers might be under the influence of drugs or alcoholic drinks, and TVs might be occluded.
System	*Design*	To achieve a good evaluation metric performance, long PH, real-time inference, low hardware resources, and robustness.
	Evaluation	Works have used different types of datasets, evaluation metrics, observation and prediction horizons, and hardware setups. Therefore, works cannot be directly compared, and the actual progress of pedestrian and vehicle behaviour prediction research cannot be measured.
Resources	*Hardware*	Smaller size GPUs that can process deep learning algorithms in real-time, sensors that enable the IV to perceive 360-degree road view, and affordable hardware to enable all social classes to afford AVs.
	Data	Several existing datasets are not publicly available and are not standardised to enable cross-dataset evaluation and progressive training pipeline techniques.
Uncertainties	*Hardware Failure*	Camera, GPS, IMU, steering wheel, and wheel encoder sensor failure.
	Cyber Attack	Remote hacking, vehicle spoofing, insider threat, and tampering with sensor data.
	Software Failure	Perception module (detection, tracking, image processing, interaction representation, and feature engineering) failure.

Behaviour Prediction System Requirements

An IV behaviour prediction system needs to meet several key requirements to ensure its effectiveness:

- **Good Evaluation Metric Performance**: IV behaviour prediction system is a safety-critical system, therefore it must perform well in terms of evaluation metric performance to prevent traffic collisions. For example, if the system fails to predict that a pedestrian will cross the road, it could lead to a serious collision.
- **Long (PH)**: A system with a long PH can plan and react well in advance, reducing the chances of collisions and improving overall safety.
- **Fast Inference Time**: Given that an IV behaviour prediction system must operate in a real-time, it must have a low inference time and require a low hardware resource.
- **Low Cost**: To make AVs accessible to a wide range of people, the behaviour prediction system should be cost-effective, ensuring that AVs are affordable for all social classes
- **Low Hardware Resource Requirement**: Efficient utilisation of hardware resources is important, as it allows the system to run on hardware with limited capacity.
- **Robustness**: The system should be robust and able to handle various scenarios and conditions on the road, ensuring reliable performance in different situations.

- **Prediction of Various Non-Static Objects**: The system should be capable of predicting the behaviour of different types of non-static objects on the road, including pedestrians, vehicles, animals, and cyclists, to ensure comprehensive safety.

Evaluation metrics, long prediction horizons, and robustness are interrelated. For instance, as the prediction horizon increases, the evaluation metric performance tends to decrease. In addition, as a system becomes more robust, its evaluation metric performance is expected to increase. The major challenges that limit behaviour prediction algorithms from meeting the previously mentioned requirements stem from the fact that an agent's behaviour depends on other agents in the scene, the local and global context, and their final goal. Various approaches have been proposed to address these challenges:

- Social pooling layers [139, 181], Graph representation, GCN, self-attention based social pooling [146], message passing mechanism [145], occupancy maps [135, 163, 169], view frustum social pooling [191], and star-like networks to model interactions between agents [149].

- CNNs to extract agents' appearance, body pose, local context, global context, and to classify intentions [211, 126, 198, 122, 172, 120, 121, 69].

- Attention mechanisms and transformer networks to focus on the most relevant information [201, 128, 123].

- 3D-CNNs and temporal-Densenet to learn short-term temporal information [76, 69, 200, 133].

- LSTMs and GRUs to learn long-term temporal information [128, 131, 202, 130, 132].

- A modified version of the LSTM cell that accepts more than one input sequence set [194].

- CVAE was used to estimate the final goals of the agents to extend the prediction horizon [73, 146, 195, 196].

- Heterogeneous agent behaviour prediction works have been presented to enable the system to predict the behaviour of different non-static object behaviour [218, 214, 215, 216, 217, 219, 220]. However, these works have primarily focused on pedestrians, cyclists, and vehicles, while there are other objects such as animals, disabled individuals, scooters, toys (balls), skate riders, etc.

- Combination of two or more methods to compensate their limitations [207, 211, 122, 76, 126, 127, 125].

- Systems that can predict the behaviour of heterogeneous agents [214, 215, 216, 217, 220].

Inference time, low cost, and low hardware resource requirements are also interrelated. For example, a system that consumes less memory and computational power results in cheaper hardware requirements, making the overall system more

cost-effective. Typically, when a system requires less memory, such as for processing image inputs, the system's overall inference time is expected to be shorter. However, there may be a trade-off between accuracy and inference time. For example, using multiple-feature information can increase the system's accuracy but may lead to longer inference times than a system using a single feature type. The following methods have been proposed to achieve low inference time, low cost, and low hardware resource requirements:

- GCN, which represents interactions between agents effectively without relying on additional information like original images, cropped images, or contextual information [144, 138].
- Dual-LSTM allows the system to learn more information from past trajectories without requiring extra input features [137].
- Fusion of multiple input features (context, interaction, trajectories, and appearance) into an enriched image representation, rather than processing a sequence of images [121].

Behaviour Prediction System Further Work

Despite the techniques presented to meet the specified requirements, work is still to be done from the authors' perspective. For example:

- Most of the works for pedestrians and vehicles were implemented using either a top-view or BEV dataset, which may not be ideal for an IV system. Researchers have only started implementing algorithms using on-board datasets such as PREVENTION, Appolo, JAAD, and PIE in the past five years. Moreover, most of the works that used onboard datasets focused on implementing intention prediction algorithms, and most of the proposed algorithms cannot be directly compared.
- While some works have used the same datasets, evaluation metrics, OH, and prediction horizon, these works were implemented on top-view and BEV datasets. For example, many vehicle trajectory predictions have used the NGSIM dataset with an OH of 3 s, a PH of 5 s, and the MSE evaluation metric. Several pedestrian prediction trajectory algorithms adopted the ETH and UCY dataset, with an OH of 3.2 s, a PH of 4.8 s, and the ADE and FDE evaluation metric. If these datasets were ideal for IV systems, then the best vehicle trajectory prediction algorithms would be GRIP [144], GRIP++ [138], and AI-TP [180]. The best pedestrian trajectory prediction algorithm would be the Bi-Trap algorithm [195].
- There is a lack of research on unusual behaviour exhibited by pedestrians and vehicles. For example, pedestrians might exhibit unusual behaviour when under the influence of toxic substances, involved in fights, or disoriented. Similarly, vehicles may display unusual behaviour when the driver is under the influence of toxic substances, distracted with their personal belongings, or if the vehicle is an emergency vehicle, garbage truck, road sweeper, carrying an abnormal load, or experiencing mechanical malfunctioning.

- There is limited research on decreasing inference time, and more emphasis should be placed on addressing this demand.
- Standardising datasets would enable cross-dataset evaluation and the development of progressive training pipeline techniques.
- Introducing universal metrics would allow for direct comparisons of algorithm performance.
- When considering a complete pipeline system (detection, tracking and behaviour prediction), it is necessary to account for perception uncertainties due to sensor noise, fuzzy features, or unknown inputs[229]. Since a limited number of works have implemented a complete pipeline system, more works considering the entire pipeline process are recommended to investigate the effect of possible noise.

Based on the literature review, the following suggestions are given to further improve and accelerate the development of the IV Behaviour Prediction System:

- Encourage more research works to adopt on-board view datasets for predicting both pedestrian and vehicle behaviour, including intention and trajectories.
- Standardise existing dataset to enable cross-dataset evaluation and progressive training pipeline techniques.
- Choose or create a standard evaluation metric to enable direct comparison among algorithms.
- Develop datasets with instances of abnormal pedestrian and vehicle behaviours to enable research on the recognition and prediction of abnormal pedestrian and vehicle behaviour.
- Implement behaviour prediction algorithms on resource-constrained hardware, such as Jetson Orin, and Jetson Xavier GPUs, which are low-cost, small in size, lightweight, and consume low power.
- Investigate more methods to select the target object and the objects that directly interact with the target object.

The general object detection problem exemplifies the importance of having a large dataset and standard evaluation metrics. The field has achieved an acceptable level of maturity because researchers have access to publicly available large image benchmark datasets, such as the ImageNet [230] and COCO [231]. These datasets enabled the authors to directly compare their detection algorithm performance and measure object detection research's advancement.

2.2.7 Conclusion

IV systems must not only detect pedestrians and vehicles but also predict their behaviour to avoid or mitigate collisions. Therefore, the purpose of this literature review, was to survey the most relevant pedestrian and vehicle behaviour prediction algorithms to identify the requirements for a behaviour prediction algorithm, the

challenges associated with predicting pedestrian and vehicle behaviour, whether current techniques have met these requirements, and what steps are needed to enable AVs to predict pedestrian and vehicle behaviours. In conclusion, the review shows that:

- An IV behaviour prediction system must have a good evaluation metric performance, long prediction horizon, and fast inference time, must be cost-effective and robust, require minimal hardware resources, and must predict various types of non-static objects on the road.
- The main challenges in predicting the behaviour of traffic agents involve modelling their interactions, establishing a relationship between the agents and the scene, and achieving a balance between good evaluation metric performance and low inference times.
- Current techniques do not fully meet these requirements for several reasons:
  - when predicting for long-term horizons, evaluation metric performance significantly decreases;
  - while top-view and BEV datasets are commonly used in the literature, there are limited works that adopted on-board datasets, which are more suitable for AVs;
  - on-board datasets usually only use a single forward-facing camera, limiting the behaviour prediction system to consider only agents ahead, whereas considering agents around the EVs using multiple cameras is essential [232];
  - more investigation is required to develop models that can predict intention and trajectory simultaneously; although some authors [144, 138] claimed that their system has achieved real-time inference times, they have used top-view cameras, whereas systems that use on-board sensors may require more processing time;
  - there are no works that consider abnormal behaviour exhibited by traffic agents.
- Most of the reviewed works have not considered the complete pipeline behaviour prediction process, which consists of detection, classification, and tracking. More research should focus on the complete pipeline process to assess the performance of each stage and its impact on the final prediction results.

2.3 Traffic Hazard Events Detection and Recognition

A traffic hazard event can be defined as a traffic situation that could cause harm to others. Authors Xiao et al. [61] have categorised traffic hazard events into three stages: potential event, developing event, and materialised event. Each stage describes a progression of traffic-related risks.

A potential event can be defined as a traffic scenario wherein no immediate collision threats are present. Nonetheless, the latent possibility exists for a situation to evolve, triggering a collision threat. For example, a road scenario where the EV is on the main road, and a cyclist is approaching from the intersecting road on the right. While the cyclist remains a considerable distance from the intersection, no current threats exist. However, a threat could evolve if the cyclist does not decrease their speed.

Developing events can be defined as a traffic scenario where an identified collision threat is imminent and corrective measures, such as braking or steering, are required to avoid a collision. For example, if the cyclist from the above example fails to reduce its speed, the EV has to intervene by braking and steering to avoid a collision.

Materialised Events can be defined as the stage where a collision has happened. In the above examples, the cyclist and the EV collided.

This categorisation framework provides a structured approach for evaluating and responding to traffic hazards on the road. Therefore, IVs such as autonomous vehicles (AVs) and advanced driver assistance systems (ADAS) must learn to recognise and detect a traffic hazard event at the potential event stage, respond to developing situations, and critically take pre-emptive action to avoid collisions before they materialise.

Detecting and recognising traffic hazard events is challenging for the following:

- Traffic environments are complex and dynamic: several types of traffic scenes exist, such as urban, country roads, and highways, each with distinct cues. For example, urban scenes have high traffic densities, complex intersections, complex traffic sign roads and traffic lights. Highways are less complex, but vehicles are driving at high speeds. Country roads lack traffic signalisation, such as road marks. In addition, weather and illumination conditions can change quickly.

- Traffic agents diversity: There are numerous types of traffic agents, each with particular appearance and motion features. Traffic agents such as pedestrians and animals are very dynamic, and their behaviour can be unpredictable.

- Perception limitation: An IV system would require a perceptions module to detect, track, and recognise traffic agents, traffic objects, and weather conditions to assess if there is a potential traffic hazard event. Perception modules have uncertainties due to sensor limitations, environmental conditions, and occlusions.

- Numerous traffic hazard events exist: A comprehensive dataset is required to represent the various types of traffic hazard events, which can be time-consuming and expensive. Furthermore, the development of algorithms gets more challenging as the number of classes to classify increases.

- Resources requirements: hazard detection and recognition must meet real-time requirements since the decision module uses the information acquired to avoid or mitigate a potential collision. Also, it must be computationally efficient since automotive hardware usually has a small footprint.

The following sections discuss the techniques and available datasets designed to train and validate algorithms for detecting and recognising traffic hazard events.

2.3.1 Traffic Hazard Detection and Recognition Algorithms

The following techniques have been used to recognise and detect traffic hazard events: Bayesian Network, Graph Convolutional Network, CNN, Unsupervised machine learning, RNN, and multiple models.

Bayesian Network: Russo et al. [233] studied the risky probability of a rear-end collision by implementing a Bayesian Network (BN). The BN takes as input the EV reaction time, the maximum available deceleration, and the braking intention of the vehicle in front, which is available through V2V communication. Although the Bayesian network deals well with uncertainties and captures the relationships between variables, it may oversimplify interactions in complex traffic scenes.

GCN: A Spatio Temporal Action Graph (STAG) network was performed by [234] to predict near-collision events. First, ResNet and RPN networks are used to generate object-bounding boxes. The feature of the generated bounding boxes and features of every possible pair of the generated bounding boxes are used as input to the STAG network to predict near-collision events. GCN has the advantage of learning spatial and temporal dependencies and captures the complex interactions among traffic agents well due to its graph representation structure. However, the system can become very complex to develop, requiring more computational resources.

CNN: The near-miss incident dataset (NIDB) was used to train and evaluate a two-stream CNN by [235] to recognise near-miss incidents. One of the two-stream networks learns spatial features from RGB images, while the other learns temporal features from semantic flow-generated images. Kim et al. Kim et al. [236] used CNN to classify if the detected objects are dangerous or not using the YouTubeCrash and the GTACrash datasets. Although two-stream CNN was used to consider spatial and temporal information, CNN's primary excellency is in learning spatial features. In addition, two-stream CNN might become computationally expensive.

Unsupervised: Yao et al. [237] proposed an unsupervised machine learning algorithm to detect traffic accidents. The unsupervised algorithm was first trained on normal driving behaviour samples, and then the trained algorithm was used to predict future trajectories of the traffic agent. The algorithm decides if an abnormal event occurs when considerable differences exist between the accuracy of the predicted bounding boxes and the past observed bounding boxes, the accuracy between the predicted bounding box mask and the observed bounding box mask, and the consistency values of the predicted bounding boxes. The unsupervised technique does not require labelled data; however, interpreting the results might be more challenging.

RNN: CNN LSTM [78] presented a new multi-modal LSTM network to predict actions in six types of traffic scenarios included in the Virtual Environment for Action Analysis (VIENA) dataset. The considered feature modalities were appearance from RGB images, motion from optical flow images, and dynamics measurements. Authors Yurtsever et al. [238] adopted the CNN LSTM network to classify dangerous lane change behaviour. A dynamic spatial-attention RNN was implemented by [239] to predict traffic accidents from dash-cam recorded videos. The LSTM with attention mechanism was trained and evaluated using the Anticipating Accidents in Dashcam (AAD) dataset. RNNs are a good candidate for learning temporal dependencies and dealing with sequential events. However, their internal processing requires a fixed-length vector, limiting the number of interacting traffic agents they can manage. Additionally, as the number of traffic agents increases, the system im-

plementation becomes more complex and demands more significant computational resources.

Multiple Models: Bao, Yu, and Kong [240] proposed a uncertainty-based **traffic accident prediction** model considering spatio-temporal relationship. A GCN combined with an RNN was used to learn traffic agents' relationships from a series of graph-embedded representations of the present traffic agents. In addition, a Bayesian Neural Network was used to predict the accident score. Malawade et al. [241] predicted collision by proposing a scene graph representation of the traffic scene and feeding it to a GCN and LSTM. The application of multiple models has the advantage of combining the strengths of each model to improve prediction performance. However, integrating multiple models can be complex, and since some models require different types of input presentation, advanced data fusion methods may be required.

2.3.2 Traffic Hazard Dataset

Traffic hazard datasets are developed to train and evaluate algorithms to detect and predict traffic road hazards. The available traffic hazard datasets, as listed in Table 2.12, can be categorised as real or synthetic data. Real data consists of data directly acquired from front camera sensors installed in EVs or indirectly gathered from dashcam videos from various YouTube channels. On the other hand, synthetic data is generated using the Grand Theft Auto V (GTA V) game. These datasets either contain samples that led to a collision or near-misses, in other words, led to a developing or materialised hazard stage.

Table 2.12: Information about the various traffic hazard datasets.

Dataset	Dataset Source	Domain	Dataset Type	Event Type	Num. of Accidents	Num. of N. Misses	Hazard Stage
NIDB [235]	Vehicle Dashcam	Private	Real	near-misses	n/a	4595	developing
A3D [237]	YouTube (dashcam)	Public	Real	collision	1500	n/a	materialised
DoTa [242]	YouTube (dashcam)	Public	Real	collision	4677	n/a	materialised
AAD [239]	YouTube (dashcam)	Public	Real	collision	678	n/a	materialised
CTA [243]	YouTube (dashcam)	Public	Real	collision	1935	n/a	materialised
CDD [240]	YouTube (dashcam)	Public	Real	collision	1500	n/a	materialised
GTACrash [236]	GTA 5 (game)	Public	Synthetic	collision	7720	n/a	materialised
VIENA [78]	GTA 5 (game)	Public	Synthetic	collision + 5 types of actions	∼1200	n/a	materialised
Collision [161]	Vehicle Dashcam	Public	Real	collision + near-misses	743	60	materialised/ developing

The Near-miss Incident Database (NIDB) was designed to study the recognition and detection of traffic near-miss incidents [235]. The dataset was captured using camera sensors installed in more than 100 taxis, and the records were made over ten years. The recording yielded 4595 near-misses video samples of 10 to 5 seconds each. The dataset contains annotations about the high and low danger levels for the following traffic objects: bicycle, pedestrian, and vehicle. The high-level danger can be considered a developing hazard event, and the low-level danger can be regarded as a potential hazard event. Unfortunately, the dataset is not publicly available due to copyright reasons.

The Collision dataset comprises 743 collisions and 60 near-collision video samples

collected from connected dashcam sensors [161]. The dataset classifies the samples as "negative", where there are no accidents, and "positive", where there is an accident.

The AnAn Accident Detection (A3D) and Detection of Traffic Anomaly (DOTA) datasets are composed of recorded dashboard videos extracted from a YouTube channel containing collision events [237, 242]. Each video includes information on when and where a collision event happened and what caused it to happen. The collision events are categorised as follows:

- Collision with a parked vehicle.
- Collision with a vehicle moving ahead or waiting.
- Collision with a vehicle moving horizontally in the same direction.
- Collision with a vehicle approaching from the opposite direction.
- Collision with a vehicle making a turn or crossing a road.
- Collision with a pedestrian.
- Collision with an obstacle on the road.
- Vehicle is out of control and leaving the road.
- Unknown event.

The Anticipating Accidents in Dashcam (AAD), Causality in Traffic Accident (CTA), and the Car Crash Dataset (CDD) are other datasets composed of recorded dashcam videos extracted from YouTube channels [239, 243, 240]. These datasets only contain two classes: the positive samples, where there is an accident event, and the negative samples, where there are no accidents.

The GTACrash and VIENA datasets are synthetic datasets generated and collected from the Grand Theft Auto V video game [236, 78]. The GTACrash dataset contains two types of samples: accident and non-accident events. VIENA dataset aims to predict actions in traffic scenarios. There are 25 classes spread across six types of scenarios, including driver manoeuvre, accidents, traffic rule, pedestrian intention, front car intention, and manoeuvres performed by heavy vehicles.

The previously cited traffic hazard dataset only considers hazards that involve motor vehicles or non-motor vehicle actors. Static and dynamic environmental cues, such as obstacles, objects on the road, road type, adverse weather, and illumination, should also be considered. As well as regulatory traffic laws such as rule breaks, traffic lights, and traffic signs [61].

2.3.3 Conclusions

This section delves into the challenges, current techniques, and datasets used to detect and recognise traffic hazard events. It concludes that while there are numerous techniques and datasets, they primarily focus on classifying the presence of traffic hazard events. However, for an IV system to comprehensively understand a traffic situation and make informed decisions, it is crucial that potential traffic hazard events be recognised. Moreover, existing studies tend to focus on detecting and

recognising traffic hazard events in their developed or materialised stages, whereas recognising them in their potential stages is more pertinent for the IV system.

This chapter's findings underscore the necessity of a novel dataset that includes a wider range of traffic hazard event classes in their potential event stage. Such a dataset would significantly enhance the AI system's ability to comprehend the intricacies of traffic road scenes and make decisions to prevent collisions resulting from other traffic agents' behaviours.

2.4 Vehicles' Rear Light Detection and Status Recognition

Rear lights and their status are important visual cues that human drivers use to predict the intent behaviour of the surrounding vehicles and decide their actions to navigate complex traffic scenes. Therefore, it is crucial that IVs acquire this human driver skill. This section discusses the challenges of detecting and recognising them and what inputs, methodologies, and outputs have been used in the literature.

2.4.1 Challenges

Detection and recognition of rear lights are challenging tasks for the following reasons:

- Rear lights are small and can be easily occluded by other road objects.
- Rear light design, location, light intensity, and blinking frequency can differ depending on the vehicle's make and model.
- Vehicles can be in different angles in an image.
- One image can have rear lights from multiple vehicles, making it harder to pair rear lights.
- Images with rear lights are affected by poor illumination and weather conditions.
- Indicator lights status is dynamic.
- It is hard to discern if a brake or a rear light is on. Although new cars have a mid brake light to help discern between brake and rear light activity, they are too small, some old vehicles do not have it, and some may not be working.

The primary sensors used to detect and recognise rear lights are mono [244], stereo [245], and high dynamic cameras [246]. However, most of the reviewed works are based on monocular camera sensors as they are cheaper and have been adopted by most car manufacturers.

The following discussion regarding the inputs, methodologies, and outputs to detect rear lights and recognise their status is based on 24 reviewed works from 2012 to 2022. Since detection and recognition are two different tasks, they will be discussed separately.

2.4.2 Rear Lights Detection

Input

The input to a rear light detection system could be the original image captured by the camera sensors, which might contain more than one vehicle, or the cropped portion of the original image containing only one vehicle.

Methodologies

Two main methods are used to detect rear lights: traditional or Deep Learning (DL) techniques.

Traditional techniques include segmentation, symmetry, morphology, colour threshold, haar-like features, or colour clustering. Li, Cai, and Tang [247] used segmentation and geometry to localise rear lights. Casares, Almagambetov, and Velipasalar [82] used colour and morphology to identify the rear light region and 2-D symmetry to perform rear light pairing. Fröhlich, Enzweiler, and Franke [248] adopted the Absolute image difference and Gaussian mixture model to detect light spots from the rear lights. A new lamp response technique was presented by [249], where they measure red pixel intensity to detect rear lights. Cui, Yang, and Tsai [83] used red pixel clustering and convex Hull to detect rear lights. Grey level image, Haar-like features, geometry and colour properties were used by [250] to detect vehicles and their respective rear lights. Fully Connect Network (FCN), colour transformation, morphology and binarisation were used by [251] to detect rear lights. Wang et al. [252] adopted colour space conversion and location correlation principle to segment tail lights. Gradient histogram, colour, and distance were used by [253] to detect rear lights. Nava, Panzani, and Savaresi [254] used L*a*b colour space and geometry to identify rear lights. Yan et al. [79] detected vehicle rear lights using image segmentation, morphology, and contour extraction. Traditional methods are usually computationally efficient. However, their performance decreases when applied in complex scenarios with poor illumination, occlusion, noise, and objects too far away.

Currently, most authors have adopted DL techniques to detect rear lights because they have outperformed traditional techniques in general object detection and recognition tasks. The DL techniques used to detect rear lights were the CNN and Recurrent Rolling Network (RCC). Authors [255] detected rear lights using Fast R-CNN and Nakagami images. Li et al. [256] adopted YOLOv3-tiny, Spatial Pyramid, and focal loss. Liu et al. [257] used ResCat, which is a combination of ResNet and DenseNet. Chang and Zhang [258] used CSPRexNext-40, Bi-Feature Pyramid Network, and integrated high-level semantic mask. Authors [259, 260] used YOLOv2 and RRC. DL techniques have outperformed traditional ones. However, they require more computational power and a substantial amount of data.

Output

The rear lights have their respective bounding boxes for each object in the original image or for the single vehicle in the cropped portion of the TV.

Evaluation Metrics

The evaluation metrics used in the literature were accuracy, true positive, mean average precision (mAP), average precision (AP), precision, and recall. Accuracy was the most used metric, followed by precision and recall.

2.4.3 Rear Lights Status Recognition

Input

It was noticed in the reviewed works that two pipelines were used to recognise the status of the rear lights. Most of the works would first detect the vehicle, then the rear lights, and finally recognise the rear light status [247, 82, 248, 249, 83, 250, 251, 255, 252, 253, 254, 256, 79, 257, 259, 260]. Some of the recent works would use the bounding box information of the detected vehicle to crop the image where the detected vehicle is and subsequently recognise the rear light status by appearance-based approach [84, 246, 81, 244]. The appearance-based approach can eliminate the stage of detecting the rear lights, which usually involves hand-crafted feature extraction and requires more computational time. Furthermore, it enables the vehicle detection network to share its feature vector with the rear light status recognition network, automatically considers the brake mid-light, and is more robust against occlusions [84]. In conclusion, the inputs to a rear lights recognition status algorithm can be the cropped image, where the detected vehicle is with/without the bounding boxes where the rear lights are located. In addition, the input can be a single frame or a sequence of frames. Sequences of frames are used if temporal dependencies are required.

Methodologies

Rear light recognition status has also been implemented using traditional or DL techniques, and it has been done using either a single image or a sequence of images as input. Single image methods have the advantage of being computationally affordable but lack temporal information, such as the action of the left or right indicator blinking.

Traditional techniques include adaptive threshold, High pass Mask, Sparse representation, mean channel values and threshold, Nakagami Histogram, X direction projections of the Nakagami image, histogram distribution, AdaBoost, SVM, and random forest classifiers. Authors [247] used adaptive threshold, and [250] used mean channel values and threshold to recognise rear light status. Comparison between adjacent images was used by [261], high pass mask was used by [249], and sparse representation was implemented by [83]. Histogram distribution was used by [252] and [253]. Authors [248, 254, 262] used AdaBoost, SVM, and Random Forest classifiers, respectively.

Deep learning techniques include CNN, RNNs, or both. Most works have treated rear light status recognition as a sequence problem. For example, authors [81] combined CNN and LSTM to recognise rear lights' status using a sequence of images. The CNN was responsible for extracting spatial features, and the LSTM was responsible for extracting temporal dependencies. Lee et al. [244] proposed a CNN with spatial attention and LSTM with a TA mechanism. A Siamese CNN and GRU

network was implemented by [257]. Kitchat et al. [263] proposed a system that uses a sequence of images to classify rear light status. The VGG16 network was adopted for feature extraction, and WaveNet [264] was used to capture temporal features. Song et al. [80] proposed an action-state joint learning-based rear light recognition system. They used YOLO and DeepSort algorithms to detect and track the vehicles, Gaussian Mask Segmentation to extract brightness features from the vehicle rear light, then CNN LSTM to classify the rear light actions, and finally, a linear Conditional Random Field (CRF) to classify the twilight status.

Only a few works studied rear light status recognition as a single-frame problem. For example, Authors [84] and [246] used AlexNet to classify the rear light status based on using as input a single image. Apart from the reason that sequence techniques consider temporal information, most works adopted sequence techniques because of the availability of datasets.

Output

The rear lights can have the following status: all rear lights are "OFF" (OOO), the right indicator is "ON" (OOR), the Left indicator is "ON" (OLO), the brake lights are "ON" (BOO), right indicator and brake lights are "ON" simultaneously (BOR), left indicator and brake lights are "ON" simultaneously (BLO), hazard lights are on, where both left and right indicator lights are "ON" simultaneously (OLR), hazard and brake lights are "ON" simultaneously (BLR), reverse light is "ON". In the literature, most works investigate the recognition of the brake lights and the status of the right and left indicators. Only a few works have tried to recognise the other categories, and no one has investigated the recognition status of the reversing lights.

Dataset

Most of the proposed works have created their own datasets to investigate rear light recognition status. According to the authors [84, 246, 257, 256, 250], there are no public datasets to train and validate rear lights recognition status.

To the authors' knowledge, there are only three publicly available datasets that provide information about some of the rear lights' status: the PREVENTION [71], the Laboratory for Intelligent and Safe Automobiles-Night (LISA-Night) Low-Density, and the UC Merced [81] vehicle rear signal datasets. The PREVENTION dataset is not specific to the vehicle's rear signal recognition; it only informs if the rear light indicators are "ON/OFF" and does not specify whether it is the right or the left one. In addition, the dataset does not provide cropped images of the TV or information about other rear light statuses (e.g., brake light status). The LISA-Night Low-Density dataset was only recorded at night and provided rear light activity for turning and braking. The UC Merced dataset does not specify the rear light indicator status for each one of the frames; it instead provides labels for a sequence of frames. Hence, it is specific for systems that recognise the rear light status in videos. The UC Merced dataset has all the rear light statuses cited above, apart from the reversing light status. It has 649 sequences and 63637 frames and was recorded under various road conditions.

An essential experimental stage of the deep learning system is splitting the dataset samples. Although the cited works mention the percentage a train and a validation set have from the complete dataset, they do not detail how the dataset

Figure 2.9: Frames of the same video samples exhibit significant similarity. For instance, frames 3 and 4, as well as frames 5 and 6 from video sequence 1, are very similar to each other. Likewise, frames 3 and 4, and frames 5 and 6 from video sequence 2, also display considerable similarity to each other

was split. To the author's knowledge, the dataset split could be reached by applying an image, sequence, or video data split technique.

Figure 2.10: The train and the validation set have similar frames belonging to the same video sample. For example, frames 3 and 4 from video sequence 1 are in the train and the validation set, respectively.

For example, a rear light dataset comprises frames from different video samples recorded by monocular cameras. As depicted in Figure 2.9, a video sample can have two or more consecutive frames very similar. Therefore, as illustrated in Figure 2.10, during an image dataset split, one of these similar frames can be placed at the train set and the other at the test set, which will facilitate the algorithm in predicting the class that the frame belongs to. Although this might yield a high training and validation accuracy, the algorithm might not generalise well when applied to unseen video samples. This phenomenon also applies to sequence splitting since some of the sequences of frames belonging to the same video samples can be placed in the train and validation set. On the other hand, as illustrated in Figure 2.11, when applying

the video-splitting technique, there is less chance of placing very similar images in both the train and the validation set.

Figure 2.11: The train and the validation set have less chance of having similar images.

2.4.4 Conclusions

This section investigated the application of current strategies to detect and recognise the light status of surrounding vehicles. It highlighted that although the status of a vehicle's rear light is crucial for providing information about other vehicles' intentions, there is limited research on this topic. This observation was also noted by Song [80]. The lack of available datasets appears to be a significant contributing factor; only three publicly available datasets exist, each designed for a different purpose and not conducive to transfer learning techniques or dataset merging.

The findings of this chapter underscore the need for more research focused on using single images for rear light status recognition, which would enhance computational efficiency and be used as one of the modules in a complete IV system. Moreover, studies using state-of-the-art feature extraction algorithms such as Vision Transformer (ViT) are recommended, as the currently adopted algorithms are outdated.

These insights contribute to the broader objective of improving IV systems by enhancing their ability to understand and react to the intentions of surrounding vehicles. Future research should focus on developing and sharing more detailed datasets and exploring advanced feature extraction techniques.

2.5 Knowledge Gap

Despite substantial progress in IV systems, behaviour recognition, and traffic hazard detection, several critical research gaps persist:

2.5.1 Discrete Intention Behaviour Recognition and Prediction

- Existing research primarily addresses general behaviour recognition, with limited focus on the prediction of discrete intention behaviours (e.g., turning, stopping, merging) of traffic agents.
- Most studies concentrate on either pedestrians or vehicles, often overlooking other essential traffic participants such as cyclists and traffic control elements (e.g., traffic lights).
- Although vehicles exhibit a wide range of behaviours (e.g., *cutting in, reversing, turning*), the majority of research is narrowly focused on lane-changing behaviour.

2.5.2 Data Limitations

- There is a notable lack of publicly available datasets specifically designed for training and evaluating models on discrete intention behaviour prediction.
- Existing datasets often fail to capture the complexity and heterogeneity of urban traffic environments, limiting their applicability to real-world scenarios.

2.5.3 Algorithmic Gaps

- Current algorithms do not explicitly address the recognition of discrete intention behaviours across diverse traffic agents.
- The visibility of target objects to the EV is frequently overlooked. For example, partial visibility (e.g., viewing only the right side of a vehicle) may indicate an imminent crossing or intersection entry, yet this contextual cue is underutilised.

2.5.4 Traffic Communication and Hazard Recognition

- While vehicles communicate intentions through light signals (e.g., indicators, brake lights), these cues are often implicitly or insufficiently integrated into behaviour recognition models [265].
- Most datasets capture hazard events only after they have materialised, lacking data on the progression and early indicators of such events.
- There is a lack of comprehensive studies categorising and recognising various types of traffic hazards in real-world conditions.

2.5.5 Integrated Behaviour Recognition Pipelines

- Few studies have proposed a holistic behaviour recognition pipeline that integrates detection, tracking, and recognition components.
- The issue of error propagation across pipeline stages remains underexplored, despite its potential to significantly affect prediction accuracy.

Chapter 3

Novel Traffic Hazard Dataset

Datasets for studying discrete traffic agent behaviour and potential traffic hazard event recognition are limited. Existing datasets typically focus on either pedestrian or vehicle behaviour and often cover only a narrow range of behavioural categories. Datasets that include a broader spectrum of behaviours tend to focus on the actions of the EV's driver rather than those of surrounding traffic agents. Moreover, most traffic hazard datasets are designed for binary hazard detection (e.g., collision or no collision) rather than for classifying different traffic behaviours or hazard types. Some datasets, such as VIENA, do consider a wider variety of traffic agent behaviours but are synthetically generated. Additionally, these datasets often emphasise near-miss or collision events, focusing on hazards in their developing or materialised stages.

This chapter introduces and analyses a novel dataset designed to enhance the understanding of potential traffic hazard behaviours in complex traffic environments. Compiled from multiple existing sources, the dataset provides diverse traffic scenarios and includes sixteen potential hazard classes. Each sample contains detailed detection information for all objects in the scene, tracking data for each target object, timestamps marking the start and end of specific hazard events, rear light status for each vehicle, and the side of each target object visible to the EV.

The chapter details the dataset's composition, offering an overview of the data collection and annotation methods. It explains the dataset format, including its directory structure and data storage. Additionally, the dataset's statistics are presented, outlining the types of classes, the number of samples per class, and a comparison with related datasets.

The chapter is organised as follows: Section 3.2 describes the dataset collection and annotation processes. Section 3.3 details the dataset's directory structure and storage format. Section 3.5 outlines potential applications for the dataset. Finally, Section 3.6 presents statistical information and compares the novel dataset with existing ones.

3.1 Novelty and Contributions of the Traffic Hazard Dataset

The dataset presented in this study introduces several novel aspects that distinguish it from existing **discrete** traffic behaviour and hazard recognition datasets:

1. **Multi-Agent Diversity**: Unlike traditional datasets that often focus solely on vehicles or pedestrians, this dataset captures behaviours from a wide range of traffic agents, including trucks, riders, individual and group pedestrians, animals, traffic signs, and temporary road objects such as cones and barriers.

2. **Granular Behaviour Labels**: It incorporates over 15 types of discrete behaviours, including nuanced traffic interactions such as overtaking, lane changes, and object crossing—many of which are underrepresented or absent in prior work.

3. **Multi-View Annotations**: The dataset provides annotations for both rear light status and the visible side of target objects, each with single-image granularity, enabling real-time perception and classification tasks.

4. **Potential Hazard Emphasis**: The dataset emphasises potential hazard states rather than active or ongoing incidents, addressing a gap in predictive hazard recognition and early behaviour anticipation.

5. **Real and Synthetic Fusion**: By combining data from both real-world sources (including PREVENTION and JAAD) and synthetic environments, the dataset captures a broad spectrum of scenarios, enhancing model generalisability and robustness.

6. **Environmental and Geographical Variety**: The dataset includes diverse scenarios from urban and highway environments across multiple countries, helping to mitigate geographical and environmental biases common in traffic datasets.

These characteristics collectively make the proposed dataset a comprehensive and novel resource for modelling complex traffic scenarios. It supports the development of intelligent systems capable of understanding and predicting traffic agent behaviour in a more holistic, proactive, and context-aware manner.

3.2 Dataset Collection and Annotation

Collecting datasets from real-world environments is expensive, time-consuming, potentially hazardous, and requires authorisation from both public and private sectors. Synthetically generating a dataset can be a viable alternative; however, synthetic data often fails to accurately capture the complexity and diversity of real-world scenarios and can also become resource-intensive. For these reasons, the novel dataset presented in this study was compiled from several publicly available existing datasets, incorporating both real-world and synthetic sources.

The datasets used include KITTI, JAAD, Oxford RobotCar Dataset, DVSA hazard perception clips, and PREVENTION. Details for each dataset are provided in Table 3.1. Note that data governance and ownership were taken into consideration, and it was confirmed that the authors of the datasets used permit sharing and adaptation under the Attribution-NonCommercial-ShareAlike 4.0 International License [266]. Additionally, all datasets used in this thesis are properly referenced. Clips containing DVSA hazard perception material were used to generate the results

Table 3.1: Information for the datasets used to generate the potential traffic hazard dataset.

Dataset	**Resolution**	**FPS**	**Format**	**N. of Extracted Samples**	**Original Purpose**	**Data** Acquisition
JAAD	1920x1080	30	PNG	12	Pedestrian Crossing	Real-world
PREVENTION	1920x600	10	MP4	177	Vehicle Lane Change	Real-world
DVSA	1920x1080	25	MP4	38	Traffic Hazard	Synthetic
KITTI	1242x375	10	PNG	7	Detection and Tracking	Real-world
oxfordRobotDataset	1280x960	16	MP4	412	SLAM	Real-world

presented; however, to comply with the terms and conditions set by the DVSA, any samples containing DVSA hazard perception clips will **not** be included in the dataset when it is made publicly available for further research.

In line with DVSA's licensing terms, I include the following required statement:

> *"The Driver and Vehicle Standards Agency (DVSA) has given permission for the reproduction of Crown copyright material. DVSA does not accept responsibility for the accuracy of the reproduction."*

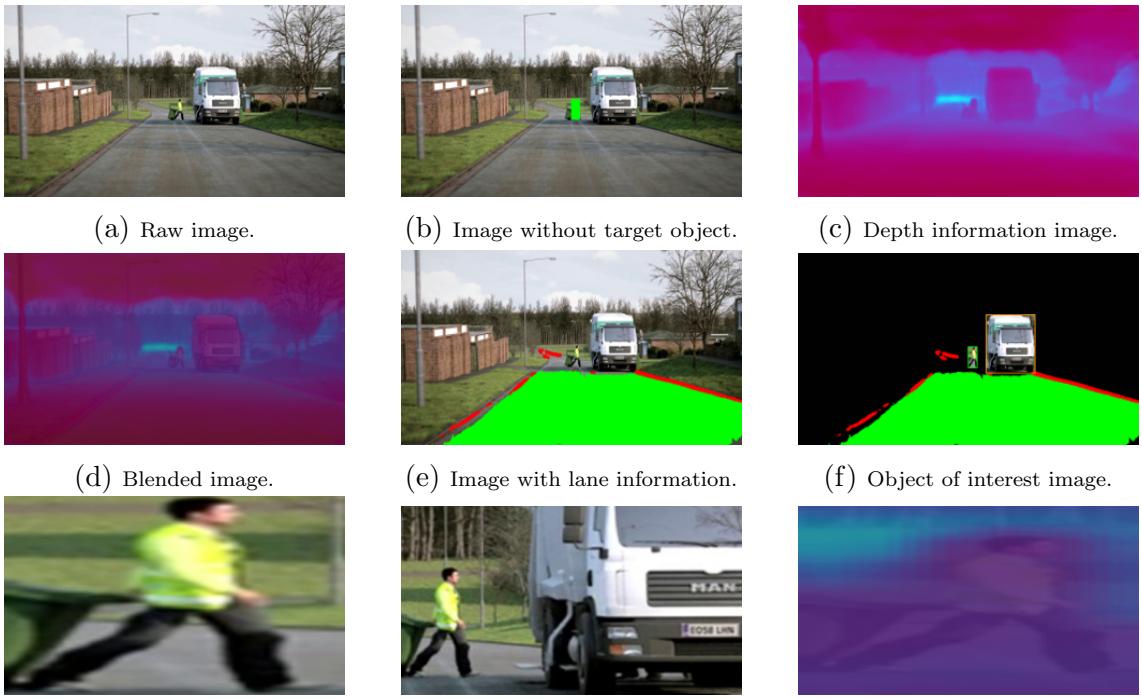
Using multiple publicly available datasets offers the advantage of combining real-world and synthetic data, covering both highway and urban environments, different driving sides, and various purposes. However, these datasets differ in image resolution, frame rate, format, and camera calibration properties. The following procedure was followed to construct the novel dataset from the aforementioned sources:

- Each dataset was downloaded and converted into frames if originally in video format. Due to varying frame rates (FPS), normalisation was necessary. A standard FPS of 10 was chosen, as it was the lowest among the datasets. Normalisation was implemented using the modulus operator.

- Traffic behaviour samples prone to potential traffic hazard events were identified from the normalised datasets. For each identified event, the following information was recorded in a CSV file: dataset name, starting frame, ending frame, hazard type, hazard description, and the directory containing the original frames.

- The recorded information was used to generate video samples for each identified traffic hazard event. Each sample included 20 frames before the starting frame and 10 frames after the ending frame, allowing algorithms to observe up to 2 seconds before and 1 second after the event. Sample durations varied depending on the behaviour type and driver response. All frames were resized to 1280×600 pixels to ensure consistency across datasets.

- Although current detection and tracking algorithms are reasonably accurate, they still fail to detect or track some road objects. Accurate recognition of a target traffic agent's behaviour requires precise detection and tracking. Therefore, the target agent performing the behaviour was manually labelled using the YOLO_label software [267]. The YOLOv8 algorithm was then used to detect and track the remaining objects. To prevent duplicate detection of the target agent, it was extracted from the original image by replacing its pixels with green, as shown in Figure 3.1b.



(a) Raw image. (b) Image without target object. (c) Depth information image.

(d) Blended image. (e) Image with lane information. (f) Object of interest image.

(g) Local context image. (h) Motion-based image. (i) Local-context blended image.

Figure 3.1: Different types of input images generated from the raw image.

- The visible side of the target object from the ego vehicle's perspective was manually labelled.
- The rear light status of each target vehicle (TV) was manually annotated.
- Various image representations were generated from the original images, including:
  - Images without the target object to capture global context as illustrated in 3.1b.
  - Monocular depth estimation images using the Iterative Elastic Bins (IEB) algorithm [268] to capture motion and 3D structure as illustrated in 3.1c.
  - Blended images combining RGB and depth information using OpenCV's `cv2.addWeighted()` function as illustrated in 3.1d.
  - Lane detection images using the YOLOP algorithm [269] to define EV boundaries as illustrated in 3.1e.
  - Object of Interest (OOI) images retaining only the pixels corresponding to the target object, surrounding objects, and lane information as illustrated in 3.1f.
  - Bounding box images to capture local context as illustrated in 3.1g.
  - Local context motion-based images, cropped using both bounding box and motion direction to align with the object's movement as illustrated in 3.1h.

```
Potential Traffic Hazard Dataset
└── frame_sequences
    ├── 000
    │   ├── det/predict/labels/
    │   │   ├── 0002.txt
    │   │   └── ...
    │   ├── img_depth_info
    │   │   ├── 0002.png
    │   │   └── ...
    │   ├── img_global_OOI_context
    │   │   ├── 0002.png
    │   │   └── ...
    │   ├── img_local_context
    │   │   ├── 0002.png
    │   │   └── ...
    │   ├── img_local_depth_context
    │   │   ├── 0002.png
    │   │   └── ...
    │   ├── img_no_target_object
    │   │   ├── 0002.png
    │   │   └── ...
    │   ├── img_original
    │   │   ├── 0002.png
    │   │   └── ...
    │   ├── img_overlayed
    │   │   ├── 0002.png
    │   │   └── ...
    │   ├── img_with_lane_info
    │   │   ├── 0002.png
    │   │   └── ...
    │   ├── tracking
    │   │   ├── surrouding_objects_tracking.csv
    │   │   ├── target_and_surrounding_objects_
    │   │   │   tracking.csv
    │   │   └── target_object_tracking.csv
    │   └── object_position_labels.csv
    ├── 001
    │   └── ...
    │       └── ...
    ├── train
    │   └── recognition_dataset_split.csv
    ├── test
    │   └── recognition_dataset_split.csv
    └── all_roadHazardDataset_videos.csv
```

Figure 3.2: Potential Traffic Hazard Dataset directory tree diagram.

3.2.1 Annotation Process and Ground Truth Validation

The author performed all data labelling to ensure consistency and domain-specific accuracy. Although a single annotator carried out the annotation process, the following measures were implemented to maintain the reliability of the ground truth:

- **Annotation Guidelines**: A detailed annotation protocol was established prior to labelling. This protocol defined class boundaries, provided visual examples, and outlined procedures for handling edge cases to minimise subjective interpretation and ensure consistent annotation logic.
- **Iterative Refinement**: The annotation process followed an iterative review strategy. Initial annotations were frequently revisited—particularly in the early stages—to refine labelling definitions and resolve ambiguities. This approach improved the overall quality and consistency of the dataset.
- **Tooling and Visual Inspection**: Annotation tools with image overlays and class consistency checks were employed to reduce human error. Manual inspection of samples across all classes was routinely conducted to validate the annotations.
- **Expertise and Domain Knowledge**: The annotator (author) possesses substantial domain expertise in traffic behaviour and intelligent vehicle systems, which informed the labelling of complex scenarios such as rear light combinations, partial occlusions, and ambiguous object orientations.
- **Label Audits**: A representative subset of annotations was audited multiple times at different stages of dataset creation. Any inconsistencies identified during these audits were corrected and used to refine the annotation strategy.

While multi-annotator consensus or inter-annotator agreement metrics are common in large annotation teams, the consistent application of a well-defined labelling protocol by a domain-aware annotator provides strong reliability in this context.

3.3 Dataset Format

This section provides information about the dataset directory structure and how the data are stored.

Figure 3.2 depicts the dataset directory structure. After the root directory, each sample has a directory, ranging from 000 to 645. Inside each sample directory, there are the following directories and files:

- det/predict/labels: containing text files, with detection and tracking information, for each frame. The data is stored as follows [*object_Type, x, y, w, h, object_ID*].
- img_depth_info: contains the depth estimation frames in *.png* format. See Figure 3.1c.
- img_global_OOI_context: contains the object of interest frames in *.png* format. See Figure 3.1f.

- img_local_context: contains the frames of the target object bounding box in *.png* format. See Figure 3.1g.
- img_local_depth_context: contains the frames of the target object bounding box with depth estimation in *.png* format. See Figure 3.1i.
- img_no_target: contains the frames without the target object in *.png* format. See Figure 3.1b.
- img_original: contains the raw frames in *.png* format. See Figure 3.1a.
- img_overlayed: contains the blended frames in *.png* format. See Figure 3.1d.
- img_with_lane_info: contains the frames with lane information in *.png* format. See Figure 3.1e.
- tracking: contains three Excel files, one has the information about the surrounding objects, another has the target object information, and the last has merged information of the surrounding and the target objects. The data is stored as follows [*video_n, frame_n, start_frame, end_frame, hazard_type_int, hazard_type_name, hazard_flag, target_obj_id, ID, object_type, xc, yc, xc_speed, yc_speed, w, h, bbox_area, x_1, y_1, x_2, y_2*].
- object_position_labels: contains the TV side labels. The data is stored as follows [*video_n, frame_n, object_type, target_object, vehicle_position*]
- all_roadHazardDataset_videos.csv: the file containing the merging information of all videos.
- train\recognition_dataset_split.csv: the file containing the videos selected to train the model.
- test\recognition_dataset_split.csv.: the file containing the videos selected to test the model.

3.4 Dataset Applications

The dataset includes annotations for object detection, tracking, traffic hazard classes, rear light status, and the visible side of each object, making it versatile for a wide range of real-world intelligent traffic system applications.

For instance, the object detection annotations are particularly useful for advancing research in traffic-specific object detection across diverse environments, including highways, rural areas, and urban settings. The tracking information supports research in object tracking and trajectory prediction, which is essential for developing more accurate and reliable intelligent vehicle systems.

A key distinguishing feature of this dataset is the inclusion of traffic hazard annotations, which enable targeted research on the recognition and prediction of potential traffic hazard events. This is especially relevant for enhancing the safety features of intelligent vehicle systems operating in complex traffic scenarios. Additionally, the rear light status and visible side annotations—explored in detail in

subsequent chapters—provide valuable data for understanding vehicle intent and behaviour from the ego vehicle's perspective. These cues can serve as supplementary inputs for recognising and predicting traffic hazards or traffic agent behaviours.

Overall, the proposed dataset supports research in real-world applications such as improving the ability of IVs to safely navigate complex urban environments and enhancing traffic management systems.

3.5 Dataset Statistics

This section provides information about the total number of classes, the number of samples per class, and the types of traffic agents considered in the dataset for each potential application, including traffic hazard event recognition, rear light status recognition, and object visible side recognition. For the traffic hazard and object visible side recognition applications, the following objects were considered: pedestrians, bicycles, cars, motorcycles, groups of pedestrians, buses, road crossings, trucks, road work, traffic lights, riders, stop signs, dogs, and horses. For the rear light status recognition application, only vehicles were considered.

3.5.1 Traffic Hazard Application

The dataset contains sixteen types of hazard classes, including:

- Object crossing: the object moves across the observer's path (Figure 3.3a).
- Object emerging: a traffic situation where an object moves from a minor side road, driveway, or parking lot onto a main road (Figure 3.3b).
- Left/right cut-in: the object enters the lane directly in front of another object with limited space (Figures 3.3c and 3.3d).
- Object meeting: two objects approach each other from opposite directions and come into proximity or pass each other (Figure 3.3e).
- Object stopping: the object reduces its speed to a halt (Figure 3.3f).
- Object reversing: the object moves backward, typically when parking or driving through a driveway (Figure 3.3g).
- Vehicle hazard lights activated: pairs of lights on a vehicle flash simultaneously in an intermittent pattern, activated by the driver to warn surrounding road users of a potential hazard or emergency ahead (Figure 3.3h).
- Red traffic light is on: the red traffic light is illuminated, indicating that road users must come to a complete stop (Figure 3.3i).
- Object turning: an object changes direction, usually by turning at an intersection, navigating a curve, or executing a U-turn (Figure 3.3j).
- Road works: road construction or maintenance activities, typically signaled through signs, cones, and barriers (Figure 3.3k).

(a) Object Crossing. (b) Object Emerging. (c) Left cut in.

(d) Right cut in. (e) Object meeting. (f) Object stopping.

(g) Object reversing. (h) Hazard light on. (i) Red traffic light.

(j) Object turning. (k) Road works. (l) Object coming out.

(m) Object pulling up. (n) Object on the side of the road. (o) Pedestrian near parked car.

(p) Object on the middle of the road.

Figure 3.3: Traffic hazard categories on the potential traffic hazard dataset.

- Object coming out: the object exits from a location on the side of the road onto the roadway (Figure 3.3l).
- Object pulling over: the object slows down and stops at the side of the road, such as a bus stopping at a bus stop (Figure 3.3m).
- Moving object on the side of the road: an item in motion located alongside the EV's lane (Figure 3.3n).
- Pedestrian near parked vehicle: an individual walking or moving near a stationary vehicle, including pedestrians unloading their cars, walking between

parked vehicles, or entering/leaving their vehicles (Figure 3.3o).

- Object in the middle of the road: an item obstructing the EV's path (Figure 3.3p).

The dataset contains more than 645 potential traffic hazard events. The number of samples per class is shown in Table 3.2. Some events occur more frequently than others—for instance, the right cut-in, left cut-in, object stopping, and object crossing classes have more samples than the rest.

The higher number of samples for the right cut-in, left cut-in, and object crossing classes can be attributed to the targeted design of the PREVENTION and JAAD datasets, which focus on studying these specific traffic agent behaviours. In contrast, the absence of dedicated datasets for the remaining classes results in fewer available samples for those categories. Among the more frequently occurring traffic hazard events—excluding those lacking dedicated datasets—are object stopping, object turning, red traffic light, object meeting, object emerging, pedestrian near parked vehicles, and road works. This class imbalance presents a challenge for machine learning training and validation.

As part of the dataset design, the inclusion of a "no hazard" class was considered essential to enable the model to distinguish between hazardous and non-hazardous situations—a critical requirement for real-world deployment. The total number of hazard samples across all classes is 600, with individual class frequencies ranging from as few as 3 to as many as 113, introducing a significant class imbalance. To mitigate further bias, 300 "no hazard" samples were deliberately selected. This 600:300 case-control ratio (hazard vs. no hazard) provides a moderate inclusion of non-hazardous scenarios without overwhelming the minority hazard classes.

The decision to include 300 "no hazard" samples was guided by both practical and methodological considerations. First, it ensures the model is exposed to a meaningful number of non-hazard examples, reflecting real-world conditions where most traffic scenes are non-hazardous. Second, it avoids introducing a disproportionate number of control samples that could lead the model to favour predicting the majority "no hazard" class—an issue known as class dominance. By setting the number of "no hazard" samples to 300, the dataset maintains a more balanced distribution across all classes, improving the model's ability to generalize and avoid overfitting. This strategy aligns with common practice in case-control learning setups, where the number of control samples is increased judiciously to ensure adequate learning without skewing the classification boundary.

3.5.2 Vehicles' Rear Light Application

As depicted in Figure 3.4, the possible rear light statuses are:

- BLO: both the left indicator and brake lights are ON simultaneously (Figure 3.4a).
- BOO: the brake lights are ON (Figure 3.4b).
- BLR: the left and right indicators, along with the brake lights, are ON simultaneously (Figure 3.4c).

Table 3.2: Number of samples per Potential Traffic Hazard Event Classes.

Class Name	**N. of Samples**
no_hazard	300
right_cut_in	113
object_stopping	63
left_cut_in	61
object_crossing	60
object_turning	51
object_hazard_light_on	48
red_crossing_traffic_light	40
object_meeting	38
object_emerging	37
pedestrian_near_parked_vehicles	34
road_works	34
object_reversing	20
object_pulling_up	14
object_on_the_side_of the_road	11
object_coming_out	11
green_crossing_traffic_light	8
object_on_the_middle_of the_road	3

- BOR: the right indicator and brake lights are ON simultaneously (Figure 3.4d).
- OLO: the left indicator is ON (Figure 3.4e).
- OLR: both the left and right indicator lights are ON simultaneously (Figure 3.4f).
- OOO: all rear lights are OFF (Figure 3.4g).
- OOR: the right indicator is ON (Figure 3.4h).
- REVERSE: the reverse lights are ON (Figure 3.4i).
- UNK: although a vehicle is detected, the rear lights may not be visible. For example, in Figures 3.4j and 3.4l, vehicles are detected, but their rear lights are not visible due to orientation or occlusion. Figure 3.4k shows another case where the rear side of the vehicle is visible, but the image quality is too poor to discern the rear lights.

The dataset contains more than 49,637 samples of rear light statuses. Table 3.3 shows the number of samples per class. It reveals that some classes occur far more frequently than others. For instance, the "OOO," "BOO," "OOR," "OLO," and "UNK" classes each have between six and fifteen thousand occurrences. In contrast, the remaining rear light classes have significantly fewer samples, ranging from just a few hundred to around two thousand.

Due to this discrepancy in class distribution, two versions of the dataset were created: the Rear Light Status with six classes (RLS-6), which includes the "BOO," "OLO," "OLR," "OOO," "OOR," and "UNK" classes; and the RLS-10, which includes all ten classes. Note that some samples from RLS-10 classes not present in

(a) BLO (b) BOO (c) BLR (d) BOR

(e) OLO (f) OLR (g) OOO (h) OOR

(i) REVERSE (j) UNK1. (k) UNK2 (l) UNK3

Figure 3.4: Rear lights possible status and unknown examples.

RLS-6 were mapped to their corresponding RLS-6 classes. For example, "BLR" was converted to "OLR," "BOR" to "OOR," "BLO" to "OLO," and "REVERSE" to "UNK."

Table 3.3: Number of samples per available classes for the RLS-6 and RLS-10 datasets.

Categories	**RLS-6**	**RLS-10**
BLO	-	621
BOO	8468	8468
BLR	-	291
BOR	-	1208
OLO	7231	6610
OLR	2453	2162
OOO	15148	15148
OOR	9491	8283
REVERSE	-	300
UNK	6846	6546

3.5.3 Object Visible Side Application

As depicted in Figure 3.5, the possible visible sides of a target object are:

- Front-left: as illustrated in Figure 3.5a, the object is viewed from an angle that shows both the front and the left side. The front face is visible but slightly rotated, allowing part of the left side to be seen. This angle reveals the corner where the front and left sides meet, highlighting features such as the front-left edge, the left front wheel (if it's a vehicle), and portions of both the front and left sides.

- Front: as illustrated in Figure 3.5b, the object is viewed head-on, capturing the entire front face. Only the front surface is visible, with no side details. This angle emphasises features such as the object's front center, frontal design elements (e.g., headlights, grille, or emblem on a vehicle), and the symmetry of the front view.

- Front-right: as illustrated in Figure 3.5c, the object is viewed from a position that shows both the front and the right side. The front face is visible but slightly rotated to reveal part of the right side. This angle highlights the front-right edge, the right front wheel (for a vehicle), and partial views of both the front and right sides.

- Right: as illustrated in Figure 3.5e, the object is viewed directly from the right, showing the entire right side. The front and rear are not visible. This angle captures the full length of the object from this side, including features such as windows, doors (if it's a vehicle), or any markings specific to the right side.

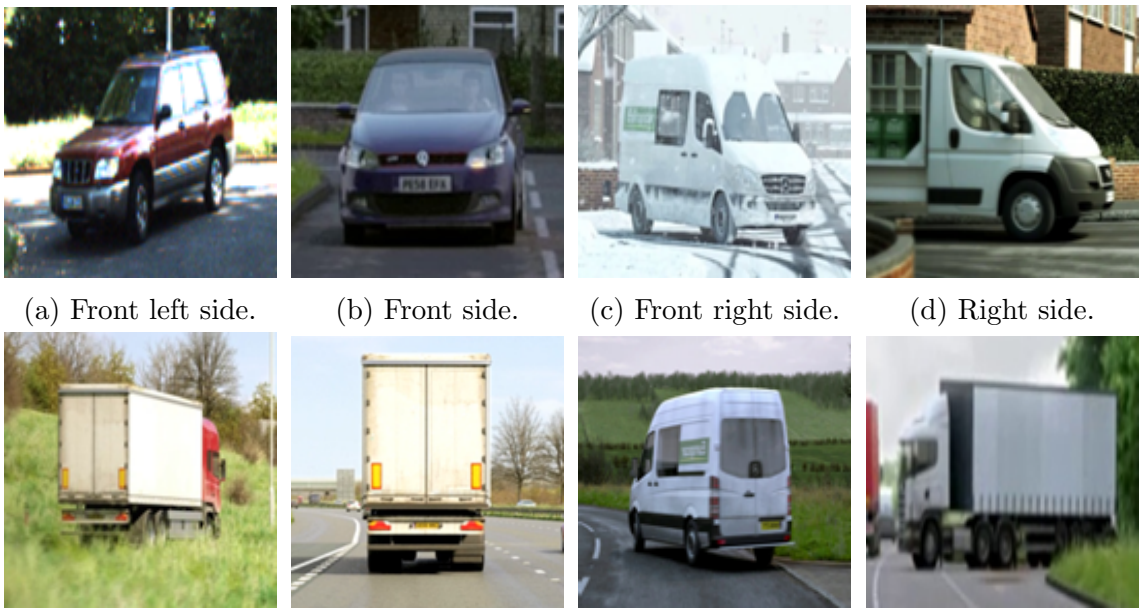
- Rear-right: as illustrated in Figure 3.5f, the object is viewed from an angle that shows both the rear and the right side. The rear is visible but slightly rotated to reveal part of the right side. This perspective shows the rear-right edge, the right rear wheel (if it's a vehicle), and parts of both the rear and right sides.

- Rear: as illustrated in Figure 3.5g, this is a direct view of the object's back, showing the entire rear face. Only the rear surface is visible, with no side details. This view emphasises rear design features such as tail lights, exhausts (for a vehicle), or any rear-specific elements.

- Rear-left: as illustrated in Figure 3.5h, the object is viewed from a position that shows both the rear and the left side. The rear is visible but slightly rotated to reveal part of the left side. This view captures the rear-left edge, the left rear wheel (for a vehicle), and partial views of both the rear and left sides.

- Left: as illustrated in Figure 3.5d, the object is viewed directly from the left, showing the entire left side. The front and rear are not visible. This angle captures the full length of the object from this side, including features such as windows, doors (if it's a vehicle), or any markings specific to the left side.

Table 3.4 reports the number of samples per class type. There is a noticeable imbalance in the number of samples among the classes. The rear, rear-right, and rear-left classes have the highest number of samples. This is primarily because TVs in the novel dataset are typically positioned in front of the EV and moving in the



(a) Front left side. (b) Front side. (c) Front right side. (d) Right side.

(e) Rear right side. (f) Rear side. (g) Rear left side. (h) Left side.

Figure 3.5: Possible sides of the TV that can be observed by the EV.

same direction. For instance, the most common object type encountered in front of an EV is another vehicle traveling in the same direction, making the rear side the most frequently observed. Additionally, the PREVENTION dataset—one of the sources used to construct the novel dataset—was specifically designed to study lane change behavior in highway scenarios, further contributing to the prevalence of rear-side views.

Table 3.4: Number of samples per available classes for the target's object visible side datasets.

Categories	**N. Of Samples**
front_side	2185
front_left_side	1739
front_right_side	1387
left_side	3738
right_side	2525
rear_side	34481
rear_left_side	23061
rear_right_side	27864
UNK	11273

3.6 Comparison to Related Datasets

The novel dataset differs from the datasets discussed in Chapter 2, Sections 2.2.5 and 2.3.2, in the following ways:

- It includes discrete behaviours from multiple types of traffic agents, such as

cars, trucks, pedestrians, traffic lights, road work signs, cones, barriers, animals, groups of pedestrians, and riders.

- It considers more than 15 types of discrete behaviours.
- The samples focus on traffic hazard events in their potential stage.
- It features a wide variety of traffic scenarios, including highways and complex urban scenes from different countries.
- It includes a mixture of real and synthetic data.

Regarding rear light annotation, as discussed in Section 2.4, existing datasets typically provide rear light status across sequences of images, making them unsuitable for systems that require single-image rear light status recognition. Moreover, datasets that do offer per-image annotations are often limited to nighttime conditions and may only indicate whether an indicator is active—without specifying which side.

In contrast, the proposed dataset includes the full range of rear light categories described in Section 3.5. Each annotation corresponds to an individual image, making the dataset suitable for both single-image and sequence-based systems. Additionally, it encompasses a wide variety of traffic scenarios and environmental conditions.

3.7 Conclusions

This chapter presented a novel dataset designed to enhance the understanding of traffic agent behaviour, with a particular focus on identifying and analysing agents that pose potential hazards. The proposed dataset is highly versatile, supporting a wide range of research areas, including rear light status recognition, object visible side analysis, traffic agent behaviour modelling, and potential traffic hazard event recognition.

Despite the large number of samples, the dataset is characterised by class imbalance, as certain traffic agent behaviours—such as objects emerging into traffic or pulling over—and specific rear light statuses—such as active brake lights combined with indicators or reverse lights—are inherently less frequent.

Compared to existing datasets focused on traffic agent behaviour and hazard detection, the proposed dataset captures the complexity of traffic scenes more effectively. It includes a broader variety of traffic agent behaviours, agent types, explicit cues, and environmental contexts, making it a more comprehensive resource for research. This complexity more accurately reflects real-world traffic scenarios and provides a richer foundation for developing and evaluating models capable of handling such challenges.

However, the presence of unbalanced classes remains a limitation. Future work could address this issue through methods such as data augmentation or targeted data collection to improve class balance.

In conclusion, this dataset makes a significant contribution to the field, offering a robust resource for advancing research in traffic agent behaviour and hazard detection. Its development supports the broader goals of this thesis by providing a foundation for more accurate and comprehensive studies—essential for improving traffic safety and enabling the development of intelligent vehicle systems.

Chapter 4

Deep Learning for Rear Light Status Detection and Visible Side Recognition

This chapter aims to develop and evaluate deep learning algorithms to independently recognise rear light status and the visible side of objects from the ego vehicle's perspective. The goal is to create additional modules for a comprehensive potential traffic hazard event recognition pipeline, where the output of these algorithms can serve as supplementary input cues for predicting potential hazard events.

The chapter revisits the possible visible sides of target objects, the various rear light statuses of vehicles, and the challenges associated with recognising them. The potential impact of this research on the development of deep learning algorithms is promising and could significantly enhance the performance of traffic hazard recognition systems. Both cues are discussed within the same chapter, as the model architecture, hardware, and evaluation metrics used are consistent across both tasks.

The chapter is structured to provide a comprehensive understanding of the research. It begins with a brief introduction in Section 4.1, followed by a detailed explanation of the model architecture in Section 4.2. Section 4.3 presents the experimental evaluation. Section 4.4 discusses the results for rear light status recognition, while Section 4.5 presents the findings for object visible side recognition.

4.1 Background

In the field of traffic agent behaviour recognition, a wide range of input features has been explored, including categorical, numerical, and image-based inputs. Common categorical inputs include object type, while numerical inputs often involve bounding box position, speed, width, height, and area distances. Image inputs range from raw images to optical flow representations. However, some important cues that conventional drivers rely on have been overlooked and not explicitly considered. These include:

- Rear light status, which drivers use to communicate their intentions to others.
- The visible side of a target object, which drivers use to assess the direction in which the object is moving.

As discussed in Section 2.4, prior research has highlighted the importance of recognising rear light status to better understand the intent of surrounding traffic agents. Similarly, the visible side of a target object is a critical, though often overlooked, cue. For example, an object crossing in front of the EV typically reveals its left or right side. An object making a left turn in front of the EV may transition from showing its rear side to rear-left, and finally to the left side. Likewise, a vehicle changing lanes from left to center may shift from rear-right to rear as its visible side.

Recognising the visible side of a target object presents several challenges:

- Image quality may be degraded due to distance, occlusion, or poor lighting conditions.
- Class boundaries between similar categories (e.g., rear vs. rear-right, front vs. front-left) can be difficult to distinguish.
- Object complexity varies—pedestrians are small and dynamic, while vehicles are larger and more rigid.
- Relative motion between the EV and the target object can alter the visible side, even if the object is stationary.
- Lack of dedicated datasets for visible side classification further complicates the task.

Rear light cues also convey important information:

- An active left or right indicator signals an intention to turn.
- Brake lights indicate that the vehicle is slowing down or has stopped.
- Simultaneous activation of both indicators suggests a hazard ahead, prompting caution from following vehicles.

While various methods have been proposed for rear light recognition, most rely on image sequences rather than single-image inputs. Existing datasets often lack per-image annotations, focus primarily on nighttime conditions, or provide only partial information (e.g., indicator status without specifying the side). Additionally, many use outdated feature extractors and do not clearly describe their dataset-splitting strategies.

To address these challenges, this chapter proposes a series of contributions:

- Vehicle Rear Light Status Recognition (VRLSR) using the dataset introduced in Chapter 3, which includes categories such as all lights off, left/right indicators on, brake lights on, and combinations of indicators and brake lights.
- Vehicle Visible Side Recognition (VVSR) using the same dataset, which includes categories such as front, front-right, front-left, left, right, rear, rear-right, rear-left, and an unknown category.

These categories are used to develop and evaluate deep learning recognition algorithms tailored to each task.

4.2 Model Architecture

The tasks of recognising a vehicle's rear light status and identifying the visible side of the TV are treated as two distinct processes, each handled by a dedicated recognition system. Although these systems operate independently, they share a common architectural structure.

As illustrated in Figure 4.1, both pipelines consist of three primary components:

- Input image: cropped from the EV's front-facing camera.
- Feature extractor and classifier module: typically a convolutional neural network (CNN) or Transformer model, followed by a FC layer.
- Output layer: a set of predefined categories specific to each task.

4.2.1 Input Image

The architecture begins with the input image, typically captured by the EV's front-facing camera. This image includes the surrounding environment, such as nearby vehicles, road markings, and other relevant objects. To prepare the image for downstream analysis, several preprocessing steps are applied.

First, the region of interest—specifically, the bounding box containing the TV—is cropped from the original frame. The cropped region is then resized to a fixed resolution of 224×224 pixels, aligning with the input requirements of standard deep learning architectures.

Next, normalization is applied to standardize the pixel value distributions. This is done using the transforms module from the PyTorch library, with mean and standard deviation values matching those used during the training of the respective pretrained backbone models (e.g., ImageNet statistics).

To further enhance model generalization and robustness, data augmentation techniques are applied to the cropped images. These include random transformations and an AutoAugment policy tuned for the ImageNet dataset, enabling the model to learn more invariant features under varying lighting conditions and occlusions.

4.2.2 Classifier

To effectively encode visual information from the input image, this work investigates two primary families of feature extractors: CNNs and Transformer-based architectures. Each offers distinct advantages based on their architectural design and inductive biases.

CNNs—such as ResNet-18, EfficientNet-V2-S, and ConvNeXt-Small—are built on the principles of spatial locality and translational invariance. They use convolutional filters to progressively extract hierarchical features, from low-level edges and textures in early layers to high-level semantic patterns in deeper layers. This design is computationally efficient and well-suited for visual tasks with relatively constrained receptive fields. For instance:

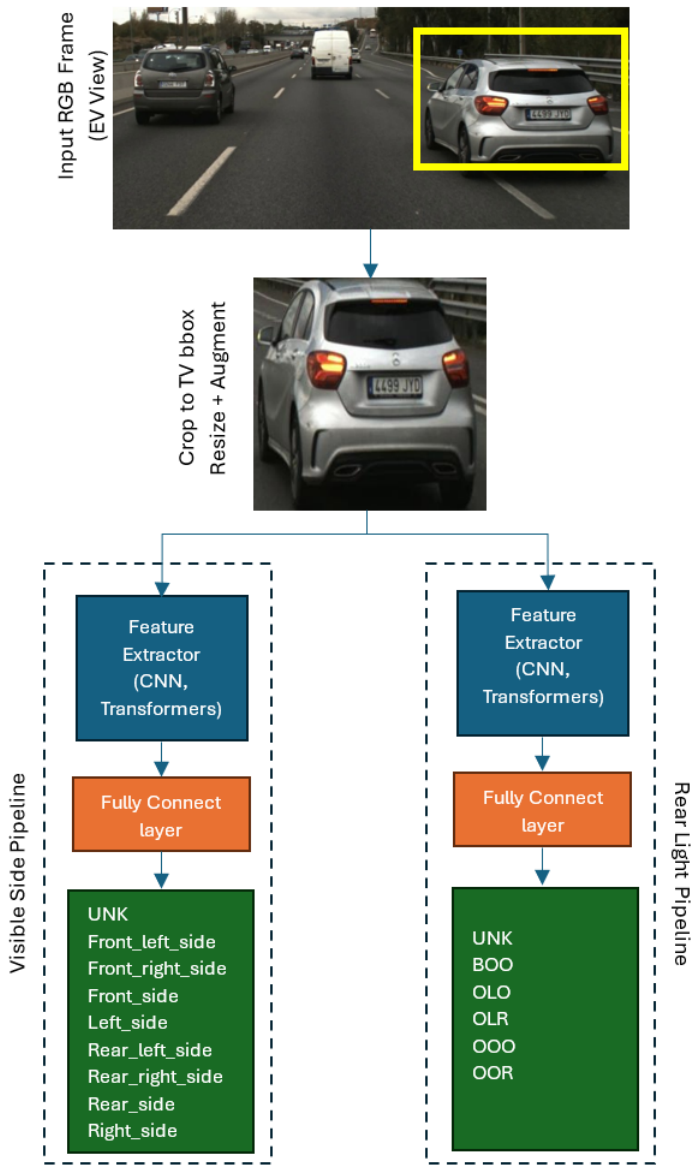


Figure 4.1: Parallel pipelines for recognising the visible side of the TV and its rear light status from an EV front-facing camera. Each pipeline consists of a dedicated feature extractor (CNN or transformer) and a FC layer to output class probabilities. These outputs can be used as high-level input features for other tasks such as lane change prediction and traffic hazard recognition.

- EfficientNet employs compound scaling to balance model depth, width, and resolution.
- ConvNeXt modernises traditional CNNs by incorporating design elements inspired by Transformers, such as inverted bottlenecks and GELU activation.

Transformer-based models, including ViT-B/16 and Swin-S, take a fundamentally different approach. Rather than relying on local kernels, they divide the input image into patches and apply self-attention mechanisms to capture long-range dependencies across the entire image. This global attention mechanism makes them more adaptive to varied visual patterns, such as asymmetric occlusions or complex lighting conditions, common in driving scenarios. Swin Transformers further

enhances efficiency by introducing hierarchical representations and windowed self-attention.

In this study, both CNNs and Transformers are employed to evaluate their respective strengths in recognising rear light status and the visible side of TVs. Each model is initialised with ImageNet-pretrained weights and fine-tuned end-to-end to adapt to the domain-specific characteristics of vehicle-centric visual cues.

4.2.3 Categories

The final output of the classifier is a set of categories corresponding to either the TV's rear light status or its visible side.

For rear light status recognition, the categories include: "BLO", "BOO", "BLR", "BOR", "OLO", "OLR", "OOO", "OOR", "REVERSE", and "UNK".

For visible side recognition, the categories include: Front, Front Right, Front Left, Left, Right, Rear, Rear Right, Rear Left, and Unknown.

The classifier outputs a probability distribution over these categories, with the highest probability indicating the predicted class for the given input image.

4.3 Experimental Evaluation

4.3.1 Classifiers

In contrast to existing research on VRLSR using single-image inputs—which often relies on traditional techniques and outdated CNNs—this chapter evaluates the performance of classifiers introduced from 2020 onward.

The selection of feature extractors was guided by several criteria: architectural diversity, proven performance in recent literature, computational feasibility, and suitability for fine-tuning on domain-specific tasks.

Transformer-based models include:

- ViT-B/16 [270], a pioneering vision transformer (ViT) that serves as a baseline for evaluating self-attention mechanisms.
- Swin-S [271], which improves upon ViT with a hierarchical design and windowed self-attention, offering a balance between performance and efficiency, particularly useful for interpreting small or partially occluded rear light configurations.

CNN-based models include:

- ResNet-18 [272], a well-established baseline known for its simplicity and interpretability.
- EfficientNet-V2-S [273], which uses compound scaling for efficient performance.
- ConvNeXt-Small [274], a modern CNN that incorporates Transformer-inspired design elements.
- RegNetY-8GF [275], a highly optimised CNN architecture known for its regular design space, efficient training, and robustness.

This curated selection ensures a broad comparison across architectural families, enabling a comprehensive evaluation of how design differences influence model performance in real-world driving scenarios. All models were implemented using the PyTorch framework and fine-tuned under consistent training protocols to ensure fair benchmarking.

The dataset was randomly split into 80% for training and 20% for validation.

4.3.2 Training Hyperparameters

The training of the VRLSR and VVSR models was conducted using a consistent architecture, with slightly tailored hyperparameters to better suit each task.

Both models used a batch size of 48 and input images resized to 224×224 pixels, ensuring compatibility with pre-trained CNN and Transformer-based backbones. Training was performed over 15 epochs with an initial learning rate of 0.0001 and a learning rate step size of 7.

However, some hyperparameters were adjusted between tasks:

- Weight decay was set to 0.0001 for VRLSR and 0.001 for VVSR, with the stronger regularization in VVSR aimed at mitigating overfitting.
- A learning rate scheduler was applied, using a gamma factor of 0.1 for VRLSR and 0.01 for VVSR.

Both models were optimized using Stochastic Gradient Descent (SGD) with a momentum of 0.9, and trained using cross-entropy loss, which is well-suited for multi-class classification tasks. These configurations were selected based on preliminary tuning experiments to optimize convergence and generalization performance for each recognition objective.

4.3.3 Hardware

The hardware used for the experiment was the NVIDIA GeForce GTX 1080Ti 12GB and an Intel(R) Core(TM) i7-10750H CPU @2.60GHz.

4.3.4 Metrics

To evaluate classifier performance, the following metrics were adopted: accuracy, precision, recall, F1-score, and the confusion matrix.

- Accuracy was used to measure how closely the predicted results match the ground truth labels.
- Precision was used to assess how consistent the predictions are within each class—i.e., how many of the predicted instances for a class are actually correct.
- The confusion matrix provides a visual representation of misclassifications, helping to identify whether the classifier is confusing similar classes (e.g., predicting left indicator instead of the ground truth right indicator).

It is important to note that, apart from accuracy, all other metrics were applied only to the best-performing classifier for each task.

4.4 Rear Light Recognition Results and Discussion

This section presents and discusses the results of the rear light recognition models evaluated on two distinct datasets: VRLSR-6 and VRLSR-10. The performance of various classifiers is compared based on their accuracy, and a more detailed analysis is provided for the best-performing model, including precision, recall, F1-score, and confusion matrix.

Classifier Performance on VRLSR-6 and VRLSR-10 Datasets

Table 4.1 shows the classification accuracy of the models listed in Section 4.3 when tested on the VRLSR-6 and VRLSR-10 datasets. The following observations can be made:

VRLSR-6 Dataset:

- ViT-B/16 achieved the highest accuracy at 77%, making it the best-performing model for this dataset.
- Swin-S followed closely with 76%, while EfficientNet-V2-S and ConvNeXt-Small both achieved 74%.
- ResNet-18 and RegNetY-8GF had the lowest accuracy, each at 72%.

VRLSR-10 Dataset:

- ViT-B/16 again led with an accuracy of 70%. EfficientNet-V2-S and ConvNeXt-Small also performed well, each achieving 70%.
- ResNet-18 achieved a moderate 66%, while RegNetY-8GF and Swin-S trailed with 58% and 56%, respectively.

These results suggest that ViT-B/16 is particularly effective for rear light recognition, especially when handling a larger number of classes (VRLSR-10). However, all models experienced a drop in accuracy when transitioning from VRLSR-6 to VRLSR-10, highlighting the increased complexity and difficulty of the task as the number of classes grows.

Table 4.1: VRLSR accuracy performance for various classifiers from 2020 onward, and for Resnet_18 as the baseline.

Classifier	**VRLSR-6 Acc(%)**	**VRLSR-10 Acc(%)**
ResNet_18	72	66
ViT_B_16	77	70
RegNet_Y_8GF	72	58
Swin_S	76	56
EfficientNet_V2_S	74	70
ConvNeXt_Small	74	70

Detailed Performance Analysis of ViT_B_16 on the VRLSR-6 Dataset

Table 4.2 presents a detailed breakdown of the ViT_B_16 model's performance on the VRLSR-6 dataset, offering insights into how well the model performs across individual classes. The table includes precision, recall, and F1-score, which are essential metrics for evaluating classification effectiveness.

Precision measures the proportion of correctly predicted instances among all instances predicted as a given class. Recall measures the proportion of actual instances of a class that were correctly identified. A model with high recall but low precision tends to produce more false positives, while one with high precision but low recall may miss actual positive cases, resulting in false negatives. The F1-score provides a harmonic mean of precision and recall, offering a balanced measure of a model's performance, especially in the presence of class imbalance. This analysis helps identify which rear light statuses are more challenging for the model to classify and where improvements may be needed.

- "BOO" Class: The model achieved a precision of 61.31% and a recall of 59.96%, resulting in an F1-score of 60.63%. This indicates moderate performance, with a slight bias toward precision over recall.
- "OLO" Class: This class achieved a precision of 77.67% and a recall of 79.78%, leading to a strong F1-score of 78.71%. The high values for both precision and recall suggest that the model reliably identifies instances of the "OLO" class.
- "OLR" Class: The model struggled with this class, achieving a low precision of 59.03% but a high recall of 78.01%, resulting in an F1-score of 67.21%. The low precision indicates a significant number of false positives, meaning the model often misclassified other classes as "OLR." One possible reason for this performance could be the relatively small number of samples compared to other classes.
- "OOO" Class: This class showed robust performance, with a precision of 80.40%, a recall of 81.42%, and an F1-score of 80.90%. The balanced precision and recall indicate that the model effectively recognises the "OOO" class.
- "OOR" Class: The model achieved a precision of 82.72% and a recall of 74.07%, resulting in an F1-score of 78.16%. The lower recall compared to precision suggests some difficulty in identifying all instances of the "OOR" class.
- "UNK" Class: The model performed exceptionally well on this class, achieving high scores across all metrics: 80.16% precision, 83.13% recall, and an F1-score of 81.62%. This indicates that the model is particularly effective at distinguishing the "UNK" class from others.

In the field of intelligent vehicles (IV)s, recognising the rear light status of other road users requires balancing high recall and high precision, as both impact safety and driving efficiency. A system with high recall accurately detects most rear light signals (e.g., brake lights, turn signals), even if it occasionally misidentifies a signal that is not present (false positives). High recall is particularly important for safety-critical functions, such as:

- **Brake Light Detection**: If an IV fails to detect a vehicle's brake lights (false negative), it may not slow down in time, increasing the risk of rear-end collisions. A high-recall system ensures that nearly all actual braking events are detected, even if it occasionally identifies braking when none is occurring.
- **Turn Signal Recognition**: If an IV does not recognise another vehicle's turn signal, it may fail to anticipate lane changes or turns, leading to hazardous situations. A high-recall system ensures that almost all turn signals are detected, enhancing response time and decision-making.

On the other hand, high precision ensures that when an IV detects a rear light signal, it is highly confident in its accuracy, reducing false positives. This is essential to prevent unnecessary or incorrect responses that could compromise driving efficiency and safety, such as:

- **False Brake Light Detection**: If an IV incorrectly detects a brake light when none is present (false positive), it may slow down unnecessarily, resulting in abrupt driving behaviour or even increasing the risk of being rear-ended. A high-precision system ensures that braking is only detected when a vehicle ahead is genuinely slowing down.
- **Turn Signal Misinterpretation**: If an IV mistakenly identifies a turn signal, it may yield or adjust its trajectory unnecessarily, disrupting traffic flow. A high-precision system minimises such errors, ensuring smoother and more predictable vehicle behaviour.

Table 4.2: Precision, recall, and F1-score results achieved by the ViT_B_16 classifier when tested with the VRLSR-6 dataset version.

Class	Precision%	Recall%	F1-Score%
BOO	61.31	59.96	60.63
OLO	77.67	79.78	78.71
OLR	59.03	78.01	67.21
OOO	80.40	81.42	80.90
OOR	82.72	74.07	78.16
UNK	80.16	83.13	81.62

A confusion matrix was also generated on the test dataset to further analyze the performance of the ViT_B_16 classifier. Figure 4.2 provides a detailed breakdown of the classifier's predictions compared to the actual labels. Each row of the matrix represents the true label, while each column represents the predicted label.

The confusion matrix reveals the following insights into the classifier's performance across different categories:

- **BOO Class**: The model correctly predicted the "BOO" status in 930 out of 1,551 instances, resulting in a moderate true positive rate. However, 621 samples were misclassified, with the most significant errors involving the "OOO" (198), "OLO" (142), "OOR" (137), and "OLR" (113) classes. This suggests the classifier struggles to distinguish between active brake and indicator lights.

Qualitative analysis (see Figure A.2) shows that brake lights were often misclassified as "OOO" due to sunlight obscuring their brightness or because the lights appeared too small relative to the vehicle size. Reflections from sunlight on the rear lights also led to misclassifications as indicators. Additionally, motorcycles with a single brake light were frequently misclassified, and vehicle angles sometimes caused left lights to appear on the right side of the image.

- **OLO Class**: The model correctly predicted the "OLO" status in 1,294 out of 1,622 instances, yielding a high true positive rate. The 328 misclassifications were primarily with "BOO" (162) and "OOO" (90). This indicates difficulty in distinguishing between left indicators, brake lights, and inactive lights. Qualitative analysis (see Figure A.3) shows that simultaneous activation of brake and indicator lights often led to confusion, and dim or small indicators—especially on trucks—were sometimes missed.

- **OLR Class**: The model correctly predicted the "OLR" status in 330 out of 423 instances. The 93 misclassifications were mostly with "OOO" (40), suggesting difficulty in distinguishing hazard lights from inactive lights. Qualitative analysis (see Figure A.1) indicates that distant vehicles or dimming indicators during their cycle contributed to these errors.

- **OOO Class**: The model correctly predicted the "OOO" status in 2,686 out of 3,299 instances. The 613 misclassifications were primarily with "UNK" (189), "BOO" (145), "OLO" (134), and "OOR" (119). This suggests challenges in distinguishing inactive lights from other statuses. Qualitative analysis (see Figure A.4) shows that poor visibility of rear lights led to confusion with "UNK," while illuminated tail lights or sunlight reflections caused misclassifications as active signals.

- **OOR Class**: The model correctly predicted the "OOR" status in 1,671 out of 2,256 instances. The 585 misclassifications were mainly with "BOO" (251) and "OOO" (187). This indicates difficulty in distinguishing right indicators from brake lights and inactive lights. Qualitative analysis (see Figure A.5) shows that simultaneous activation of brake and indicator lights, as well as dim or small indicators, contributed to these errors.

- **UNK Class**: The model correctly predicted the "UNK" status in 1,079 out of 1,298 instances. The 219 misclassifications were mostly with "OOO" (140). The "UNK" class includes samples where the rear lights are obscured by sunlight, the reverse light is on, or the front side of the vehicle is visible. Qualitative analysis (see Figure A.6) shows that confusion often arose from misidentifying the vehicle's orientation or mistaking bright reflections for inactive lights.

4.5 Visible Side Results and Discussion

This section presents and discusses the results of the VVSR models tested on the proposed novel dataset. Like the VRLSR, various classifiers are compared based on their accuracy, and a more detailed analysis is provided for the classifier that achieved the best result, including precision, recall, F1-score, and confusion matrix.

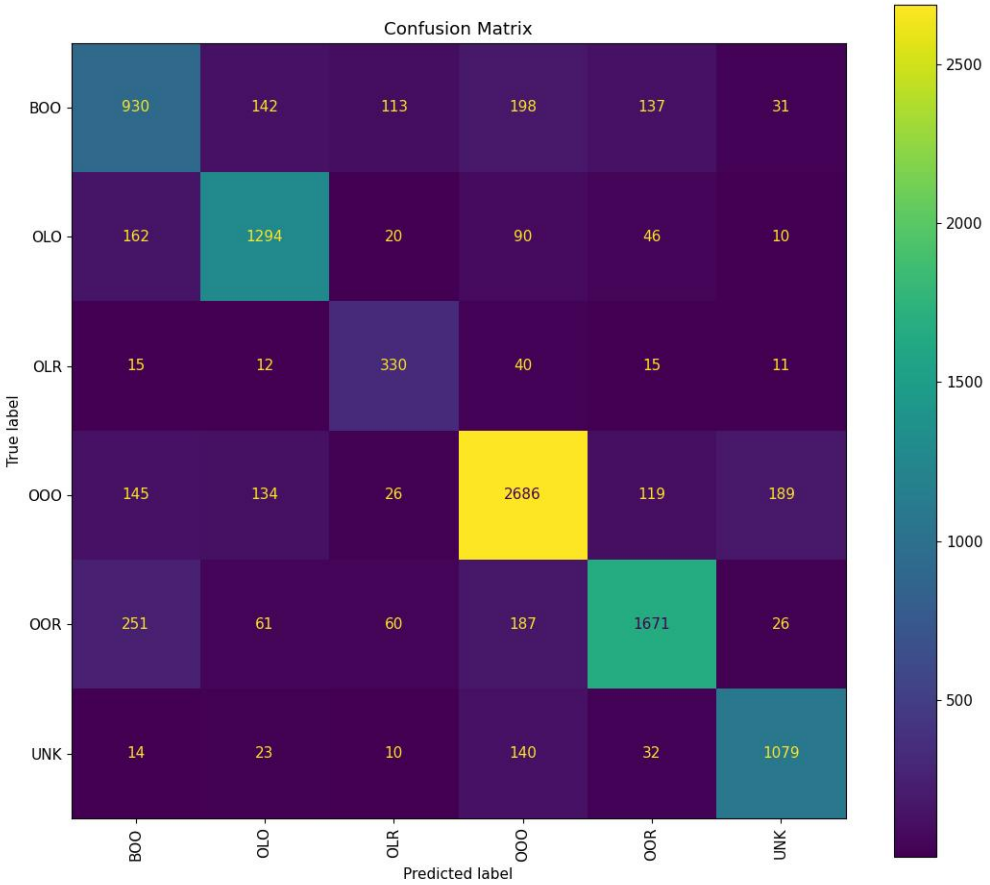


Figure 4.2: Confusion matrix for the ViT_B_16 classifier when using the test set of the VRLSR-6 dataset.

Classifier Performance Across Models

Table 4.3 presents the classification accuracy of the models listed in Section 4.3, evaluated on the proposed novel dataset. The following observations can be made:

- The **ConvNeXt_Small** model achieved the highest accuracy at **83%**, making it the best-performing model.
- The **ViT_B_16**, **ResNet_18**, **EfficientNet_V2_S**, and **Swin_S** models also performed well, achieving accuracies of **82%**, **81%**, **80%**, and **79%**, respectively.
- **RegNet_Y_8GF** was the least accurate, with an accuracy of **70%**.

These results suggest that the **ConvNeXt_Small** model is particularly effective for object visible side recognition tasks.

Table 4.3: VVSR accuracy performance for various classifiers from 2020 onward, and for Resnet_18 as the baseline.

Classifier	**Acc(%)**
ResNet_18	81
ViT_B_16	82
RegNet_Y_8GF	70
Swin_S	79
EfficientNet_V2_S	80
ConvNeXt_Small	83

Detailed Performance Analysis of ConvNeXt_Small

Table 4.4 provides a breakdown of the ConvNeXt_Small model's performance, offering insights into how well the model performs across different classes.

- **UNK Class**: The model performed exceptionally well on this class, achieving a precision of 91.66% and a recall of 90.32%, resulting in an F1-score of 90.98%. The high precision and recall indicate that the model reliably identifies instances of this class.

- **front_left_side Class**: This class had a precision of 61.87% and a recall of 51.25%, leading to an F1-score of 56.06%. This reflects fair performance, with a bias toward precision over recall—indicating that while the model is reasonably accurate in its predictions, it misses nearly half of the actual instances.

- **front_right_side Class**: The model struggled with this class, achieving a low precision of 55.17% and a recall of 52.98%, resulting in a poor F1-score of 54.05%. This suggests the model frequently misclassifies instances and fails to detect many true positives.

- **front_side Class**: The model's performance on this class was moderate, with a precision of 56.02% and a high recall of 72.88%, resulting in an F1-score of 63.35%. The significantly higher recall suggests that the model detects most instances but also produces many false positives.

- **left_side Class**: The model struggled with this class, achieving a low precision of 44.90% but a high recall of 76.52%, resulting in a fair F1-score of 56.59%. The very low precision indicates a high number of false positives, meaning the model often incorrectly classifies other instances as belonging to the "left_side" class.

- **rear_left_side Class**: The model performed exceptionally well on this class, achieving high scores across all metrics: 90.09% precision, 86.08% recall, and an F1-score of 88.04%. This indicates that the model is particularly effective at distinguishing the "rear_left_side" class from others.

- **rear_right_side Class**: The model also performed well on this class, with 84.30% precision, 89.04% recall, and an F1-score of 86.60%. This suggests the model effectively distinguishes the "rear_right_side" class from others.

- **rear_side Class**: The model achieved strong performance on this class, with 87.04% precision, 78.58% recall, and an F1-score of 82.59%. The slight bias toward precision over recall indicates that while the model misses a few instances, it is generally reliable.

- **right_side Class**: The model showed moderate performance on this class, achieving a fair precision of 60.39% and a high recall of 71.74%, resulting in an F1-score of 65.58%. The higher recall suggests that the model detects most instances, though at the cost of some false positives.

In the development of IVs, accurately recognising the visible side of target objects from the EV's perspective is essential for assessing traffic dynamics and identifying potential hazards. Achieving an optimal balance between high recall and high precision in this recognition process is crucial for ensuring both safety and efficiency within an advanced traffic hazard event recognition system. A system with high recall correctly identifies most visible sides of surrounding objects, even if some classifications are incorrect (false positives). Prioritising recall is particularly important when missing a visible side (false negative) could result in inaccurate trajectory predictions or safety-critical errors. For example:

- **Obstacle Avoidance and Path Planning**: If the system fails to detect a visible side (false negative), it may incorrectly assess an object's orientation, leading to inaccurate motion predictions. For instance, if a pedestrian's right or left side is not correctly detected, the IV might fail to anticipate their movement across the vehicle's path, increasing the risk of a collision.

- **Intersection and Turning Scenarios**: At intersections, failing to recognise a turning vehicle's visible sides (e.g., transitioning from rear side to rear-left and then to left side) may lead to incorrect assumptions about its intended path. High recall ensures the system captures these transitions, improving predictive accuracy and decision-making.

On the other hand, high precision ensures that when the IV classifies a visible side, it is highly confident in its correctness, reducing false positives. This is particularly important for preventing misleading detections that could result in inefficient or unsafe driving behaviour. For example:

- **Minimising Incorrect Manoeuvrers**: If the system misclassified a vehicle's *rear side* as its *front*, it may assume the vehicle is approaching rather than moving away. This misinterpretation could lead to unnecessary braking or evasive actions, disrupting traffic flow. High precision helps prevent such errors, ensuring more accurate decision-making.

Overall, the classifier performs well on classes such as UNK", rear_lef_side", and rear_right_side". However, performance significantly drops for classes like front_left_side", front_right_side", front_side", and "left_side". A potential explanation for the reduced performance in these classes is the smaller number of available samples compared to other classes.

A confusion matrix was generated using the test dataset to further evaluate the performance of the ConvNeXt_Small classifier in identifying the visible side of the

Table 4.4: Precision, Recall, and F1-score values for each class when using the ConvNeXt_Small classifier.

Class	Precision%	Recall%	F1-Score%
UNK	91.66	90.32	90.98
front_left_side	61.87	51.25	56.06
front_right_side	55.17	52.98	54.05
front_side	56.02	72.88	63.35
left_side	44.90	76.52	56.59
rear_left_side	90.09	86.08	88.04
rear_right_side	84.30	89.04	86.60
rear_side	87.04	78.58	82.59
right_side	60.39	71.74	65.58

target object. Figure 4.3 presents a detailed comparison between the predicted and actual labels. In the matrix, each row corresponds to the true label, while each column represents the predicted label.

The confusion matrix provides the following insights into the classifier's performance across different categories:

- **UNK class**: The model correctly predicted the "UNK" class in 2,220 out of 2,458 instances, yielding an exceptional true positive rate. However, 202 samples were misclassified, primarily as "rear_right_side" (79 instances) and "rear_side" (62 instances). Qualitative analysis (see Figure B.9) indicates that misclassifications often occurred due to the presence of multiple objects within the target bounding box. Although these instances were manually labelled as "UNK," the model based its prediction on one of the objects, leading to incorrect classifications.

- **front_left_side class**: The model correctly predicted the "front_left_side" class in 185 out of 361 instances, resulting in a fair true positive rate. A total of 114 samples were misclassified, most notably as "front_side" (86) and "right_side" (51). Qualitative analysis (see Figure B.1) shows that confusion with "front_side" occurred when only a small portion of the left side was visible. Misclassification as "right_side" often happened when only a small portion of the front side was visible.

- **front_right_side class**: The model correctly predicted the "front_right_side" class in 80 out of 151 instances, indicating a poor true positive rate. Of the 65 misclassified samples, the most significant confusion was with "front_side" (17 instances). Qualitative analysis (see Figure B.2) suggests that this confusion arose when only a small portion of the right side was visible.

- **front_side class**: The model correctly predicted the "front_side" class in 465 out of 638 instances, resulting in a fair true positive rate. However, 365 samples were misclassified, most notably as "left_side" (66 instances). Qualitative analysis (see Figure B.3) indicates that these errors occurred in blurred images where the subject was too far away, reducing visibility.

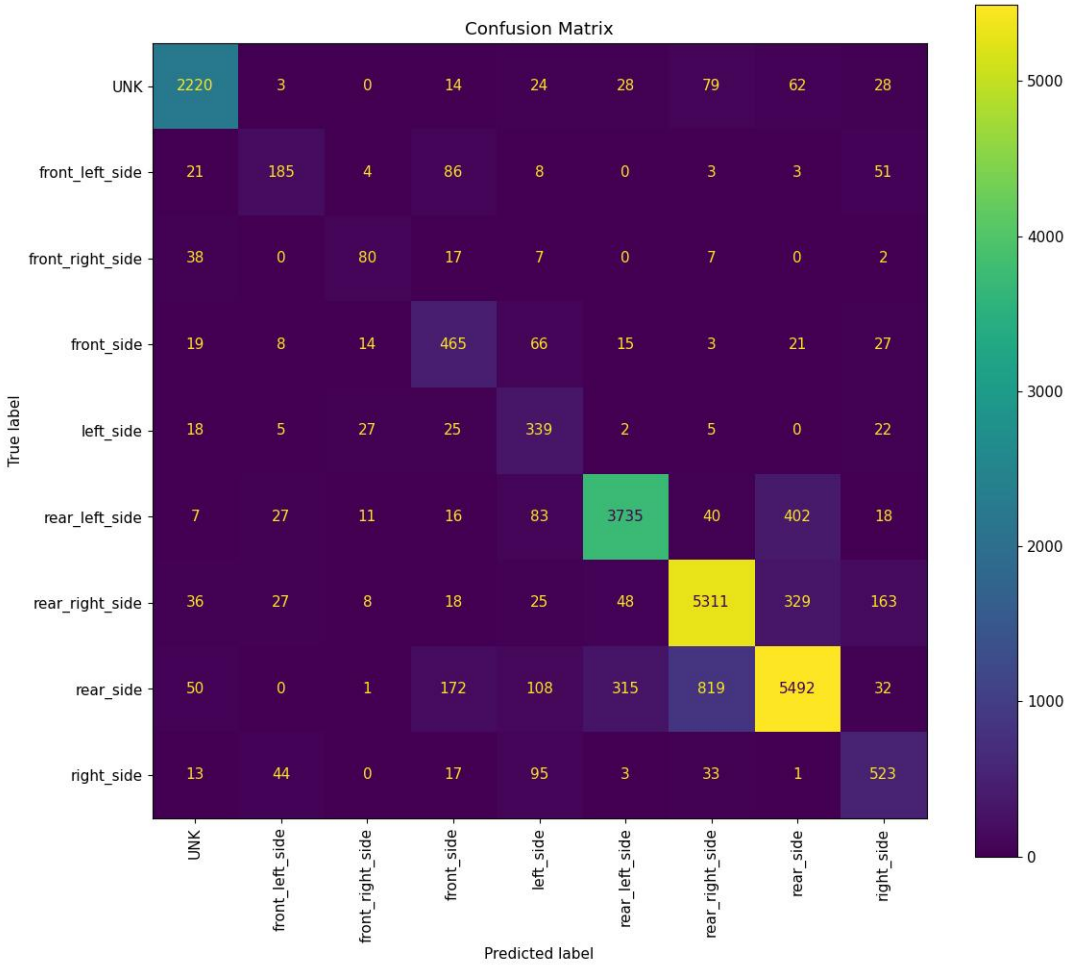


Figure 4.3: Confusion matrix for the ConvNeXt_Small classifier when using the test set proposed novel dataset.

- **left_side class**: The model correctly predicted the "left_side" class in 339 out of 443 instances, indicating a poor true positive rate. A total of 416 samples were misclassified, most frequently as "front_right_side" (27), "front_side" (25), and "right_side" (22). Qualitative analysis (see Figure B.4) reveals that "left_side" was often confused with "front_right_side" when images included both the left and a small portion of the front side. Misclassifications as "front_side" typically involved multiple objects or distant subjects. Confusion with "right_side" was common when the object was a walking person, as their posture made it difficult to distinguish between sides.

- **rear_left_side class**: The model correctly predicted the "rear_left_side" class in 3,735 out of 4,339 instances, resulting in a high true positive rate. However, 411 samples were misclassified, primarily as "rear_side" (402 instances). Qualitative analysis (see Figure B.5) shows that confusion occurred when only a small portion of the left side was visible, making it difficult to distinguish between "rear_left_side" and "rear_side."

- **rear_right_side class**: The model correctly predicted the "rear_right_side" class in 5,311 out of 5,965 instances, also indicating a high true positive rate. However, 989 samples were misclassified, most notably as "rear_side" (329) and "right_side" (163). Qualitative analysis (see Figure B.6) reveals that confusion with "rear_side" occurred when only a small portion of the right side was visible. Misclassification as "right_side" happened when only a small portion of the rear side was visible.

- **rear_side class**: The model correctly predicted the "rear_side" class in 5,492 out of 6,989 instances, resulting in a high true positive rate. However, 818 samples were misclassified, primarily as "rear_right_side" (819) and "rear_left_side" (315). Qualitative analysis (see Figure B.7) indicates that confusion arose when images of the rear side included small portions of either the right or left side.

- **right_side class**: The model correctly predicted the "right_side" class in 523 out of 729 instances, resulting in a moderate true positive rate. A total of 343 samples were misclassified, most significantly as "left_side" (95 instances). Qualitative analysis (see Figure B.8) shows that confusion between the right and left sides was common when the object was a walking person, as their posture made side differentiation more challenging.

4.6 Conclusions

This chapter aimed to develop deep learning algorithms for recognising the rear light status and the visible side of a target object. The approach involved evaluating the performance of various classifiers introduced since 2020 and selecting the most effective model for each specific task.

The results demonstrate that classifier performance varies depending on dataset complexity. Notably, the ViT_B_16 model consistently outperformed others across both datasets, highlighting the advantage of transformer-based architectures in complex visual recognition tasks such as rear light detection.

A detailed analysis of the ViT_B_16 model on the VRLSR-6 dataset revealed class-specific performance challenges. Variability in precision, recall, and F1-score across different classes indicates that, while the model performs well overall, it struggles with certain classes, particularly "OLR," which shows high recall but low precision, suggesting frequent false positives.

Misclassifications in rear light status were primarily influenced by factors such as sunlight reflection, occlusion, similar signals across different rear light statuses, vehicle angle and position, disproportionate rear light size relative to the vehicle, proximity of rear lights, weak brightness, and confusion between the rear and front sides of the vehicle.

In IV applications, safety remains the top priority, making high recall essential to avoid missing critical rear light or visible side classifications. However, precision is equally important to prevent false alarms that could lead to inefficient or erratic driving. An optimal system must balance recall and precision to ensure both safe and smooth navigation.

These findings suggest that while current models achieve reasonable accuracy in rear light recognition, there is room for improvement, particularly in handling challenging classes and maintaining performance as the number of classes increases. Future work could explore advanced techniques such as ensemble learning or data augmentation to enhance model robustness and accuracy. Additionally, investigating the underperformance of models like RegNet_Y_8GF and Swin_S on the VRLSR-10 dataset may offer insights for further refinement. Incorporating headlight status into the analysis could also prove beneficial.

The ConvNeXt_Small model demonstrated strong performance in object-visible side recognition, particularly excelling in the "UNK," "rear_left_side," and "rear_right-_side" classes. However, challenges remain with classes such as "front_left_side," "front_right_side," and "left_side," often due to limited visibility and the presence of multiple objects. Ambiguities in class boundaries—such as between "rear_right_side" and "rear_side," or "rear_left_side" and "rear_side"—further complicate classification.

Addressing these challenges may involve improving image quality, enhancing the model's ability to handle partial visibility, increasing sample sizes for underrepresented classes, and refining the training dataset with more diverse examples.

Chapter 5

Vehicle Lane Change Recognition and Prediction

This chapter presents a vehicle intention recognition and prediction system that integrates a deep learning algorithm, manually extracted features, and a novel evaluation metric. Leveraging the onboard PREVENTION dataset, the proposed approach employs Long Short-Term Memory Recurrent Neural Networks (LSTM-RNNs) to predict lane change (LC) manoeuvres. Unlike previous studies that relied on top-view datasets, this method utilises onboard data, which is more representative of real-world scenarios in Intelligent Vehicle (IV) Systems.

The algorithm incorporates several distinctive manually extracted features, including indicator light status, object bounding box areas, the distance between the Target Vehicle (TV) and lane boundaries, lane identification, and the visible side of TVs within the ego vehicle's (EV) perception field. This marks the first application of some of these features within an LSTM-based framework for LC prediction.

A key innovation of this system is the introduction of the Prediction Horizon Time Ratio (PHTR), a novel metric designed to evaluate the algorithm's effectiveness in forecasting future LCs. An ablation study was also conducted to assess the impact of different input features and hyperparameters, including a comparison between LSTM and GRU network architectures.

The chapter is structured as follows: Section 5.1 formulates the problem of vehicle intention recognition and prediction. Section 5.2 describes the selected input features and model architecture. Section 5.3 outlines the experimental setup, including dataset details, parameters, hardware, and evaluation metrics. Finally, Section 5.4 presents and discusses the results.

5.1 Problem Formulation

In a highway environment, a TV can perform three primary manoeuvres: No Lane Change (NLC), also known as Lane Keeping (LK); Right Lane Change (RLC); and Left Lane Change (LLC). As described by [71] and illustrated in Fig. 5.1, a LC manoeuvre consists of three main stages: the starting point (F0), the Lane Change Event (LCE) stage (F1), and the endpoint (F2).

The starting point occurs when the TV begins to approach one of its lane boundaries or when its indicator light is activated. The LCE stage, as defined by [71], is the moment when the centreline of the TV crosses the highway lane boundary,

transitioning from the current lane to the target lane. Finally, the endpoint stage corresponds to the phase in which the TV aligns itself within the target lane and continues driving straight.

Based on these stages, an LC prediction manoeuvre can be defined as the ability of an algorithm to recognise a LC before the LCE stage occurs. Therefore, to predict an LC manoeuvre, the algorithm must first identify the type of manoeuvre the TV is performing and then determine how early it can recognise the LC manoeuvre prior to the LCE stage.

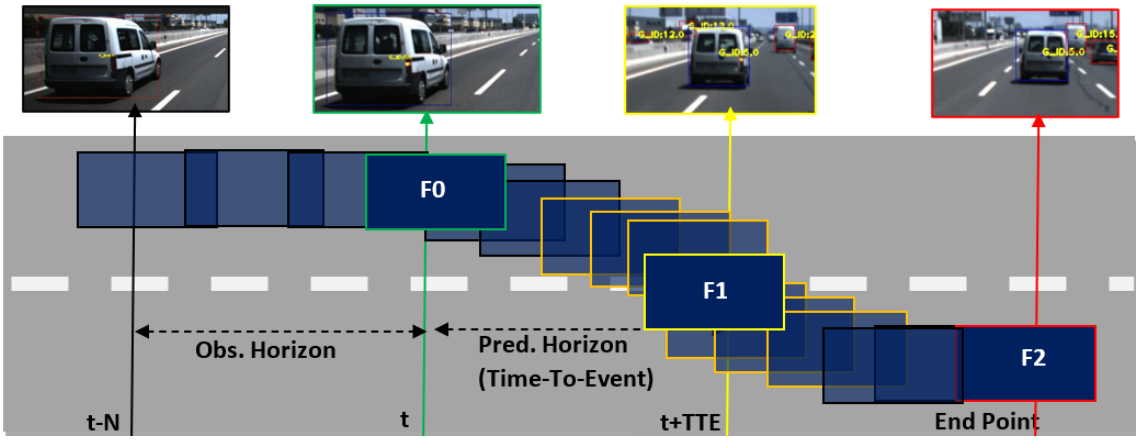


Figure 5.1: Lane change stages: The start point (t), LCE, and endpoint are represented by green, yellow, and red lines and borders, respectively. The Observation Horizon Time (OHT) refers to the number of frames preceding the start point. Frames with orange borders indicate those with greater displacement

.

Given that LC manoeuvres fall into three main categories, this study treats LC recognition as a classification problem. In this context, recognition refers to the algorithm's ability to classify the specific LC manoeuvre being executed by the TV. The LC recognition problem is formulated as follows: a series of feature vectors, $\{F_{t-OH}, ..., F_t\}$, extracted from a sequence of consecutive video frames $\{t-OH, ..., t\}$ obtained from an image sensor, is used by the proposed model to determine the type of LC manoeuvre the TV is performing. This can be expressed as:

$$LC_a^{t+n} \epsilon \{0, 1, 2\}, \tag{5.1}$$

where t is the timestamp of the last observed frame, n denotes the next frame after t, and the values 0, 1, and 2 represent NLC, RLC, and LLC, respectively. The features used in this study are described in Section 5.2.

It is important to note that during the prediction process, the recognition algorithm must be applied sequentially, with the observation time horizon shifting forward by one frame at each step, as illustrated in Figure 5.2 (a). The overall equation for recognising an LC manoeuvre is given by:

$$p(LC_a | F_{t-T_{obs}:t}). \tag{5.2}$$

In this study, prediction is defined as the instance in which the algorithm correctly recognises an LC manoeuvre between the starting point and the LCE stage. Figure 5.2 b) shows an example of LC prediction, where the PHT is 0.9 seconds, as

the first correct recognition occurs at frame 12—nine frames before the LCE. The average recognition accuracy from the first correct prediction to the LCE is calculated to evaluate how consistently the algorithm continues to make correct predictions. In this example, the average recognition accuracy is 90%, with 7 correct recognitions out of 9 frames.

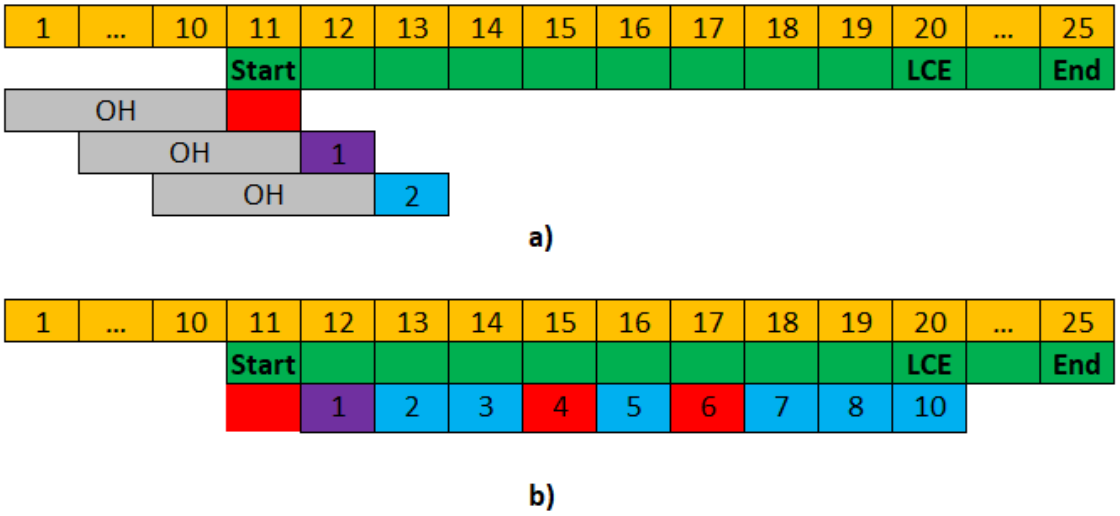


Figure 5.2: LC prediction example: The orange cells represent the frames of a complete LC manoeuvre sequence. The green cells indicate the ground truth labels between the starting point and the endpoint stages. The purple cell marks the first frame where the predicted LC matches the ground truth. Blue cells represent subsequent correct recognitions, while red cells indicate incorrect recognitions.(a) The recognition OH (grey cells) shifts one frame to the right at each prediction step. (b) The PHT is 9 frames before the LCE, as the first correct recognition occurs at frame 12.

5.2 Methods

Understanding the distinction between discretionary and mandatory LCs is essential when designing IV systems capable of recognising complex traffic behaviours. Discretionary LCs are typically initiated by the driver based on personal preferences or contextual advantages, such as overtaking slower traffic or moving into a faster lane. In contrast, mandatory LCs are driven by external factors, such as upcoming exits, lane terminations, or road obstructions. These two manoeuvre types differ not only in intent but also in the temporal patterns and contextual cues that precede them. Therefore, selecting an appropriate modelling framework is critical for accurately recognising and predicting such behaviours.

LSTM networks are a compelling choice for this problem due to their ability to capture and retain temporal dependencies in sequential data. Lane change behaviours evolve over time—particularly in discretionary cases where the intent may be subtle and influenced by surrounding traffic dynamics—and LSTMs are well-suited to learn such progression patterns. Their memory architecture enables the model to process sequences of observations while maintaining contextual awareness across multiple time steps. This makes LSTMs effective at distinguishing between

structured, externally driven mandatory changes and more variable discretionary ones. By leveraging this capability, LSTM-based systems can enhance prediction performance and support more adaptive, reliable decision-making in IVs.

This work introduces novel input representations and a deep learning framework named Vehicle Intention Prediction LSTM (VIP_LSTM), specifically designed to address the LC recognition and prediction problem using an onboard camera perspective. The following subsections describe the proposed input features and the architecture of the developed model.

5.2.1 Inputs

Previous research has primarily used vehicle type, position, and the velocity of the centre coordinates of the TV bounding boxes as input features for predicting LC manoeuvres. However, other important features—such as indicator light status, highway lane boundary information, and the distance between the corners of the TV bounding box and the lane boundaries—have often been considered only implicitly. Typically, these features are learned indirectly by applying a convolutional neural network (CNN) to raw RGB images. A significant drawback of this approach is that CNNs function as black boxes, making it difficult to determine which specific features are being utilised. The limited explicit use of these features is partly due to the lack of datasets that provide this information directly.

This work explicitly incorporates the aforementioned features by manually labelling samples from the existing PREVENTION dataset. Additionally, several new features are introduced: the bottom-right and bottom-left corners of the TV bounding box, which indicate whether the TV is approaching the right or left lane boundaries; the height, width, and surface area of the TV bounding box, which better capture the TV's longitudinal distance relative to the EV; and road lane identification, which provides insight into the TV's potential travel path. For example, on a three-lane highway, if the TV is in the left lane, it can only perform an RLC to move to the middle lane. Conversely, if the TV is in the right lane and there is no upcoming junction, it can only perform an LLC to move to the middle lane. Figure 5.3 illustrates these input features.

Another feature proposed in this study —previously unmentioned in the literature—is the visible side of the TV from the perspective of the EV. Figure 5.4 presents examples of the possible visible sides. This feature is relevant because it provides visual cues about the TV's motion.

For instance, the top row of Figure 5.4 illustrates a scenario where the EV is in the middle lane and the TV is in the right lane. While the TV remains in the right lane and the EV maintains its position, the visible side of the TV is categorised as "rear-left." However, once the TV initiates a LC from the right lane to the middle lane, its visible side shifts, and only the rear of the TV becomes visible from the EV's perspective.

5.2.2 Model Architecture

Although LSTM networks have been used to predict LC manoeuvre in previous works, they were mostly used on top-view datasets. The VIP_LSTM network depicted in Fig. 5.5, is a conventional vanilla LSTM that uses different combinations

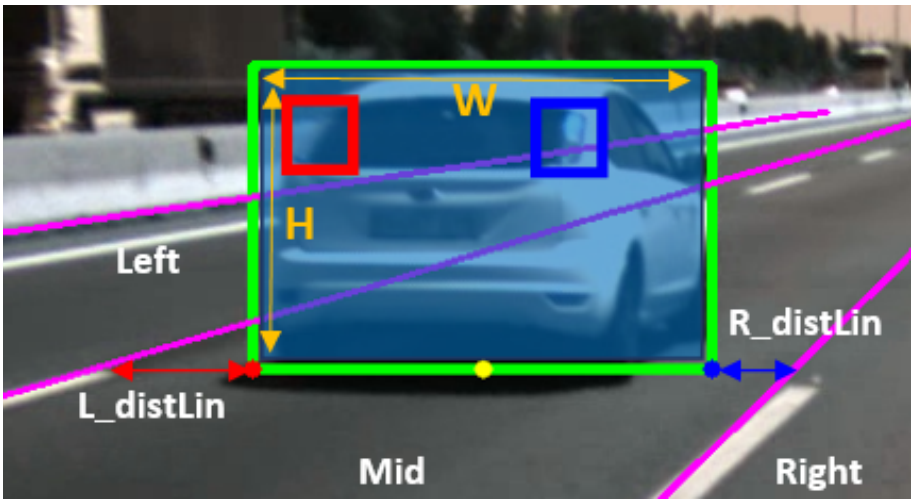


Figure 5.3: Features used as inputs to predict a LC manoeuvre include: red and blue dots representing the bottom-left and bottom-right corners of the TV bounding box (bbox), respectively; red and blue squares indicating the status of the left and right indicator lights of the TV; red and blue arrows showing the distances from the left and right corners of the TV bbox to the highway lane boundaries; pink lines representing the highway lane boundaries; a transparent blue square denoting the surface area of the bbox; and white labels ("Left", "Mid", "Right") indicating lane information.

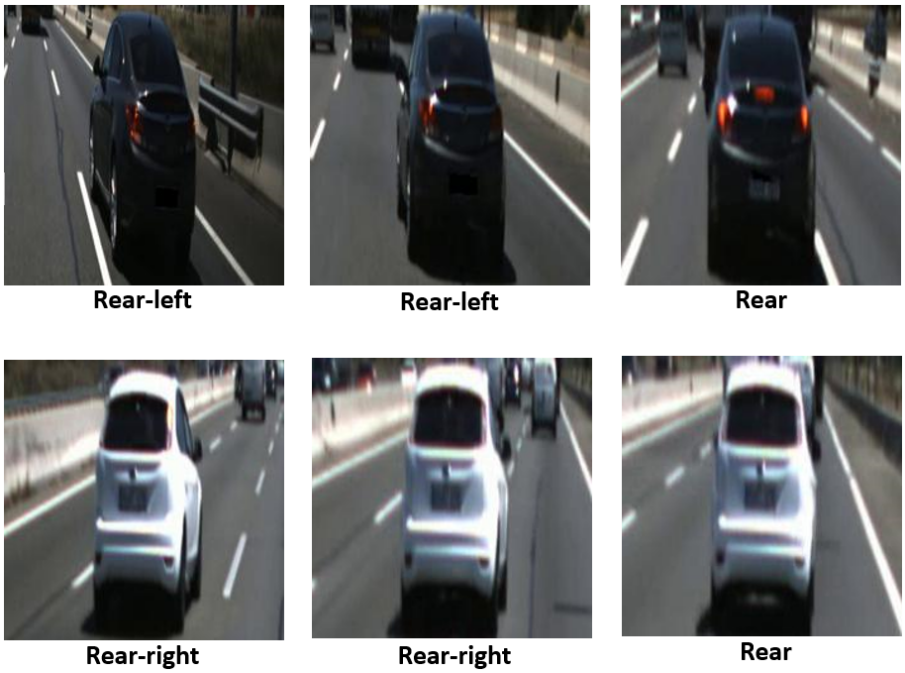


Figure 5.4: The first two figures in the top row are labelled "rear-left" because both the left and rear sides of the TV are visible from the EV's perspective. Similarly, the first two figures in the bottom row are labelled "rear-right," as the right and rear sides of the TV are visible. The remaining images are labelled "rear" since only the rear side of the TV is visible.

of the input features discussed in Section 5.3 to make recognition and predictions.

The input tensor to the system is given by

$$\mathbf{X} \in \mathbb{R}^{B \times T \times d_{\text{in}}}$$

where B is the batch size (number of sequences processed in parallel), e.g., $B = 32$, T is the sequence length (number of time steps per sequence), and $d_{\mathbf{in}}$ is the input feature dimension (number of features per time step), e.g., $d_{\text{in}} = 7$.

The input tensors are initially embedded using a linear transformation to map $d_{\mathbf{in}}$ dimensional input to a hidden dimension expressed by:

$$\mathbf{H}^{(0)} = \mathbf{X} \cdot \mathbf{W}_1^T + \mathbf{b}_1, \tag{5.3}$$

where $\mathbf{W}_1 \in \mathbb{R}^{d_{\text{model}} \times d_{\text{in}}}$ and $\mathbf{b}_1 \in \mathbb{R}^{d_{\text{model}}}$ are the weights and biases of the linear transformation layer, and d_{model} is the dimension of the transformed features (i.e., the hidden size). This results in a transformed tensor $\mathbf{H}^{(0)} \in \mathbb{R}^{B \times T \times d_{\text{model}}}$, which is then passed into an LSTM layer to capture the temporal dependencies:

$$\mathbf{H}^{(1)}, (\mathbf{h}_T, \mathbf{c}_T) = \text{LSTM}(\mathbf{H}^{(0)}), \tag{5.4}$$

where $\mathbf{H}^{(1)}$ is the sequence of hidden states, and $\mathbf{h}_T \in \mathbb{R}^{B \times d_{\text{lstm}}}$ is the final hidden state at the last time step, and $\mathbf{c}_T \in \mathbb{R}^{B \times d_{\text{lstm}}}$ final cell state.

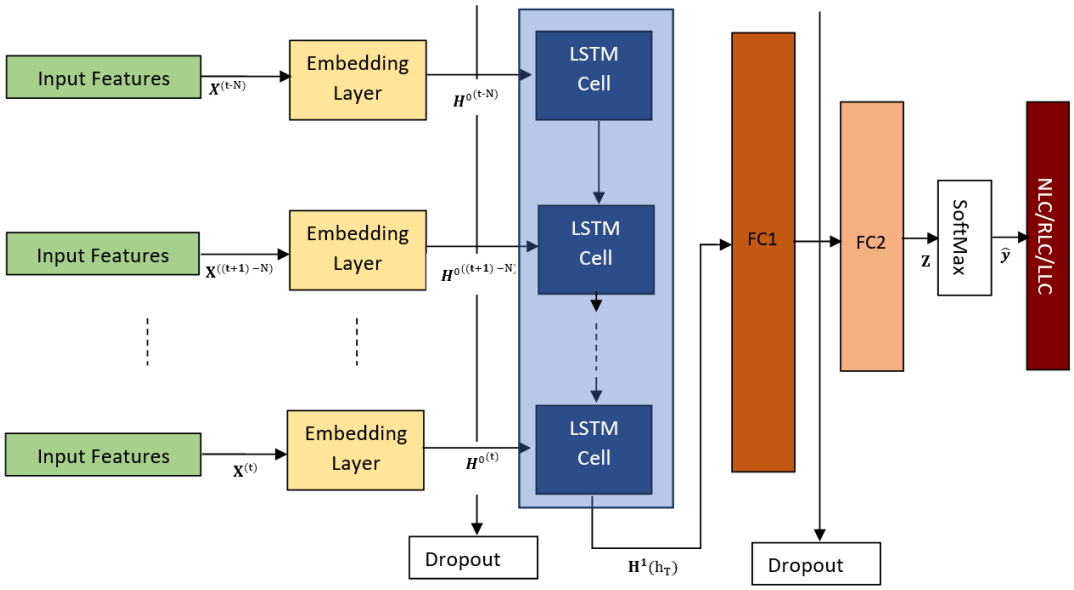


Figure 5.5: The input features are first passed through embedding layers, followed by the application of dropout. The resulting output is then fed into LSTM cells. The output from the LSTM layer is passed through a Fully Connected (FC) layer, after which another dropout is applied. Finally, the data flows through a second FC layer, and a SoftMax function is used to classify the manoeuvre.

A basic LSTM cell unit is illustrated in Figure 5.6, containing three inputs: the previous cell state $c^{<t-1>}$, previous hidden state $h^{<t-1>}$, and the current transformed input sequence $\mathbf{H}^{(0)} \in \mathbb{R}^{B \times T \times d_{\text{model}}}$. Additionally, three outputs: the current hidden state $h^{<t>}$, current cell state $c^{<t>}$, and the current prediction output $\hat{y}^{<t>}$ [276]. The previous hidden state and the current input sequence are used by the forget gate

$f^{<t>}$, the update gate $i^{<t>}$, the candidate function $\tilde{c}^{<t>}$, and the output gate $o^{<t>}$. The forget gate is expressed as,

$$f_t = \sigma(W_{if}\mathbf{H}_t^{(0)} + b_{if} + W_{hf}h_{t-1} + b_{hf}) \tag{5.5}$$

and it decides if the information from the previous cell state should be kept or discarded. The candidate function is expressed as,

$$\tilde{c}^{<t>} = tanh(W_{ic}\mathbf{H}_t^{(0)} + b_{ic} + W_{hc}h_{t-1} + b_{hc}) \tag{5.6}$$

it uses a hyperbolic tangent activation function ($tanh$) to propose new information for the current cell state. The update gate is expressed as,

$$i_t = \sigma(W_{ii}\mathbf{H}_t^{(0)} + b_{ii} + W_{hi}h_{t-1} + b_{hi}) \tag{5.7}$$

decide if the newly generated information should be considered for the cell state. The output gate is expressed as,

$$o_t = \sigma(W_{io}\mathbf{H}_t^{(0)} + b_{io} + W_{ho}h_{t-1} + b_{ho}) \tag{5.8}$$

it determines what information from the previous hidden state, current input sequence, and the current cell state should be used as the output of the current LSTM cell unit. Using the previous equations, the current cell state c^t is given by the following equation,

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}^{<t>} \tag{5.9}$$

$\odot$ is a Hadamard product, and the current hidden state h_t is given by

$$h_t = o_t \odot tanh(c_t). \tag{5.10}$$

The hidden output of the last LSTM cell unit is then fed into two FC layers, to learn non-linear combinations of the extracted features from the previous LSTM cell unit, given by,

$$\mathbf{z} = \mathbf{h}_T \cdot \mathbf{W}_2^T + \mathbf{b}_2, \tag{5.11}$$

where $\mathbf{W}_2 \in \mathbb{R}^{3\times d_{\text{lstm}}}$, and $\mathbf{b}_2 \in \mathbb{R}^3$ are the weights and the biases of the FC layers. The process of the FC operation is followed by a SoftMax activation function to yield the final class probabilities, which are expressed by:

$$\hat{\mathbf{y}} = \text{SoftMax}(\mathbf{z}), \tag{5.12}$$

where $\hat{\mathbf{y}} \in \mathbb{R}^{B\times 3}$ represents the predicted probabilities over the output classes. This architecture allows the model to learn temporal patterns from the sequence of features and classify each instance into one of three predefined categories.

The neural network model in this study employs two key activation functions: **Leaky ReLU** and **Softmax**. Leaky ReLU is used in the hidden layers to address the "dying ReLU" problem. Unlike the standard ReLU, which outputs zero for all negative inputs, Leaky ReLU allows a small, non-zero gradient when the input is negative. This helps the model continue learning even when some neurons receive

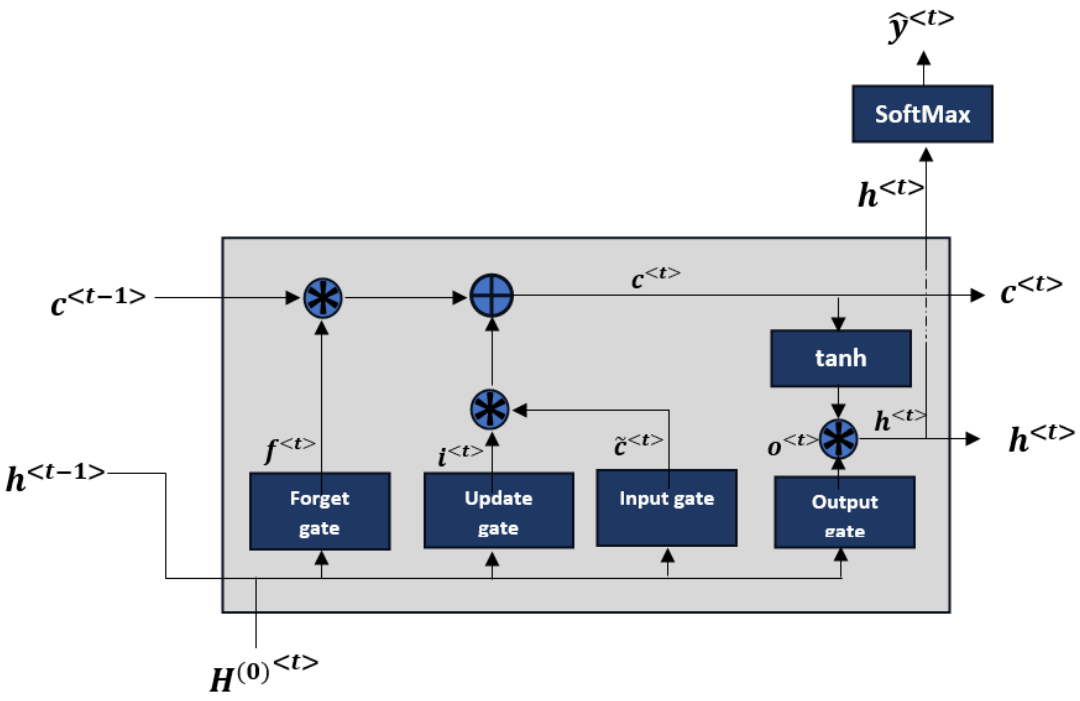


Figure 5.6: LSTM basic unit.

negative values, making it more robust in deep networks, especially when working with datasets that include negative inputs[277].

The **Softmax** function is applied in the output layer for multi-class classification. It transforms raw output scores into probabilities that sum to 1, making it ideal for tasks where each input must be classified into one of several categories [277].

Both Leaky ReLU and Softmax are widely used in machine learning applications. Leaky ReLU helps mitigate the vanishing gradient issue, while Softmax is a standard choice for multi-class classification problems. Their modular nature also enhances the adaptability of the model, making it suitable for a wide range of domains beyond the specific problem addressed in this thesis.

5.3 Experimental Evaluation

This section outlines the dataset splitting strategy, the combinations of input features evaluated, the different OH values explored, the network hyperparameters that yielded the best results, and the hardware used to conduct the experiments.

5.3.1 Dataset

The dataset used in this study is derived from the PREVENTION dataset. A summary of its characteristics is provided in Table 5.1. The data was split into 80% for training, 10% for validation, and 10% for testing. Importantly, the split was applied at the video clip level rather than at the individual sequence level to prevent overfitting. This ensures that sequences from the same video do not appear across the training, validation, and testing sets. Key observations about the dataset include:

- The video clips vary in length—some contain more than 50 frames, while others have fewer than 30. As a result, the number of sequence samples per video varies.
- All features were pre-processed by normalising them using the mean and standard deviation.
- As shown in Table 5.1, the number of NLC sequences is significantly higher than those of the other classes, despite a similar number of video clips. This is because LLC and RLC videos also include NLC frames before and after the LC manoeuvre.
- Only videos with at least 19 observation frames prior to the start of the LC manoeuvre were included in the dataset.

Table 5.1: Main statistics information for the dataset before and after splitting.

	ALL	**NLC**	**LLC**	**RLC**
Video Clips	494	171	160	163
Training Videos	399	137	127	135
Validating Videos	43	16	16	11
Testing Videos	52	18	17	17
Sequences	32385	14789	8493	9103
Training Sequences	26318	11927	6824	7567
Validating Sequences	2658	1161	893	604
Testing Sequences	3409	1701	776	932

Note: Non-Lane Change (NLC), Left Lane Change (LLC), and Right Lane Change (RLC).

5.3.2 Parameters

This subsection describes the different input feature combinations used, the OH values explored, and the hyperparameters selected after fine-tuning the model.

Input Features

The VIP_LSTM algorithm was trained, validated, and tested using various combinations of input features to evaluate their impact on recognition and prediction performance. The network was trained with the following feature sets:

- Position only (P).
- Position and speed (P+S).
- Position, speed, and indicator light status (P+S+I).
- Position, speed, and the distance from the bottom corners of the TV's bounding box to the highway lane boundaries (P+S+LiDist).
- Position, speed, indicator light status, and lane boundary distances (P+S+I+LiDist).

- Position, speed, indicator light status, lane boundary distances, and lane identification (P+S+I+LiDist+LanInfo).

- Position, speed, indicator light status, lane boundary distances, lane identification, and the visible side of the TV (P+S+I+LiDist+LanInfo+VehAngl).

Observation Horizon Time

When using the full feature set (P+S+I+LiDist+LanInfo+VehAngl), the OH was varied from 5 frames (0.5 seconds) to 19 frames (1.9 seconds) to assess its effect on model performance. The OH value that yielded the best results was then used consistently in all subsequent experiments.

Model Hyperparameters

The VIP_LSTM model underwent fine-tuning, and the final hyperparameters are reported in Table 5.2.

Table 5.2: Parameters used to train the VIP_LSTM network.

Parameter	**Value**
Random Seed	469
Input Sequence (N)	5-19 frames (05s - 1.9s)
Embedding Layer	256
Dropout1	0.02
LSTM hidden Size	128
LSTM nu. of Layers	1
FC1	128
Dropout2	0.71
FC2	3
LeakyRelu negative slope	0.1
Batch Size	64
Num Epochs	20
Optimiser	Adam
Loss criterion	Cross-Entropy Loss
Learning rate	0.000029
LR step size	5
LR gamma	0.1
Clip Grad.	5
Weight Decay.	0.001

5.3.3 Hardware

All the experiments were executed using an Intel(R) Core(TM) i7-1075 CPU @ 2.60GHz and an NVIDIA GeForce GTX 1080 Ti.

5.3.4 Metrics

The LC manoeuvre recognition model's performance was evaluated by calculating the accuracy score between the predicted results $\hat{y}$ and the ground truth values y, which is expressed as follows:

$$accuracy(y, \hat{y}) = 1/n_{samples} \sum_{i=0}^{n_{samples}-1} 1(\hat{y}_i = y_i). \tag{5.13}$$

Unlike other works that have calculated the accuracy score only from the F0 to F1 stage, this study also calculates accuracy scores between the F to F0 and F1 to F2 stages. The accuracy between F and F1 measures how well the algorithm continues to correctly recognise the remaining frames after the first correct recognition until the LCE.

The proposed and employed metric to measure PHT performance is the PHTR, which is expressed as:

$$PHTR = (F1 - F)100/(F1 - F0), \tag{5.14}$$

where F is the frame where the first correct LC recognition is made between the F0 and LCE stages. This metric is used to avoid the weighting effects between longer and shorter manoeuvres [121].

In summary, the following metrics are used to measure recognition and prediction performance:

- **Acc-F0toF2:** Accuracy score from the F0 to F2 stage, i.e., across all frames in a given video sample.
- **Acc-FtoF1:** Accuracy score from the first correct prediction (F) to the F1 stage.
- **Acc-F0toF1:** Accuracy score from the F0 to F1 stage.
- **Acc-F1toF2:** Accuracy score from the F1 to F2 stage.
- **PHTR:** Prediction Horizon Time Ratio.

It is important to note that NLC video samples are not included in the calculations for Acc-FtoF1, Acc-F0toF1, Acc-F1toF2, and PHTR values.

5.4 Results and Discussions

The proposed algorithm's performance was first analysed in terms of its ability to classify different LC manoeuvre classes. Next, its capability to predict LC manoeuvres was evaluated. Finally, the results were compared with those from relevant studies in the literature. As a baseline for comparison, the first row of Table 5.3 shows the human performance in classifying and predicting LC manoeuvres, as reported by [121].

Table 5.3: Accuracy and PHTR results for the different types of input features and different LC manoeuvre stages. OH set as 1.9s

Input Features	Acc(%) F0toF2	Acc(%) F0toF1	Acc(%) FtoF1	Acc(%) F1toF2	PHTR
Human Performance	83.9	-	-	-	1.66s
P	71.5	70.1	73.7	70.7	0.920
P+S	74.6	73.3	78.7	86.5	0.870
P+S+I	82.1	84.7	88.0	77.6	0.928
P+S+LiDist	79.6	78.6	81.7	86.1	0.888
P+S+I+LiDist	85.8	88.7	90.6	88.2	**0.947**
P+S+I+LiDist+LanInfo	85.3	**88.8**	**91.0**	87.6	0.946
P+S+I+LiDist+LanInfo+VehAngl	**86.4**	87.8	90.0	**89.2**	0.944

5.4.1 Lane Change Recognition Results

The LC recognition results using different combinations of input features are reported in Table 5.3, and the following observations can be made:

- In most scenarios, the recognition accuracy increases with the inclusion of more input features. For example, when using only the "position" feature, the algorithm achieved an Acc-F0toF2 value of 71.5%. However, with the P+S+I+LiDist+LanInfo+VehAngl feature set, the accuracy improved to 86.4%. The lower accuracy scores observed with P, P+S, and P+S+LiDist are due to these features being more related to the TV's motion. As shown in Figure 5.1, the most significant displacement during an LC manoeuvre typically occurs in the frames just before and after the LCE. Consequently, the algorithm may misclassify portions of the manoeuvre where there is minimal movement.

- The Acc-FtoF1 values are higher than the Acc-F0toF2 values, indicating that recognising the initial frames of an LC manoeuvre is more challenging.

- The Acc-F1toF2 values improve when speed and lane distance features are included. For example, using P+S, P+S+LiDist, P+S+I+LiDist, P+S+I+LiDist+LanInfo, and P+S+I+LiDist+LanInfo+VehAngl yields Acc-F1toF2 values of 86.5%, 86.1%, 88.2%, 87.6%, and 89.2%, respectively. The limited improvement with P+S+I is likely due to the algorithm's reliance on indicator lights, which are often deactivated in the final frames before the end of the LC manoeuvre or may be incorrectly activated. The best result is achieved with P+S+I+LiDist+LanInfo+VehAngl, as the algorithm benefits from integrating indicator status, lane distance, lane information, and the TV's visible side.

- The addition of the TV's visible side feature improves only the Acc-F0toF2 and Acc-F1toF2 values, suggesting that this feature helps the algorithm determine when the TV is realigning within the highway lane.

- Although the P+S+I+LiDist+LanInfo+VehAngl feature set yields the best Acc-F0toF2 and Acc-F1toF2 values, the P+S+I+LiDist+LanInfo combination performs best for Acc-F0toF1 and Acc-FtoF1. In the context of LC manoeuvre recognition, achieving higher performance during the F0 to F1 and F to F1 stages is more critical, as these stages mark the initiation of the manoeuvre—information that is essential for the EV to make timely decisions. Furthermore, while P+S+I+LiDist provides the best PHTR value, it is only marginally better (by 0.001) than P+S+I+LiDist+LanInfo.

- The proposed algorithm outperforms human performance in recognising LC manoeuvres by 2.5%.

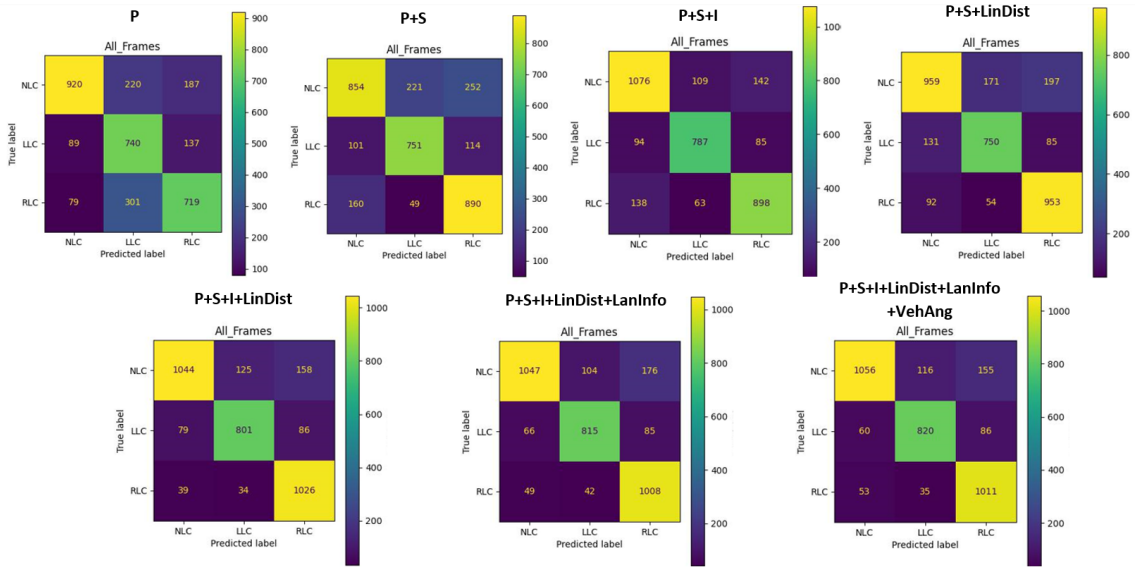


Figure 5.7: Confusion matrix for the different combinations of input features.

The confusion matrix depicted in Fig. 5.7 reveals the following:

- When using only the position feature, the algorithm frequently confuses NLC with LLC and RLC, and RLC with LLC. Additionally, it sometimes misclassifies LLC as RLC.
- Incorporating the speed feature improves the classification of LLC and RLC samples. However, it also increases the misclassification of NLC samples as LLC and RLC, and vice versa.
- Adding the indicator light status enhances the algorithm's ability to classify all samples, particularly NLC. Nonetheless, a considerable number of NLC samples are still misclassified as LLC and RLC, and many RLC samples are misclassified as NLC and LLC.
- Introducing the LiDist feature significantly improves the classification of RLC samples and slightly improves the classification of other samples. However, it reduces the algorithm's ability to distinguish LLC from NLC.
- Combining the P+S+I+LiDist features significantly enhances the algorithm's performance, especially in distinguishing LLC and RLC samples. However, it still struggles with NLC samples, misclassifying some as LLC and RLC.
- Including the LanInfo feature slightly improves the classification of NLC and LLC samples but increases the misclassification of RLC samples.
- Finally, the addition of the VehAngl feature results in notable improvements in correctly classifying all samples compared to the P+S+I+LiDist+LanInfo feature set.

Table 5.4 and Figure 5.8 present the LC recognition results for varying OHT values. It is evident that, in most cases, recognition accuracy improves as the OHT increases.

Table 5.4: Accuracy and PHTR results when varying the OHT values from 5 (0.5 seconds) to 19 frames (1.9 seconds)

OHT	Acc F0toF2(%)	Acc F0toF1(%)	Acc FtoF1(%)	Acc F1toF2(%)	PHTR
5	78.9	71.0	74.8	78.6	0.917
6	78.7	70.5	74.8	79.2	0.908
7	80.1	74.2	77.7	80.0	0.919
8	81.1	74.7	80.5	80.6	0.885
9	82.1	75.8	81.4	81.6	0.893
10	82.9	81.8	84.9	84.7	0.929
11	82.8	81.0	84.7	83.4	0.919
12	83.3	84.0	87.5	86.0	0.923
13	83.3	82.9	87.9	84.2	0.904
14	83.7	82.5	85.8	83.6	0.926
15	84.0	84.8	87.8	86.0	0.930
16	84.5	84.3	87.5	85.6	0.927
17	85.7	86.9	89.2	87.2	0.944
18	86.4	86.2	88.1	87.4	**0.947**
19	**86.4**	**87.8**	**90.0**	**89.2**	0.944

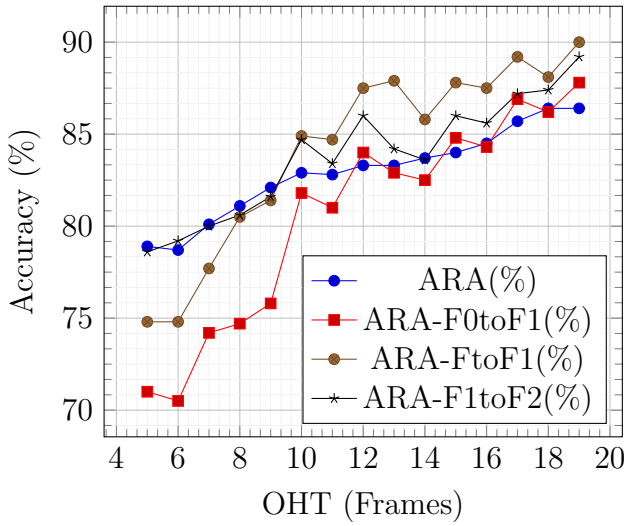


Figure 5.8: Relationship between OHT and LC recognition accuracy score.

Table 5.5 presents the recognition accuracy values obtained using GRU and LSTM networks. The results indicate that LSTM outperformed GRU in recognising LC manoeuvres at various stages. This superior performance may be attributed to LSTM's enhanced capability to learn long-term dependencies.

Table 5.5: Recognition accuracy and PHTR achievement when using LSTM or GRU networks.

Framework	Acc(%) F0toF2	Acc(%) F0toF1	Acc(%) FtoF1	Acc(%) F1toF2	PHTR
GRU	85.6	88.4	89.6	88.2	0.957
LSTM	86.4	87.8	90	89.2	0.944

5.4.2 Lane Change Prediction Results

By analysing Table 5.3, Table 5.4, and Fig. 5.9, the following observations were made regarding the PHTR results:

- An ideal system would achieve a PHTR value of 1.0 and an Acc-FtoF1 value of 100%, indicating that all frames between F0 and F1 are correctly recognised. The feature combination that achieved the best results was P+S+I+LiDist, with a PHTR value of 0.946 and an Acc-FtoF1 value of 91.0%.
- The highest PHTR values were obtained when the indicator light status was included as a feature. For instance, using the feature sets P+S+I, P+S+I+LiDist, P+S+I+LiDist+LanInfo, and P+S+I+LiDist+LanInfo+VehAng resulted in PHTR values of 0.928, 0.947, 0.946, and 0.944, respectively. This underscores the importance of indicator status in predicting LC behaviour, as drivers typically activate indicators to signal their intentions before executing a manoeuvre.
- Although there is some fluctuation in PHTR values as OHT increases, a clear trend emerges: PHTR generally improves with longer OHs.

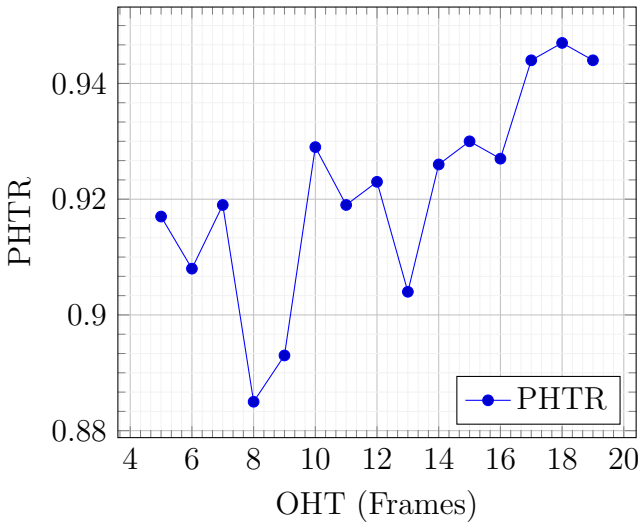


Figure 5.9: Relationship between OHT and PHTR.

Table 5.5 reports the PHTR values obtained using GRU and LSTM networks. The results reveal that GRU outperformed LSTM in predicting LC manoeuvres. One possible explanation for GRU's superior performance is its ability to capture short-term dependencies, which is particularly relevant when predicting LC manoeuvres. For instance, in a sequence of features with an OHT of 20 frames, the most informative frames indicating the beginning of a LC are likely the last 5 frames.

On the other hand, accurately recognising an LC manoeuvre once it has been initiated requires an algorithm capable of capturing long-term dependencies, which is where LSTM excels.

Both batch and online inference tests were conducted for the proposed algorithm. The batch inference time was 0.01 seconds, while the online inference time was 0.69 seconds.

5.4.3 Comparison With Other Works

A direct comparison between the proposed work and existing studies is not feasible due to differences in datasets, methodologies, and evaluation metrics. However, this section briefly presents the results reported by other works and discusses how they differ from the proposed approach.

Authors [69, 120] reported classification and LC prediction accuracies of up to 90.98% and 91.94%, respectively. However:

- They only considered two classes: LC and lane keeping, whereas this work distinguishes between LLC, RLC, and NLC.
- They used 3,110 NLC, 342 LLC, and 430 RLC sequences. In contrast, this work used a significantly larger number of samples, as shown in Table 5.1.
- Their dataset is heavily imbalanced, with a much higher number of NLC events.
- They used a minimum OH of 20 frames (2 seconds), while this work used 19 frames (1.9 seconds). Using fewer frames reduces computational requirements.
- Their models used raw images as input, which demands significantly more computational power. In contrast, the proposed approach uses manually extracted features such as position, velocity, distance, and discrete vehicle signals, which are computationally more efficient.
- They fixed the prediction horizon time (PHT) at 1 or 2 seconds. In contrast, this work does not fix a PHT value. Instead, it evaluates prediction performance based on the first correct recognition and the recognition accuracy. This approach is more realistic, as LC manoeuvres vary in duration, and a correct prediction 1 or 2 seconds before the LCE may still be followed by incorrect classifications in subsequent frames.

Author [121] reported a classification accuracy of 86.9% and a manoeuvre prediction length of 83.3%. They used the term "anticipation" instead of "prediction," and their work is the most closely related to the proposed approach. They normalised the manoeuvre duration and calculated the percentage of correctly classified frames. However, they did not evaluate classification accuracy for frames after the prediction period or following the LCE.

5.5 Conclusions and Future Works

This chapter introduced an LSTM-based algorithm for recognising and predicting various types of LC manoeuvres in highway scenarios, including LLCs, RLCs, and NLC, from an onboard perspective. Specifically, this study manually extracted different motion and contextual input features to investigate their impact on the recognition and PHTR of LC manoeuvres.

The motion features used include position, speed, and bounding box surface area, while the contextual features include the TV's indicator light status, the TV's visible side from the EV's perspective, the distance between the TV's bounding

box corners and the highway lane boundaries, and highway lane identification. It is worth noting that previous works predominantly relied on top-view datasets. In cases where onboard datasets were used, they typically relied on raw images and CNNs to implicitly extract these features.

The main advantages of the proposed approach are twofold: (1) the use of onboard datasets aligns more closely with the real-world context of an IV system, and (2) manually extracted features require significantly less computational power and time compared to deep learning techniques that learn features from raw image data. Additionally, this approach enables a detailed investigation into the individual contribution of each feature to the final results.

Moreover, a novel evaluation metric—PHTR—was introduced to accommodate LC manoeuvres of varying lengths. Beyond assessing PHT, this work also evaluates how well the algorithm continues to recognise the remaining frames of an LC manoeuvre after the first correct prediction.

This study concludes that:

- The indicator light status is a crucial feature for effective LC recognition and prediction. However, it must be combined with other features, as some drivers may not use their indicators correctly, and in some cases, the TV's indicator lights remain on after completing the manoeuvre.

- Combining indicator light status with the distance between the TV and the highway lane boundaries yields notable improvements in recognition accuracy.

- Despite the promising results achieved with the P+S+I+LiDist feature set, challenges remain. For example, several NLC samples were misclassified as LLC or RLC, and vice versa.

- Using only manually extracted features, the algorithm achieved a recognition accuracy of 86.4%, outperforming the human benchmark of 83.9%. In terms of prediction, the algorithm achieved a PHTR of 0.946.

- LSTM achieved the best recognition accuracy, while GRU achieved the best PHTR value.

- By identifying the most relevant features for recognising and predicting LC manoeuvres, this research provides valuable guidance for future studies. For instance, given the importance of indicator light status and lane distance, future work could explore CNNs with attention mechanisms tailored to these features. Additionally, the significance of the TV's visible side suggests that algorithms for classifying vehicle visibility from the EV's perspective could be developed.

The following directions are suggested for future research:

- Incorporate surrounding vehicle information to assess its impact on LC manoeuvre recognition and prediction.

- Investigate methods for automatically detecting the TV among the surrounding traffic. In this study, the TV was manually labelled, but in real-world applications, the IV should be capable of identifying the TV autonomously.

- Explore the use of images from cameras installed in different vehicle locations, such as rear and side cameras. As mentioned in Section 5.3, only samples with at least 19 frames of OH were considered. However, some TVs initiated LC manoeuvres while not visible to the EV, as they were located behind it. Including a rear camera would enable the algorithm to account for such scenarios.

Chapter 6

Potential Traffic Hazard Event Recognition System

This chapter investigates the performance of various proposed machine learning model architectures in recognising different potential traffic hazard behaviours , using the novel dataset introduced in Chapter 3. The goal is to identify the most effective machine learning strategies for recognising traffic agent behaviours in complex traffic scenes by conducting a series of experiments. These experiments explore different types of input features, observation horizon time (OHT) s, and varying numbers of potential traffic hazard classes. This analysis contributes to understanding why current algorithms struggle to recognise traffic agent behaviour in complex environments and highlights the strengths and limitations of the proposed dataset.

The chapter is organised as follows: Section 6.1 formulates the traffic agent behaviour recognition problem. Section 6.2 explores the types of input features used in the literature and those adopted in this thesis. Section 6.3 presents the proposed machine learning models for recognising potential traffic hazard behaviours. Section 6.4 details the experimental setup. Section 6.5 reports and discusses the results. Section 6.5.7 presents an ablation study. Finally, Section 6.6 summarises the conclusions and outlines potential directions for future work.

6.1 Problem Formulation

In complex traffic scenarios , both static and dynamic traffic agents can exhibit a variety of behaviours. As illustrated in Fig. 6.1, a traffic agent's behaviour consists of two main stages: the starting point (F0) and the endpoint (F1). The starting point marks the moment when the traffic agent begins to perform a specific behaviour, while the endpoint corresponds to the moment the behaviour concludes.

Based on these stages, an AI-based traffic agent behaviour recognition system can be defined as a model's ability to correctly identify the frames in which a specific behaviour occurs—from its initiation at F0 to its conclusion at F1.

Since traffic agent behaviour falls into multiple categories, this study treats behaviour recognition as a classification problem. In this context, recognition refers to the algorithm's ability to classify the specific behaviour being executed by the traffic agent.

The behaviour recognition problem is formulated as follows: a series of feature vectors $\{F_{t-OH}, ..., F_t\}$, extracted from a consecutive sequence of video frames

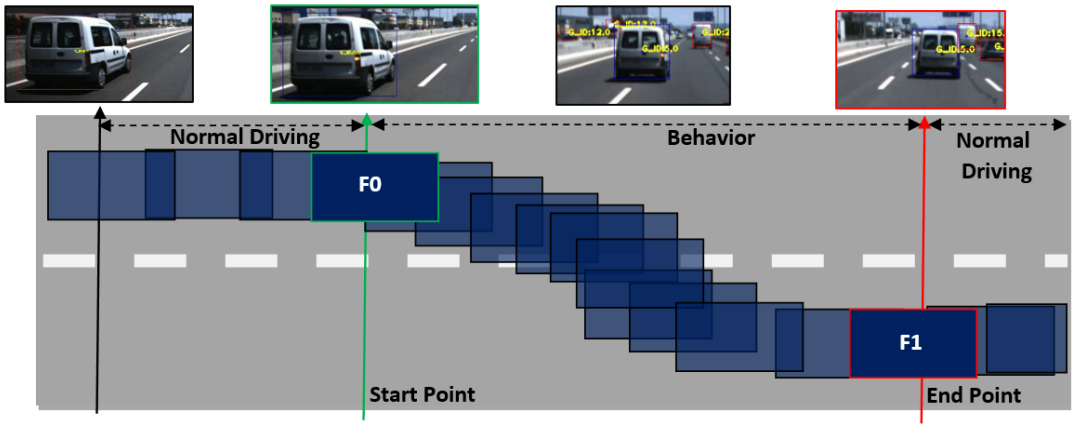


Figure 6.1: The frames belonging to a lane change behaviour are between the F0 to the F1 stage.

$\{t-OH, ..., t\}$ obtained from an image sensor, is used by the proposed model to determine the type of behaviour the traffic agent is performing. This can be expressed as:

$$Behaviour_a^{t+n} \epsilon \{0, n+1\}, \tag{6.1}$$

where t is the timestamp of the last observed frame, n denotes the next frame after the previously observed frame, and $0, n+1$ represents the different behaviour categories, respectively. The features used in this work are described in Section 6.2. Fig. 6.2 illustrates the process of applying the behaviour recognition model to the frames associated with the behaviour event.

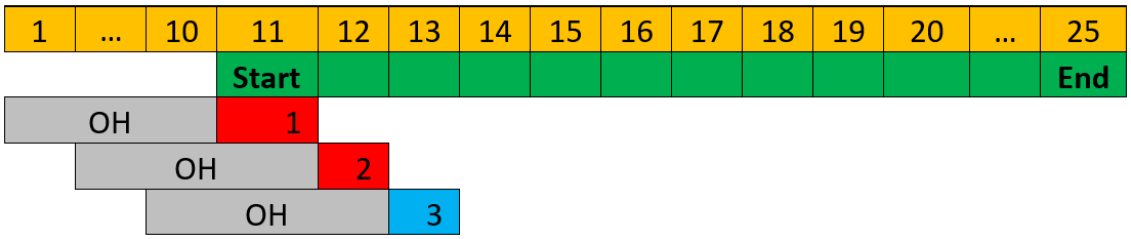


Figure 6.2: The orange cells represent the frames of a video sample, while the green cells indicate the frames in which the traffic agent behaviour event occurs. The red and blue cells denote an incorrect and a correct recognition, respectively. The grey cells correspond to the frames used as the OH. The recognition OH frames are employed to predict the behaviour in the subsequent frame (cell 11). Thereafter, the OH is shifted forward by one frame to recognise the next frame (cell 12). This process continues iteratively until the final frame of the current event is reached.

6.2 Inputs

Previous research has predominantly utilised object type, position, and the velocity of the centre coordinates of the target objects' bounding boxes as explicit input features for recognising traffic agent behaviours. These features are referred to in

this work as *conventional features*. Additionally, various representations of raw images—such as images without target objects, images with lane information, images of the bounding box, and optical flow images—have been employed as implicit features.

However, other types of explicit and implicit input features, such as the vehicle's indicator light status, the side of the target object observed by the EV, blended image representations, and OOI image representations, have not been considered, as no publicly available datasets provide them.

This work considers the following explicit features as input: x and y position and speed, the width and height of the target object's bounding box, the side of the target object visible to the EV, and the target vehicle's rear light status.

The following implicit features are also considered as input in this work: raw images, images without target objects, depth information images, blended images, images with lane information, object-of-interest images, local context images, motion-based context images, and local context blended images.

6.3 Proposed Model Architectures

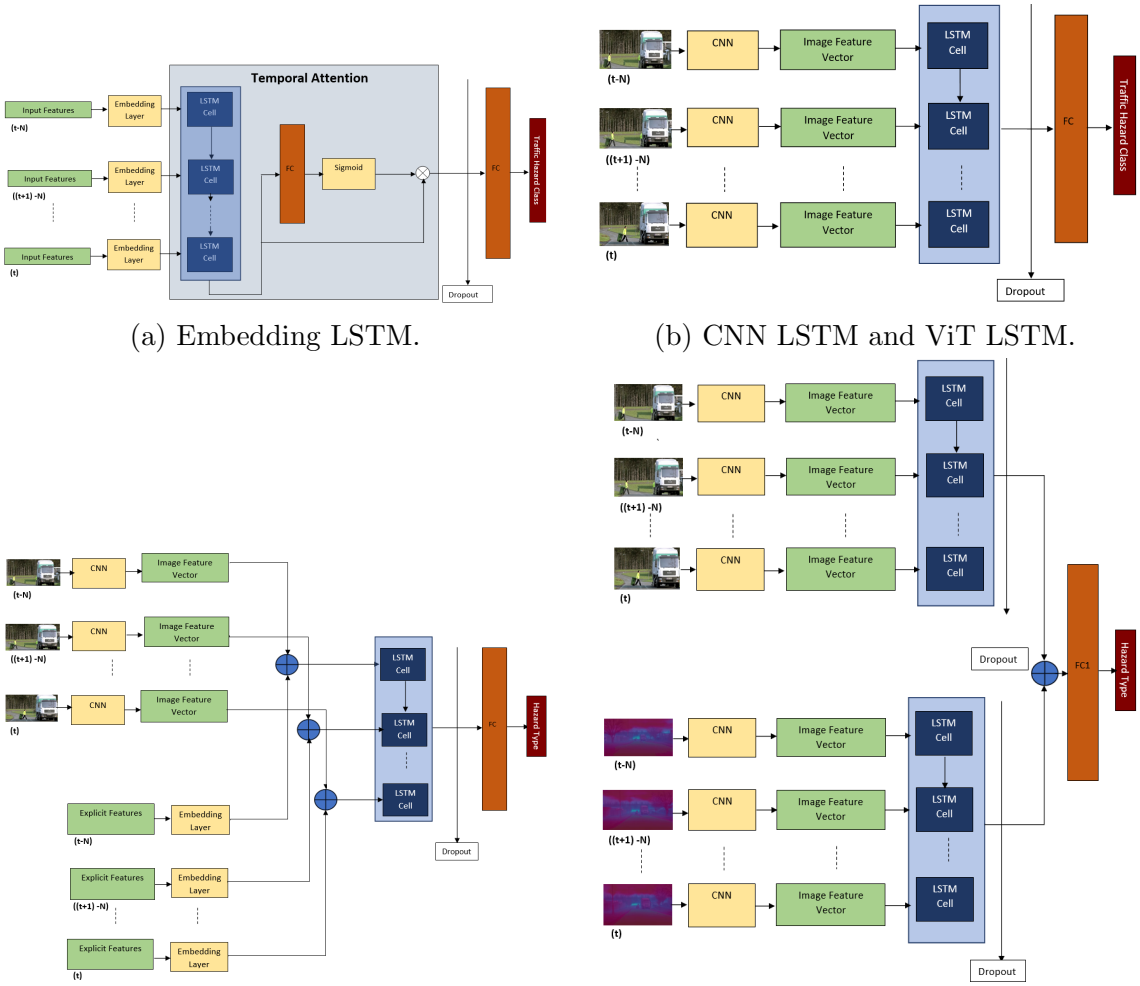
As discussed in Chapter 2, various models have been explored to study traffic agent behaviour, including both time series and non-time series approaches. Time series models, such as Recurrent Neural Networks (RNNs)—including Gated Recurrent Units (GRUs) and Long Short-Term Memory (LSTM) networks—as well as Transformer-based architectures, are commonly employed due to their ability to capture temporal dependencies. In contrast, non-time series models include Convolutional Neural Networks (CNNs), graph-based models, probabilistic models (e.g., Bayesian Networks), and Reinforcement Learning.

Graph-based models are well-suited to modelling spatial and relational dependencies between multiple traffic agents, but they require complex graph construction processes and are less effective at capturing temporal dynamics unless combined with temporal extensions. Probabilistic models offer interpretable outputs and can explicitly model uncertainty; however, they often require substantial prior domain knowledge and become computationally expensive when applied to large and complex datasets. Convolutional models are effective at extracting spatial features and can be extended to 3D CNNs to capture temporal patterns, although this significantly increases computational cost. Reinforcement Learning approaches are adaptive and capable of improving with experience, but they typically require large volumes of data and considerable computational resources to train effectively.

Despite the strengths of these alternative approaches, a key limitation they share is the absence of a natural mechanism for learning temporal dependencies. Since potential traffic hazard recognition is inherently a temporal problem—where traffic scenes evolve sequentially over time—time series models are more appropriate for this task. As supported by studies referenced in Chapter 2, time series models consistently outperform non-temporal models in tasks involving sequential prediction. For this reason, this study adopts time series models, as well as hybrid models that combine time series and non-time series components, to address the challenge of traffic hazard recognition.

Several types of machine learning architectures were implemented in this work, including Artificial Neural Networks (ANNs), CNNs, Vision Transformer Networks

(ViT)s, and LSTM networks. As illustrated in Fig. 6.3, these architectures were used to develop multiple model variants, including an embedded Temporal Attention (TA) LSTM model (Figure 6.3a), a standalone CNN-LSTM model (Figure 6.3b), a ViT-LSTM hybrid model (Figure 6.3c), a combined embedded and single-stream CNN-LSTM model, and a multi-stream CNN-LSTM model (Figure 6.3d).



(a) Embedding LSTM.

(b) CNN LSTM and ViT LSTM.

(c) Embedding CNN LSTM.

(d) Multi-stream CNN LSTM.

Figure 6.3: .

6.3.1 Embedding LSTM Model

The Embedding LSTM model is depicted in Fig. 6.3a. It is a conventional vanilla LSTM enhanced with a TA mechanism, utilising the previously mentioned explicit features as input. Each sequence of input features is initially embedded using a linear transformation to obtain a more expressive representation of the inputs. This process is described by the following equation:

$$\mathbf{H}^{(0)} = \mathbf{X} \cdot \mathbf{W}_1^T + \mathbf{b}_1, \tag{6.2}$$

Subsequently, each embedded feature sequence is fed into an LSTM unit, where the hidden output of one unit is passed to the next in the sequence. The hidden output of the final LSTM unit is then passed through FC layers followed by a Sigmoid

activation function to learn TA weights. These attention weights are multiplied by the LSTM outputs, enabling the model to focus on the most relevant features of the input sequence at different time steps. Finally, the last hidden output of the final LSTM unit is selected and used as input to an FC layer to learn non-linear combinations of the extracted features.

6.3.2 CNN LSTM Model

The CNN-LSTM model, depicted in Fig. 6.3b, comprises a CNN that extracts feature vector representations from the input sequence of images. These feature vectors are subsequently input into an LSTM network to capture temporal dependencies. The output of the LSTM network is passed through a dropout layer and then fed into a FC layer. Finally, the resulting feature vector from the FC layers is processed using an `argMax` function.

6.3.3 ViT LSTM Model

The ViT, introduced by [278], is based on the standard Transformer architecture [279]. The model was adapted to process 2D images by reshaping the input image $x \in \mathbb{R}^{H \times W \times C}$ into a sequence of flattened 2D patches $x_p \in \mathbb{R}^{N \times (P^2 \cdot C)}$, where $H \times W$ denotes the image resolution, C is the number of channels, and $N = \frac{HW}{P^2}$ is the number of patches. These patches are projected into a latent dimension D via a learnable linear layer, resulting in patch embeddings. A learnable class embedding is prepended to this sequence, and its output from the Transformer encoder represents the image. During pre-training, a classification Multi-layer Perceptron (MLP) head is used, while a single linear layer is employed during fine-tuning. Positional information is preserved using 1D learnable position embeddings. The Transformer encoder alternates between multi-headed self-attention (MSA) and MLP blocks, with LayerNorm applied before and residual connections after each block. Owing to the self-attention mechanism, ViT can learn global relationships, capturing long-range dependencies within an image.

However, the standard ViT is not designed to process sequences of images. To address this, [280] adopted the Video Vision Transformer (ViViT), which extends ViT to handle video data by processing image sequences. Nevertheless, ViViT demands significant computational resources. In this work, a hybrid model is proposed that combines ViT with LSTM. The MLP layers from the ViT architecture are removed, and the sequence of features generated by the Transformer is used as input to an LSTM network to capture temporal dependencies. The model architecture is illustrated in Fig. 6.3b; it follows the structure of a CNN-LSTM model but replaces the CNN with ViT as the feature extractor.

6.3.4 Embedding CNN LSTM model

The Embedding CNN-LSTM model, shown in Fig. 6.3c, integrates the Embedding LSTM and CNN-LSTM models. Feature vectors from the CNN and embedding layers are each fed into separate LSTM networks to capture temporal dependencies. The outputs of these LSTM networks are concatenated, passed through a dropout layer for regularisation, and then processed by a FC layer. Finally, the output vector

from the FC layer is passed through an `argMax` function to produce the model's final prediction.

6.3.5 Multi Stream CNN LSTM

The multi-stream CNN-LSTM model uses two parallel CNN streams to extract information from different types of input images. For example, in Fig. 6.3d, the top CNN stream processes a sequence of raw images, while the bottom stream uses a sequence of depth estimation images derived from the raw inputs. Each CNN stream includes its own LSTM network to capture temporal dependencies. The output from each LSTM is passed through a dropout layer before being concatenated. The concatenated feature vectors are then fed into an FC layer, and the resultant feature vector is processed using an `argMax` function.

6.4 Experimental Evaluation

This section outlines the dataset used and its preprocessing, the parameters applied to the models discussed in Section 6.3, the hardware configuration, and the metric adopted to evaluate model performance.

6.4.1 Dataset

The novel dataset introduced in Chapter 3 was used to train and evaluate the models described in Section 6.3. As previously mentioned, the dataset exhibits class imbalance. To address this, all samples from classes with fewer than 63 instances were retained, while 63 samples were randomly selected for classes exceeding this threshold. Additionally, classes with fewer than 11 samples were excluded. In total, 14 classes were considered for the experiments: *right cut-in, object stopping, left cut-in, object crossing, object turning, object hazard light on, red crossing traffic light, object meeting, object emerging, pedestrian near parked vehicles, road works, object reversing, object pulling up*, and *object coming out*.

Experiments were conducted using four different class groupings:

- **All (14 classes)**: Includes all the classes listed above.
- **Motion-dependent (10 classes)**: Some classes rely more on visual features than motion cues (e.g., red traffic lights, vehicle hazard lights, road works, and pedestrians near parked vehicles). These were excluded when training and evaluating the Embedding LSTM model, resulting in a subset of ten classes.
- **Literature (5 classes)**: Includes classes commonly investigated in prior studies—*right cut-in, left cut-in, object stopping, object crossing*, and *object turning*. Note that although these classes have been studied, previous works did not consider all of them simultaneously.

The dataset was split into two portions: 80% for training and 20% for testing. Importantly, the split was performed at the video sample level rather than at the sequence level to prevent redundancy between training and testing phases. All

numerical explicit features were normalised, and categorical explicit features were converted into indicator variables.

For all experiments involving image inputs, the image size was set to 224×224 pixels. During training, image data augmentation was applied using the RandAugment [281] and AutoAugment [282] methods available in the PyTorch framework to mitigate overfitting.

6.4.2 Models Parameters

All the used models underwent fine-tuning, and the final parameters are reported in Table 6.1.

Table 6.1: Parameters values for each model.

Parameters	Embedding LSTM	CNN LSTM	Embedding CNN LSTM
			General
Random Seed	100	205	100
FC output Size	128	N. of Classes	N. of Classes
		Embedding Network	
Embedding Layer Output Size	112	None	576
		CNN Network	
CNN	None	ResNet18, ViT	ResNet18
		LSTM Network	
LSTM Layers	1	1	1
Dropout	0	0	0
Input Sequence Length	10-18	10-13	10-13
Encoder Hidden Size	112	128	128
		Hyperparameters	
Dropout Rate	0.8	0.8	0.8
Momentum	0.9	0.9	0.9
Loss function	Cross Entropy	Cross Entropy	Cross Entropy
Batch Size	128	50	50
Number of Epochs	15	15	40
Clip Gradient	5	5	5
Weight Decay	0.0006	0.0001	0.0001
Learn Rate	0.0001	0.0003	0.0002
LR Step Size	4	15	40
Gamma	0.01	0.01	0.01
Optmiser	Adam	SGD	SGD
Number of Classes	5, 10, 14	5, 10, 14	5, 10, 14

6.4.3 Software and Hardware Configurations

All experiments were conducted using an Intel(R) Xeon(R) Gold 5118 CPU @ 2.30 GHz and an NVIDIA Quadro RTX 6000 GPU with 24 GB of memory. All code and experiments were implemented in Python using the PyTorch framework.

6.4.4 Metrics

In order to evaluate the performance of traffic hazard event recognition, the following metrics were adopted: accuracy, precision, recall, F1-score, and the confusion matrix.

Accuracy was chosen to indicate how closely the predicted results $\hat{y}$ match the ground truth values y, and it is expressed as follows:

$$Accuracy(y, \hat{y}) = 1/n_{samples} \sum_{i=0}^{n_{samples}-1} 1(\hat{y}_i = y_i). \tag{6.3}$$

Precision measures the proportion of predicted positive cases that are correct, and it is expressed by:

$$Precision = \frac{TruePositives}{TruePositives + FalsePositives}. \tag{6.4}$$

Recall measures the proportion of actual positive instances that the model correctly classifies, and it is expressed as follows:

$$Recall = \frac{TruePositives}{TruePositives + FalseNegatives}. \tag{6.5}$$

The F1-score is the harmonic mean of precision and recall, and it is used to evaluate the balance between the two. It is expressed as follows:

$$F1\_score = 2 \times \frac{(Precision \times Recall)}{(Precision + Recall)}. \tag{6.6}$$

The confusion matrix is a valuable tool for visualising and understanding the types of errors a model makes. It provides a clear breakdown of how predictions align with actual classes, making it easier to identify patterns in misclassifications. For instance, it helps in analysing how frequently a specific class is incorrectly classified as another, offering insights into potential weaknesses or biases in the model.

6.5 Results and Discussions

The models described in Section 6.3 were subjected to various experiments to evaluate their performance using the novel dataset. These experiments involved variations in input types, input sequence lengths, and the number of classes.

6.5.1 Embedding TA LSTM Model

Table 6.2 presents the results of the Embedding TA LSTM model across different combinations of input features, using an input sequence length of 13 frames. The model's performance was evaluated using several explicit feature sets: conventional features alone; conventional features combined with the visible side; conventional features combined with rear light status; and a combination of conventional features, visible side, and rear light status.

Table 6.2: Results for the Embedding TA LSTM model when using different input types, number of classes, and an input sequence length of 13.

Input Type	**N. of Classes**	**Acc (%)**	**Prec (%)**	**Rec (%)**	**F1 (%)**
Conventional	10	51.8	51.0	38.0	37.0
Visible Side	10	70.8	66.0	63.0	63.0
rear light Status	10	70.8	62.0	60.0	59.0
rear light Status + Visible Side	10	**77.9**	75.0	69.0	69.0
rear light Status + Visible Side	5	90.4	90.0	90.0	90.0

Using only conventional input features results in the lowest performance across all metrics, with an accuracy of 51.8%, precision of 51%, recall of 38%, and an F1 score of 37%. This highlights the model's difficulty in making reliable predictions when provided with limited input data.

In contrast, incorporating additional features significantly improves accuracy and other performance metrics. For instance, adding either the visible side or rear light status to the conventional features increases accuracy by up to 19.0%. These additional features also enhance precision, recall, and F1 scores. The most substantial improvement—up to 26.1%—is observed when both the visible side and rear light

status are combined with conventional features. This improvement aligns with expectations, as these features are closely associated with specific manoeuvres. For example:

- Objects performing a stopping manoeuvre often reveal their rear or rear-side views.
- Crossing manoeuvres typically display an object's left or right side.
- Turning manoeuvres are indicated by active left or right indicator lights.

When considering the "literature classes" set, which includes only five classes, the model achieves significantly better performance, with accuracy, precision, recall, and F1 score all reaching approximately 90%. The variation in performance across class sets can be attributed to the number of classes: a larger number increases classification complexity. Additionally, the "motion-dependent classes" set, which includes 10 classes, includes several classes with fewer samples, further challenging the model. This performance disparity also suggests that prior studies focusing on a limited number of behaviours or agent types may overestimate model performance, as expanding the number of classes and agent types can reveal reduced accuracy in more complex, realistic scenarios.

The confusion matrices in Fig. 6.4 illustrate the model's performance using different feature sets. In Fig. 6.4a, the model struggles to classify classes such as *object emerging*, *object meeting*, *object coming out*, and *object pulling up* when using only conventional features. Additionally, many instances of *object turning* are misclassified as *object stopping*, and vice versa, while a significant number of *object reversing* instances are misclassified as *object turning*.

Incorporating visible side information (Fig. 6.4b) improves classification across most classes, with the exception of the *object stopping* class. The most notable improvements are observed in the *object crossing*, *emerging*, *meeting*, *reversing*, *coming out*, and *pulling up* classes.

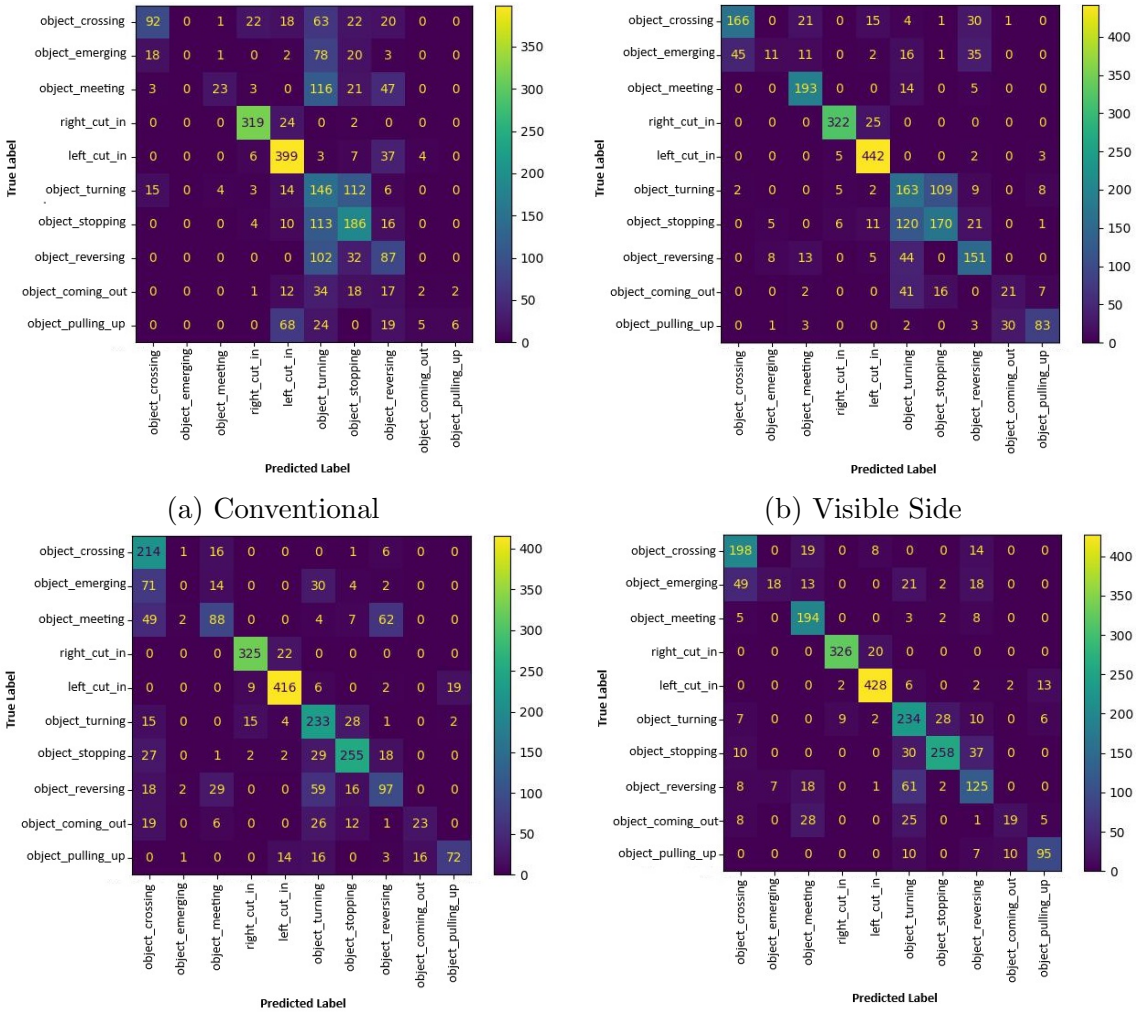
Adding rear light status (Fig. 6.4c) further enhances classification for all classes except *object emerging*. Notably, rear light status features help distinguish between *object turning* and *object stopping*, as these classes are associated with distinct rear light signals.

Combining conventional features with visible side and rear light status (Fig. 6.4d) achieves the best overall performance across all classes. However, some confusion remains, particularly in distinguishing:

- *Object emerging* from *object crossing*, *object meeting*, *object turning*, and *object reversing*.
- *Object coming out* from *object meeting* and *object turning*.
- *Object reversing* from *object turning*.

This highlights the need for further refinement to address residual ambiguities between these closely related classes.

Fig. 6.5 shows how accuracy varies with input sequence length. These fluctuations may be attributed to variations in the occurrence of visible side and light status features. The model achieved its highest accuracy of 77.9% with a sequence length of 13 frames.



(a) Conventional

(b) Visible Side

(c) Tailight

(d) Visible Side + Tailight

Figure 6.4: Confusion matrices for the explicit feature experiments.

6.5.2 CNN LSTM model

The results presented in Table 6.3 demonstrate the performance of the CNN-LSTM model, using ResNet18 as the feature extractor, under various configurations of input sequence lengths and class numbers. Only the local context image was considered as input, as it yielded the best results compared to other image input types. A detailed comparison with other input image types is discussed in the ablation study section.

Table 6.3: Results for the CNN LSTM model when using different input sequence lengths and number of classes. The ResNet18 CNN network was used as the feature extractor.

Network	Input Seq	N. of Classes	Acc (%)	Prec (%)	Rec (%)	F1 (%)
Resnet18 (pretrained)	13	14	68.5	67.0	61.0	60.0
Resnet18	13	14	68.7	66.0	60.0	60.0
Resnet18	13	10	69.4	62.0	62.0	60.0
Resnet18	13	5	77.3	79.0	77.0	77.0
Resnet18	18	14	69.2	68.0	67.0	66.0

Increasing the input sequence length from 13 to 18 results in modest improvements across all metrics when considering 14 classes. For example, accuracy in-

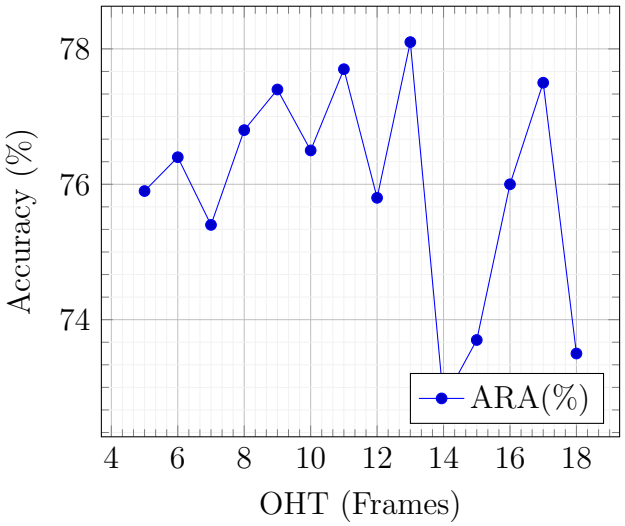


Figure 6.5: Relationship between OHT and LC recognition accuracy score for the Embedding TA LSTM model when using the combination of the conventional, visible side, and the rear light statuses information.

creases from 68.7% to 69.2%, precision rises from 66% to 68%, and both recall and F1 score improve significantly—from 60% to 67% and 60% to 66%, respectively. This suggests that providing more temporal information enhances the model's ability to distinguish between classes. However, this improvement comes at the cost of increased computational complexity.

The highest accuracy (77.3%) is achieved when the model is tested with five classes. This aligns with the general observation that reducing the number of classes simplifies the classification task, thereby improving accuracy.

Another experiment involved training the CNN model with the ResNet feature extractor, initially using the UCF101 action recognition dataset, and then fine-tuning the model on the novel dataset by initialising the CNN model parameters with the learned weights from UCF101 [283]. As reported in Table 6.3, this transfer learning approach did not significantly improve accuracy. There was only a slight increase in precision (67% vs. 66%) and recall (61% vs. 60%) compared to the non-pretrained version when considering 14 classes. The lack of improvement can be attributed to the differing nature of the datasets: UCF101 focuses on human actions, whereas the novel dataset pertains to traffic-related objects (e.g., vehicles), which may not share sufficient common features to benefit from pretraining.

The results for the CNN-LSTM model were also reported when considering the dataset with 10 classes to allow a fair comparison with the Embedding TA LSTM model. The best accuracy achieved by the Embedding TA LSTM model was 77.9%, whereas the CNN-LSTM model achieved 69.4%. Although the Embedding TA LSTM model outperformed the CNN-LSTM in terms of accuracy, it relies on explicitly selected features. In contrast, the CNN-LSTM model requires only input images, making it simpler and more flexible.

The confusion matrices shown in Fig. 6.6 illustrate that the CNN-LSTM model performs better at correctly classifying certain classes, such as *object crossing*, *object emerging*, *left cut-in*, *object reversing*, and *object pulling up*. However, it struggles more with other classes, suggesting that features like the target object's visible side

and rear light status help improve classification accuracy. It is important to note that the visible side and rear light information used by the Embedding TA LSTM model was manually labelled. If this information were instead extracted using a recognition algorithm, it would introduce additional uncertainties, potentially leading to a decrease in accuracy.

6.5.3 ViT LSTM Model

Table 6.4 reports the results of the ViT-LSTM model when trained and evaluated using the local context image as input. The ViT-LSTM model outperformed the CNN-LSTM model by 1.1% and 5.6% when trained and tested on the dataset of 14 classes and 5 classes, respectively. Additionally, the model achieved relatively balanced precision, recall, and F1 scores for both sets—ranging from 66% to 68% when considering 14 classes, and from 83% to 84% when considering 5 classes.

On the other hand, when considering the dataset with 10 classes, the CNN-LSTM model outperformed the ViT-LSTM by 9.6% in accuracy. Furthermore, the ViT-LSTM's precision, recall, and F1 scores dropped significantly, falling within the range of 52% to 53%.

Table 6.4: Results for the ViT LSTM model when using different input sequence lengths and number of classes. The CNN used were ResNet18 and ViT.

Network	Input Seq.	N. of Classes	Acc. (%)	Prec. (%)	Rec. (%)	F1 (%)
ViT	13	14	69.8	68.0	67.0	66.0
ViT	13	10	59.8	53.0	53.0	52.0
ViT	13	5	82.9	84.0	83.0	83.0

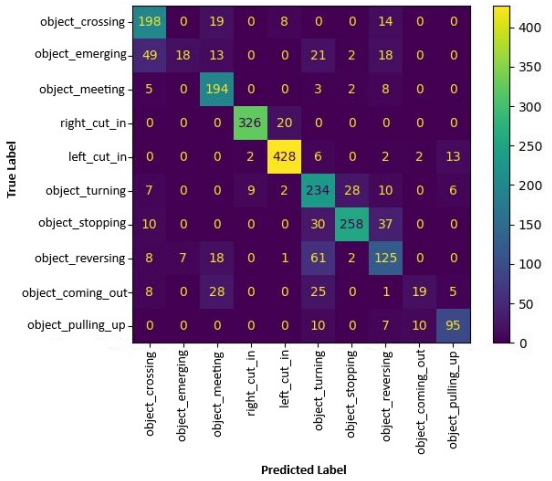
Although the ViT-LSTM models outperformed the CNN-LSTM models for certain dataset configurations, the remaining experiments were conducted using the CNN model due to limited GPU memory resources. For instance, the other experiments involved multi-stream methods, which require multiple streams of diverse input data, making them more resource-intensive.

6.5.4 Embedding CNN-LSTM Model

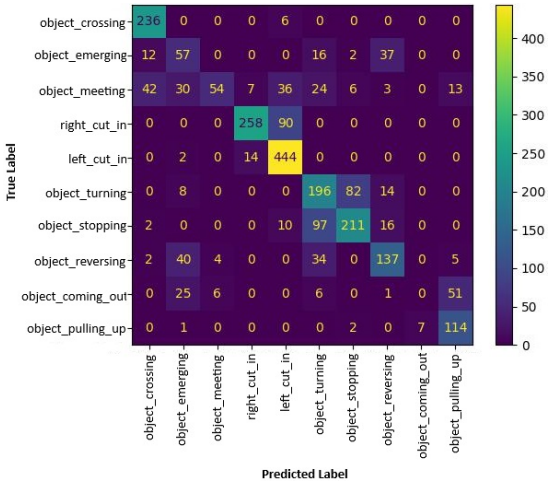
The results in Table 6.5 illustrate the performance of the Embedding CNN-LSTM model across various input types, sequence lengths, and numbers of classes. Experiments were conducted using the dataset with 10 classes ("motion-dependent classes") to facilitate a direct comparison with the Embedding TA LSTM model. Consistent with the findings from the Embedding TA LSTM model, the best performance for the Embedding CNN-LSTM model was achieved by incorporating both the visible side of the target object and the rear light status.

Including only the visible side improved accuracy by 8.4%, while using only the rear light status resulted in an 8.3% increase. Combining both features led to a 10.1% improvement, yielding the highest precision, recall, and F1 score values.

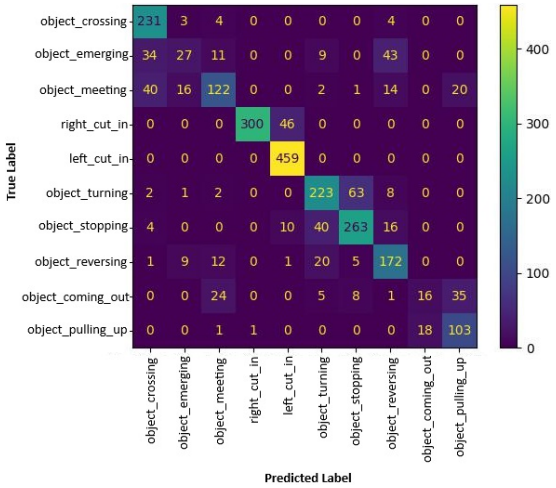
When comparing the Embedding TA LSTM and Embedding CNN-LSTM models, the latter achieved the highest accuracy (78.2% vs. 77.9%). However, the Embedding CNN-LSTM model had lower precision (71% vs. 75%), while both models achieved the same recall (69%) and F1 score (69%).



(a) Embedding TA LSTM



(b) CNN LSTM



(c) Embedding CNN LSTM

Figure 6.6: Confusion matrices for different models.

The confusion matrices in Fig. 6.6c highlight how the Embedding CNN-LSTM model combines the strengths and limitations of the Embedding TA LSTM and CNN-LSTM models. For instance, while the Embedding TA LSTM model struggles to classify the *object emerging* class, it performs well on *object meeting* and *object coming out*. Conversely, the CNN-LSTM model accurately classifies *object emerging* but struggles with *object meeting* and *object coming out*. The Embedding CNN-LSTM model balances these strengths, yielding intermediate classification results across classes—for example, achieving 18, 57, and 27 correct classifications for *object emerging* with the Embedding TA LSTM, CNN-LSTM, and Embedding CNN-LSTM models, respectively.

Further experiments examined the Embedding CNN-LSTM model's performance across different input sequence lengths and class sets, as shown in Table 6.5. The highest accuracy, 77.9%, was achieved with a sequence length of 18 frames when using a combination of conventional features, visible side, and rear light status. Notably, even when using only conventional features, the model achieved an accuracy of 76.1%, suggesting that the CNN layers may be implicitly learning aspects of the visible side and rear light status. When transitioning from the *motion-dependent*

classes set to the more comprehensive *all classes* set, accuracy generally decreases slightly for individual input types, though combined features still maintain high accuracy.

As the sequence length increases, accuracy improves slightly, indicating that longer sequences help the model capture temporal dependencies more effectively. However, this improvement comes at the cost of increased computational demands, highlighting a trade-off between accuracy gains and resource requirements.

Table 6.5: Results for the Embedding CNN LSTM model when using different input sequence lengths and the number of classes. The CNN used was ResNet18.

Input Type	**Input Seq.**	**N. of Classes**	**Acc. (%)**	**Prec. (%)**	**Rec. (%)**	**F1 (%)**
Conventional	13	10	68.1	60.0	58.0	58.0
Visible Side	13	10	73.5	69.0	63.0	64.0
Rear light Status	13	10	76.4	71.0	67.0	67.0
Rear light Status + Visible Side	13	10	78.2	71.0	69.0	69.0
Conventional	13	14	76.1	72.0	69.0	69.0
Visible Side	13	14	74.0	73.0	67.0	68.0
Rear light Status	13	14	74.7	71.0	68.0	68.0
Rear light Status + Visible Side	13	14	77.2	73.0	70.0	69.0
Rear light Status + Visible Side	10	14	75.7	73.0	67.0	67.0
Rear light Status + Visible Side	12	14	76.2	74.0	68.0	68.0
Rear light Status + Visible Side	15	14	77.3	74.0	70.0	69.0
Rear light Status + Visible Side	18	14	**77.9**	76.0	70.0	70.0

6.5.5 Multi-stream CNN-LSTM Model

Different input image combinations were used to train the Multi-stream CNN-LSTM model. The best performance was achieved using motion-based and local context images, yielding an accuracy of 67.8%. The second-best result was obtained using original and local context images, with an accuracy of 60.5%. Further details on experiments with other input types are provided in the ablation study section.

The lower performance of the Multi-stream CNN-LSTM model compared to the single-stream CNN-LSTM model can be attributed to the following factors:

- **Limited Data:** Although the dataset contains 645 samples, it spans 16 different classes, with some classes having fewer than 20 samples. Given the increased complexity of the Multi-stream CNN architecture compared to the CNN-LSTM model, the limited data may lead to overfitting or reduced generalisation capability.

- **Computational Resources:** Multi-stream CNN-LSTM models demand significantly more computational resources, which restricts the batch size and the number of training iterations available for model optimisation.

6.5.6 Comparison With Other Works

Although there is no direct comparison between our work and existing studies, we have presented the accuracy metrics from prior research on pedestrian and vehicle intention recognition in Tables 6.6 and 6.7, respectively. These studies demonstrate reasonable performance in recognising the discrete behaviours of traffic agents; however, most focus on narrow aspects, such as pedestrian intentions to cross or vehicle lane changes. Furthermore, many of these studies examine pedestrian or vehicle

behaviours in isolation, without accounting for interactions between multiple agents in real-world traffic scenarios.

Our proposed novel dataset addresses this limitation by encompassing a more comprehensive range of behaviours and agent types. As shown by our results, there is a noticeable drop in accuracy as the complexity of the dataset increases. For instance, when using the full dataset with 14 behaviour classes (the 'all classes' set), the best accuracy achieved was 77.9%. In contrast, when using the 'literature classes' set, which contains only five behaviour classes, accuracy reached 90.4%. It is important to note that even the 'literature classes' set includes more classes and object types than those reported in prior works. This suggests that while high accuracy rates in pedestrian and vehicle intention recognition may indicate progress in these tasks, they may not fully reflect the complexities of real-world traffic scenarios.

Table 6.6: Pedestrian discrete intention recognition accuracy for different works when using different types of datasets, intention types, and models.

Work	**Intention Types**	**Model**	**Dataset**	**Acc(%)**
[128]	C, NC, W	LSTM	PIE, JAAD	79.0
[284]	C, NC/TL, TR, STOP	CNN	JAAD	88.0/96.0
[285]	C, NC, ST	GCN+Conv-LSTM	PIE	81.0
[286]	C, NC	LSTM	PIE/JAAD	91.0/83.0
[211]	C, NC	LSTM+GCN	PIE	79.0
[287]	C, NC	Transformer	PIE/JAAD	89.0/86.0
[288]	C, NC	3DCNN	JAAD	84.9
[289]	C, NC	STGCN+LSTM	PepScenes	94.4
[205]	C, NC	Factor-CRF	JAAD	70.0
[122]	C, NC	CNN+MLP	PIE/JAAD	84.0/87.0
[209]	C, NC	3DResNet50	JAAD	75.2
[76]	C, NC	3DCNN+GRU+Attention	PIE, JAAD	87.0/85.0
[290]	C, NC	CNN+GRU	JAAD	75.7
[291]	C, NC	ViT	PIE, JAAD	86.2/85.5
[210]	C, NC	SVM	Self-Collected	92.4
[212]	C, NC	GCN+CNN	PIE, JAAD	89.0/86.0
[124]	C, NC	GCN	JAAD	63.0
[126]	C, NC	CNN+GRU	JAAD	83.0
[123]	C, NC	Transformer	PIE	91.0
[127]	C, NC	GCN	PIE	83.0
[292]	C, NC	CNN+GRU	PIE, JAAD	87.6/91.2
[293]	C, NC	CNN+GRU	PIE	91.0
[294]	C, NC	Transformer	PIE, JAAD, PSI	**92.0**/88.0/85.0
[295]	C, NC	CNN+MLP	JAAD	85.0
[296]	C, NC	CNN+MLP	Self-Collected	90.23
[297]	STOP/GO	Stacked GRU	PIE, JAAD, TITAN	76.85/67.8/64.35
[298]	C, NC	Transformer	PIE, JAAD	91.0/87.0
[203]	C, NC	LSTM	PIE, JAAD	91.0/**89.0**
Ours	5 classes	ViT+LSTM	Novel Dataset	82.9
Ours	14 classes	Embedding+CNN+LSTM	Novel Dataset	77.9

6.5.7 Ablation Study

CNN-LSTM Model

The CNN-LSTM model was trained and evaluated using the input image types described in Section 6.4. The results are reported in Table 6.8. The best performance was achieved when using input image types that contain more local context information, such as the *local context*, *local depth*, and *motion-based local context* images.

Table 6.7: Vehicle discrete intention recognition accuracy for different works when using different types of datasets, intention types, and models.

Work	Intention Types	Model	Dataset	Acc(%)
[299]	LLC, RLC, LK	ANN	NGSIM	73.33
[171]	LLC, RLC, LK	Surrounding-Aware LSTM	NGSIM	86.19
[300]	LLC, RLC, LK	Hybrid GMM+CHMM	OWN	94.4
[301]	LLC, RLC, LK	CNN+LSTM	PREVENTION	74.46
[173]	LLC, RLC, LK	ANN and SVM	NGSIM	98.8
[173]	LLC, RLC, LK	ANN and SVM	NGSIM	97.1
[120]	LLC, RLC, LK	Spatio-Temporal CNN	PREVENTION	91.94
[302]	LLC, RLC, LK	BLLC	NGSIM	66.41
[303]	LK, TL, TR, UT	LSTM+Attention	NDS	99.6
[304]	LC, LK	LSTM	HighD	98.8
[305]	LLC, RLC	LSTM	INTERACTION	95.2
[306]	LLC, RLC	RNN	Self-Collected	96.0
[307]	LLC, RLC, LK	LSTM	NGSIM	86.21
[305]	LLC, RLC, LK	Fuzzy+LSTM	NGSIM	92.40
[308]	LLC, RLC, LK	Two-level CNN	OWN	73.97
[69]	LLC, RLC, LK	Spatio-Temporal CNN, 3DCNN	PREVENTION	91.91 / 86.51
[121]	LLC, RLC, LK	CNN	PREVENTION	83.4
[309]	LLC, RLC, LK	Random Forest + SVM	NGSIM	82
[310]	LLC, RLC, LK	CNN+ViT	PREVENTION	81.23
[311]	MRD, Y, M	Variational RNN	INTERACTION	82.97
[312]	LLC, RLC, LK	KNN	HighD	98.02
[313]	LLC, RLC, LK	Temporal CNN+Attention	CitySim	98.2
[314]	LLC, RLC, LK	Dual Transformer	NGSIM	93.57
[314]	LLC, RLC, LK	Dual Transformer	HighD	98.98
[315]	LC, ACCE, DECE	Transformer	NGSIM, HighD	84.38
[280]	LLC, RLC, LK	Video ViT	PREVENTION	85.0
[316]	LLC, RLC, LK	Bi-LSTM+Attention	NGSIM	92.2
Ours	5 classes	ViT+LSTM	Novel Dataset	82.9
Ours	14 classes	Embedding+CNN+LSTM	Novel Dataset	77.9

Local context information is less susceptible to variations caused by lighting conditions, viewpoint changes, occlusion, and complex backgrounds typical of traffic scenarios. As a result, the CNN model can learn features that are more robust to these variations, leading to improved generalisation performance.

Table 6.8: Results for the CNN LSTM model when using different input image types.

Input Image 1	Acc (%)
Raw	37.6
Local Context	69.1
Depth	41.7
Local Depth	64.5
Object of Interest	40.8
No Target	42.1
Blended	40.8
Lane Information	38.9
Motion-based Local Context	65.4

Multi-stream CNN-LSTM Model

The Multi-stream CNN-LSTM model was trained and evaluated using the different input image types described in Section 6.4. The results are reported in Table 6.9. Since the best performance of the CNN-LSTM model was achieved using the local

context input image type, it was set as the default for *input image 2*. The best results for the Multi-stream model were obtained when *input image 1* was either the raw image or the motion-based local context image.

Table 6.9: Results for the Multi-stream CNN LSTM model when using different combinations of input image types.

Input Image 1	**Input Image 2**	**Acc (%)**
Raw	Local Context	60.5
Depth	Local Context	51.5
Local Depth	Local Context	54.2
Object of Interest	Local Context	51.6
Object of Interest Graph	Local Context	51.8
No Target	Local Context	50.0
Blended	Local Context	57.2
Lane Information	Local Context	56.3
Motion-based Local Context	Local Context	67.8

6.6 Conclusions

This chapter presented a comprehensive evaluation of various machine learning models trained on a novel traffic hazard dataset. The experiments investigated three key factors influencing recognition performance: class granularity, input types, and model architectures.

The results revealed a trade-off between class diversity and model accuracy. Using the *'literature' class set* with five behaviours categories, the best-performing model achieved an accuracy of 90.4%. In contrast, expanding to the *'all' class set* with 14 categories reduced accuracy to 77.9%, underscoring the increased complexity of recognising a broader range of traffic agent behaviours and the need for more balanced training data.

Regarding input types, images providing local contextual information (e.g., cropped views centred on the target agent) outperformed those with global context. Furthermore, combining explicit and implicit features led to notable performance gains compared to using either feature type alone, highlighting the value of multimodal feature integration.

Model comparisons showed that ViTs consistently outperformed CNNs, particularly when both feature types were utilized. However, ViTs demanded significantly more computational resources, which constrained their scalability in this study.

The experiments also demonstrated that recognising the visible side (front or rear) of traffic agents enhanced prediction accuracy, especially when relying solely on explicit features. This directional cue proved useful in disambiguating similar behaviourss.

Finally, incorporating vehicle light indicators (e.g., brake and turn signals) improved model performance, suggesting their practical utility in real-world intelligent vehicle systems—particularly in environments lacking widespread V2X infrastructure.

These findings offer valuable insights into the factors affecting hazard recognition and provide a foundation for future improvements in model design and dataset development.

Chapter 7

Complete Potential Traffic Hazard Recognition Pipeline

This chapter presents a complete pipeline for recognising potential traffic hazard events. It includes modules for object detection and tracking, rear light status recognition, identification of the visible side of objects from the ego vehicle's perspective, and the recognition of potential traffic hazard events. As concluded in Chapter 2, few studies have addressed a comprehensive pipeline system for understanding the behaviour of road users. Implementing such a system is essential, as it enables performance assessment of each module and its influence on subsequent stages in the pipeline. This facilitates targeted improvements, particularly in modules with lower performance. Moreover, a complete pipeline can automatically extract a greater number of traffic hazard samples, helping to address data imbalance and improve generalisation during machine learning development. Additionally, it supports the evaluation of real-time system performance.

The structure of this chapter is as follows: Section 7.1 provides an overview of the complete pipeline system, including details of the modules and their functions. Section 7.2 discusses the object detection, tracking, and overlooked cues recognition modules. Sections 7.3 and 7.4 cover the minimum trajectory filtering and potential traffic hazard event recognition modules, respectively. Section 7.5 outlines the experimental evaluation methods. Finally, Section 7.6 presents and discusses the results.

7.1 Complete Pipeline Overview

Figure 7.1 presents a general flow diagram of a complete potential traffic hazard event recognition system pipeline. The diagram illustrates two possible methods: one that includes only the blue stages, and another that incorporates an additional stage highlighted in orange. Both methods involve input data, object detection and tracking, rear light and visible side recognition, target object filtering based on a minimum trajectory sequence length, and potential traffic hazard event recognition modules.

In the method with the additional orange stage, the detected and tracked target objects are filtered based on specific rear light statuses and visible side classes before applying the potential traffic hazard event recognition. For example:

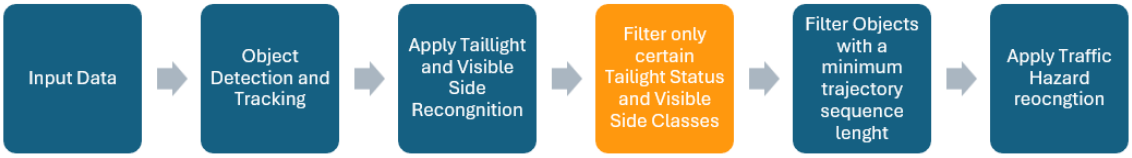


Figure 7.1: Hazard perception complete pipeline.

- A braking rear light status could indicate a braking manoeuvre.
- A left or right rear light status could indicate a right or left turn or cut-in.
- Simultaneous left and right rear light indicators could suggest that a vehicle has stopped or broken down on the road.
- A visible left or right side could suggest that the target object is crossing in front of the EV.

This approach consumes fewer computational resources, as the potential traffic hazard event recognition algorithm is applied only to the filtered objects. However, it may overlook other objects that do not exhibit these characteristics but are still relevant based on their motion or other visual cues—such as reversing vehicles, construction equipment, traffic lights, or vehicles turning or cutting in without using rear light indicators. Additionally, it may still include objects that do not pose any hazard, such as stationary vehicles.

Conversely, the method containing only the blue stages considers every detected and tracked object. While this reduces the chance of overlooking a hazardous object, it also includes many non-hazardous ones, such as stationary vehicles and pedestrians walking on the pavement. This significantly increases computational and memory requirements. Despite the higher resource consumption, this method is likely to yield higher accuracy, as all detected and tracked objects are considered.

In conclusion, the proposed complete pipeline includes the blue stages, where:

- The input data consist of a sequence of frames from a traffic scene.
- The object detection and tracking module generates bounding boxes, assigns object classes, and tracks each object over time within the input data.
- The rear light status recognition module assigns a rear light status to each detected and tracked vehicle.
- The visible side recognition module determines which side of the target object is visible from the EV's perspective.
- The target object filtering module retains only those with a minimum trajectory sequence length.
- Finally, the potential traffic hazard event recognition module classifies whether the filtered target objects represent a hazard and identifies the type of hazard.

The following challenges should be considered when developing the complete pipeline system:

- The complete pipeline is evaluated using the test set of the proposed road hazard dataset. Each dataset sample contains information about a specific target object performing a particular behaviour. However, the detection and tracking module considers all objects in the frame and assigns different object IDs. Not all objects in the frame—including some target objects in the test set—are detected and tracked.

- Incorrect object type classification may occur; for example, a pedestrian could be misclassified as a vehicle and subsequently considered for rear light status recognition.

- Objects may be mistracked due to occlusion or changes in appearance across frames. For instance, objects may appear smaller or larger, and lighting variations can alter their appearance.

- Some video frames include the front of the EV, which is detected and tracked by the detection and tracking module.

- Several objects are tracked only briefly, resulting in insufficient sequence length to recognize their behaviour.

- An object might meet the minimum tracking length requirement but not in a continuous sequence, meaning it may be absent in some frames within the sequence.

7.2 Detection, Tracking and Recognition Modules

The goal of the object detection and tracking, rear light status recognition, and object visible side recognition modules is to acquire the necessary information to run the potential traffic hazard event recognition algorithm. This information includes the cropped image of the object, the object class, the object bounding box, the object ID, the object's rear light status, and the object's visible side. For this reason, these modules are implemented as a single function.

Figure 7.2 depicts the stages of these modules. First, all models are initialised with pre-trained weights obtained from previous sections. Each frame of a traffic scene video is fed into the detection and tracking module to acquire bounding boxes and object classes, and to assign tracking IDs to all objects. The acquired information for each object is then used to crop the input image at the location of the detected objects. Each cropped image is subsequently input into the rear light status recognition and visible side recognition modules.

Note that only cropped objects classified as vehicle types are fed into the rear light status recognition module; other objects are assigned a value of `-1` (unknown) for their rear light status.

The output of these processes is a file named `detected_tracked_objects.csv`, which contains the following columns:

- `clip_source`

- dataset
- frame_n
- target_obj_id
- object_type
- tailight_status
- tailight_status_int
- x_1, y_1, x_2, y_2
- img_path
- object_visible_side
- object_visible_side_int
- hazard_type_int
- hazard_type_name
- hazard_flag
- img_width, img_height
- ID

7.2.1 Object Detection and Tracking Module

The detection and tracking stages are crucial in the complete pipeline system, as all other stages depend on their output. The YOLO algorithm was chosen to implement these stages due to its state-of-the-art object detection performance and its capability to integrate tracking tasks. YOLO provides an out-of-the-box solution with several versions and models, including pre-trained weights available through the Ultralytics Python library. These features make it versatile and easy to set up and run.

The YOLO version 9 model 'e'—the latest available in the Ultralytics Python library and the one with the best performance—was chosen for this task. The current mAP(50–95) and mAP(50) performance metrics for the YOLOv9e model, when validated with the COCO dataset, are 55.6% and 72.8%, respectively. YOLOv9e offers two types of tracking algorithms: BoT-SORT [317] and ByteTrack [318]. The BoT-SORT tracker is robust against occlusion and supports re-identification, although it is computationally intensive. In contrast, ByteTrack is simpler and more efficient but struggles with occlusion and lacks re-identification capabilities. Since this research prioritises accuracy over efficiency, the BoT-SORT tracker was selected.

YOLOv9e can detect all object categories available in the COCO dataset. However, the traffic road hazard dataset includes additional categories not covered by the pre-trained detection weights, such as groups of pedestrians, road crossings, road work objects, and riders. Therefore, the traffic objects considered during the detection stage were pedestrians, bicycles, cars, motorcycles, buses, trucks, dogs, and horses.

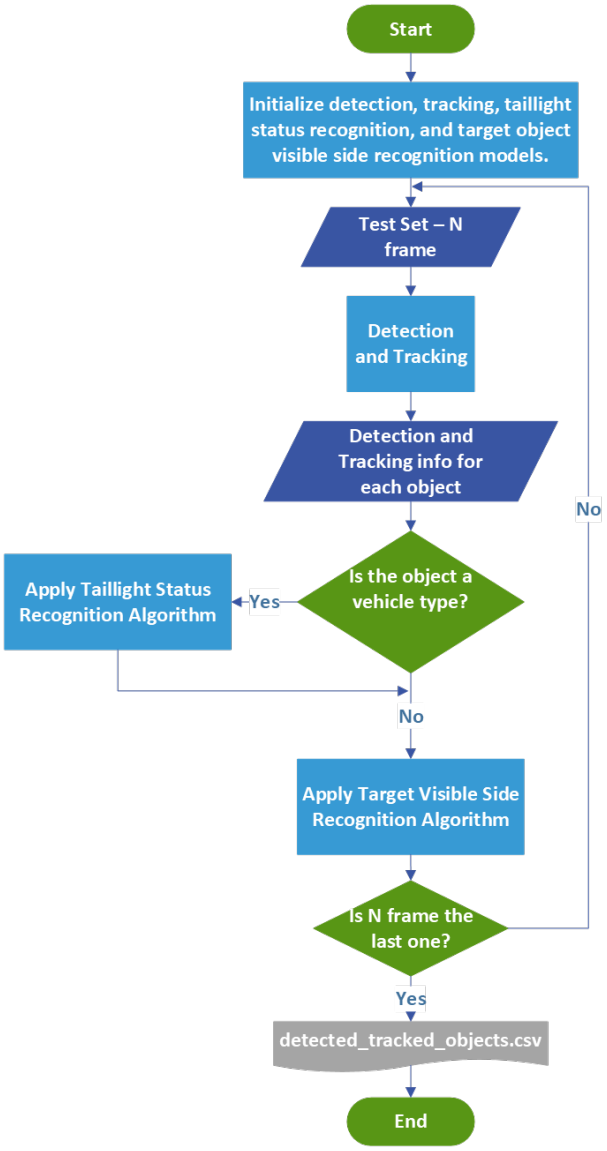


Figure 7.2: Hazard perception detection, tracking and recognition stage.

7.2.2 Rear Light Status Recognition Module

The proposed rear light status recognition algorithm, which adopts the state-of-the-art Vision Transformer (ViT) network described in Section 4, was chosen to predict the rear light status of detected and tracked vehicles. The model classifies the following rear light statuses: *brake light on, left indicator on, right indicator on, both indicators on, all lights off*, and an *unknown* class, achieving an accuracy of 77.0%.

Although image sequence-based methods have demonstrated higher accuracy [244], their reliance on multiple consecutive images limits their suitability for integration into a complete pipeline. The selected model was chosen for its ability to perform recognition using a single input image, making it a more practical and efficient solution.

7.2.3 Object Visible Side Recognition Module

The proposed object visible side recognition algorithm, based on the ConvNeXt model described in Section 4, was selected to predict the visible side of each detected and tracked object. The model distinguishes between the following visible sides: *front side, front left side, front right side, left side, right side, rear side, left rear side, right rear side*, and *unknown side*, achieving an accuracy of 83.0%.

To the best of the researcher's knowledge, this study is the first to explore the recognition of target objects' visible side as an input feature for traffic object behaviour recognition.

7.3 Minimum Trajectory Sequence Length Filtering Module

The potential traffic hazard event recognition algorithm proposed in Section 6 requires a minimum OH of 13 frames. Therefore, only target objects with a trajectory sequence length of at least 13 frames were considered. To enforce this requirement, the filtering approach illustrated in Figure 7.3 was proposed.

The process begins by reading the `detected_tracked_objects.csv` file generated by the detection and tracking modules. Next, the algorithm determines the number of frames in which a particular object appears in the sequence of traffic frames. If this number exceeds the minimum specified trajectory sequence length, the algorithm checks for any significant gaps between consecutive frames in which the object is present. If the gaps are within the defined limits, the information for the current object is retained; otherwise, the algorithm proceeds to the next object.

Each filtered object is treated as a separate video sample. For example, if a traffic scene video contains eight objects that meet the filtering requirements, each will be considered a distinct video clip. This approach is necessary because the potential traffic hazard event recognition algorithm accepts only one object per run.

The information for each object is recorded in the `all_extracted_videos.csv` file, which contains the following fields: `video_n`, `frame_n`, `start_frame`, `end_frame`, `hazard_type_int`, `hazard_type_name`, `hazard_flag`, `target_obj_id`, `ID`, `object_type`, `xc`, `yc`, `xc_speed`, `yc_speed`, `w`, `h`, `bbox_area`, `x_1`, `y_1`, `x_2`, `y_2`, `img_path`, `img_width`, `img_height`, `tailight_status`, `tailight_status_int`, `object_visible_side`, `object_visible_side_int`, `clip_source`, and `dataset`.

This file includes additional information compared to the one created by the object detection and tracking module. For instance, it contains `video_n`, `start_frame`, `end_frame`, `xc`, `yc`, `xc_speed`, `yc_speed`, `w`, `h`, and `bbox_area`. The `video_n` is an integer value assigned to each video clip corresponding to a filtered object. The `start_frame` and `end_frame` indicate the frames where tracking begins and ends, respectively. The other values are derived from `x_1`, `y_1`, `x_2`, and `y_2`. These additional values are generated because they are required as input for the potential traffic hazard event recognition algorithm.

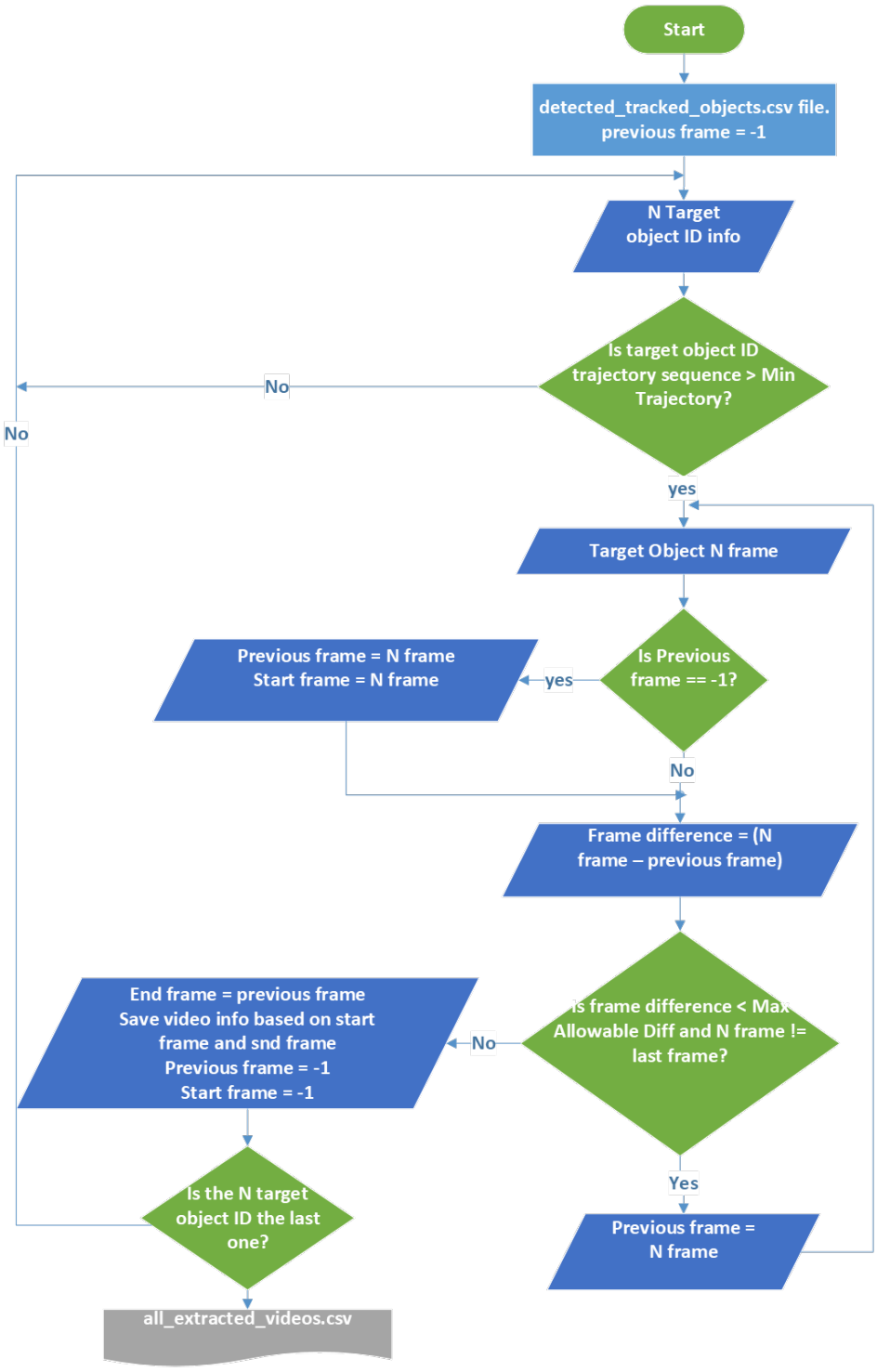


Figure 7.3: Target object filtering based on target object minimum trajectory.

7.4 Potential Traffic Hazard Event Recognition Module

The Embedding LSTM, CNN LSTM, and Embedding CNN LSTM models presented in Chapter 6 were investigated as potential traffic hazard event recognition modules in the complete pipeline.

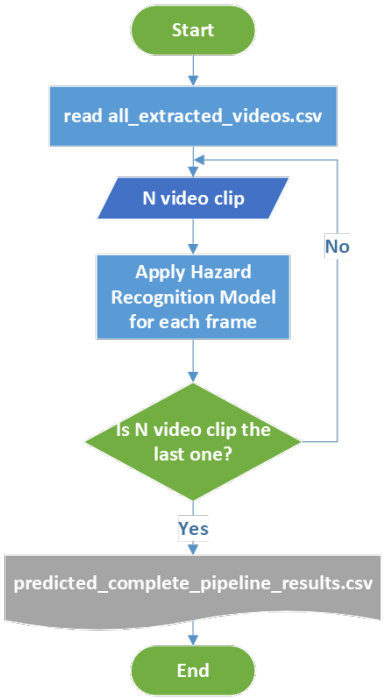


Figure 7.4: Potential traffic hazard event recognition flow diagram.

The flow diagram shown in Figure 7.4 illustrates the processes of the Potential Traffic Hazard Event Recognition Module. The process begins by reading the `all_extracted_videos.csv` file. Each video clip generated by the Minimum Trajectory Sequence Length Filtering Module is used as input to the recognition model, and each frame is assigned an integer value representing the predicted hazard type.

The output of this process is stored in the `predicted_hazard_results.csv` file, which contains the following fields: `frame_n`, `all_together_all`, `pred_hazard_type_int`, `img_dir_all`, `video_nu_all`, `f0_all`, `f1_all`, and `clip_source`. The `all_together_all` field includes all numeric and categorical input types, such as speed, object class, etc. The `f0_all` and `f1_all` columns store the start and end frames, respectively. These columns are required for computing the evaluation metrics.

7.5 Experimental Evaluation

This section describes the experimental setup, including the dataset, models, software and hardware, and metrics.

7.5.1 Dataset

The road hazard dataset test set was used to evaluate the complete pipeline algorithm, as it was not used to train the rear light status recognition, object visible side recognition, or potential traffic hazard event recognition algorithms. Additionally, it contains all the ground truth values necessary to measure the performance of the complete pipeline. However, not all hazard classes from the road hazard dataset were included, as some object classes—such as traffic lights, groups of pedestrians, road crossings, road works, and riders—were not supported by the pre-trained de-

tection and tracking model. Consequently, hazard classes such as red crossing traffic lights and road works were excluded.

As reported in Chapter 3, the novel potential traffic hazard event dataset did not include any no-hazard samples. However, no-hazard samples are essential for a complete pipeline system, as real-world scenarios often include non-hazardous events. Therefore, 300 no-hazard video samples were extracted: 250 were used for training, and the remaining 50 for testing.

The complete pipeline was also evaluated using three different class groupings, as discussed in Chapter 3: all classes, motion-towards classes, and literature-based classes. The input data to the system consists of sequences of RGB traffic scene frames, resized to (1280, 640).

7.5.2 Models

As discussed in Section 7.2.2, three potential traffic hazard event recognition models were used to evaluate the complete pipeline: the Embedding LSTM, CNN_LSTM, and Embedding CNN LSTM models. The pre-trained weights with the best accuracy results were selected for evaluation.

For instance, the best pre-trained weights for the Embedding_TA_LSTM model included both rear light status and the target object's visible side features. The best CNN_LSTM model used the ResNet network with the local context image as input. Lastly, the best pre-trained weights for the Embedding_CNN_LSTM model were trained using the local context image, rear light status, and object visible side features.

The best accuracy achieved by each model, along with the dataset type used, is reported in Table 7.1, under the column titled *Potential Traffic Hazard Event Recognition Accuracy with True Values (PTHERATV)*.

7.5.3 Software and Hardware

All experiments were executed using an Intel(R) Core(TM) i7-10750H @ 2.60GHz and an NVIDIA GeForce GTX 1080 Ti 12GB. All the codes and experiments were done using Python and the PyTorch Framework.

7.5.4 Metrics

To evaluate the performance of the complete pipeline, the following metrics were used: Target Object Detection Accuracy (TODA), Rear Light Status Recognition Accuracy (RLSRA), Object Visible Side Recognition Accuracy (VSRA), and Hazard Type Classification Accuracy.

The **TODA** measures the proportion of objects identified by the detection and tracking module that are correctly matched with the objects labeled in the test set of the road hazard dataset. It is defined as:

$$\text{TODA} = \frac{\text{present\_target\_objects}}{\text{all\_target\_objects}} \tag{7.1}$$

where `present_target_objects` is the number of target objects retained by the filtering module that are present in the test set, and `all_target_objects` is the total number of target objects in the test set.

The **RLSRA** measures how often the rear light status recognition module correctly predicts the actual rear light status as labeled in the test set. The **Object VSRA** measures the accuracy of the visible side predictions. The **Hazard Type Classification Accuracy** evaluates how often the hazard types predicted by the complete pipeline match the ground truth labels.

All these accuracies were computed using the `accuracy_score` function from the `sklearn.metrics` Python library, defined as:

$$\text{Accuracy}(y, \hat{y}) = \frac{1}{n_{\text{samples}}} \sum_{i=0}^{n_{\text{samples}}-1} \mathbb{1}(\hat{y}_i = y_i) \tag{7.2}$$

where y_i and $\hat{y}_i$ are the true and predicted values for the i-th sample, and n_{samples} is the total number of samples.

Figure 7.5 illustrates the flow diagram for calculating performance metrics. The process begins by reading two key files: `predicted_hazard_results.csv`, which contains the output from the hazard recognition module, and `ground_truth_information.csv`, which holds the actual ground truth data. To ensure accurate evaluation, the intersection method is used to identify common frames between the two files, allowing calculations even when some objects are missed by the detection and tracking module.

Since object IDs in the test set may not align with those generated by the detection and tracking algorithm, the Intersection over Union (IoU) metric is used to match bounding boxes. An IoU threshold of 0.35 is applied, meaning that if the overlap between predicted and actual bounding boxes is at least 35%, they are considered a match. This threshold balances avoiding false matches with neighboring objects and accepting partially occluded objects.

Once an object meets the IoU threshold, it is counted as correctly detected, and the corresponding predicted and actual values for rear light status, visible side, and hazard type are recorded.

After processing all video clips, the system calculates the key metrics. It is important to note that the TODA is computed over all predicted target objects compared to the full test set, while the other metrics are calculated only for correctly detected objects.

7.6 Results and Discussion

Table 7.1 presents the results of the complete pipeline, comparing different input types, models, and class groupings. It details the performance metrics for TODA, RLSRA, and VSRA. Additionally, the table reports the accuracy of the potential traffic hazard event recognition module, evaluated using both predicted values and actual ground truth data for detection and tracking, rear light status, and visible side recognition.

Detection Performance

The achieved TODA remained relatively stable across different class groupings, ranging from 67.19% to 69.87%. The lowest performance was observed when all classes from the traffic hazard dataset were included, likely due to the increased sample

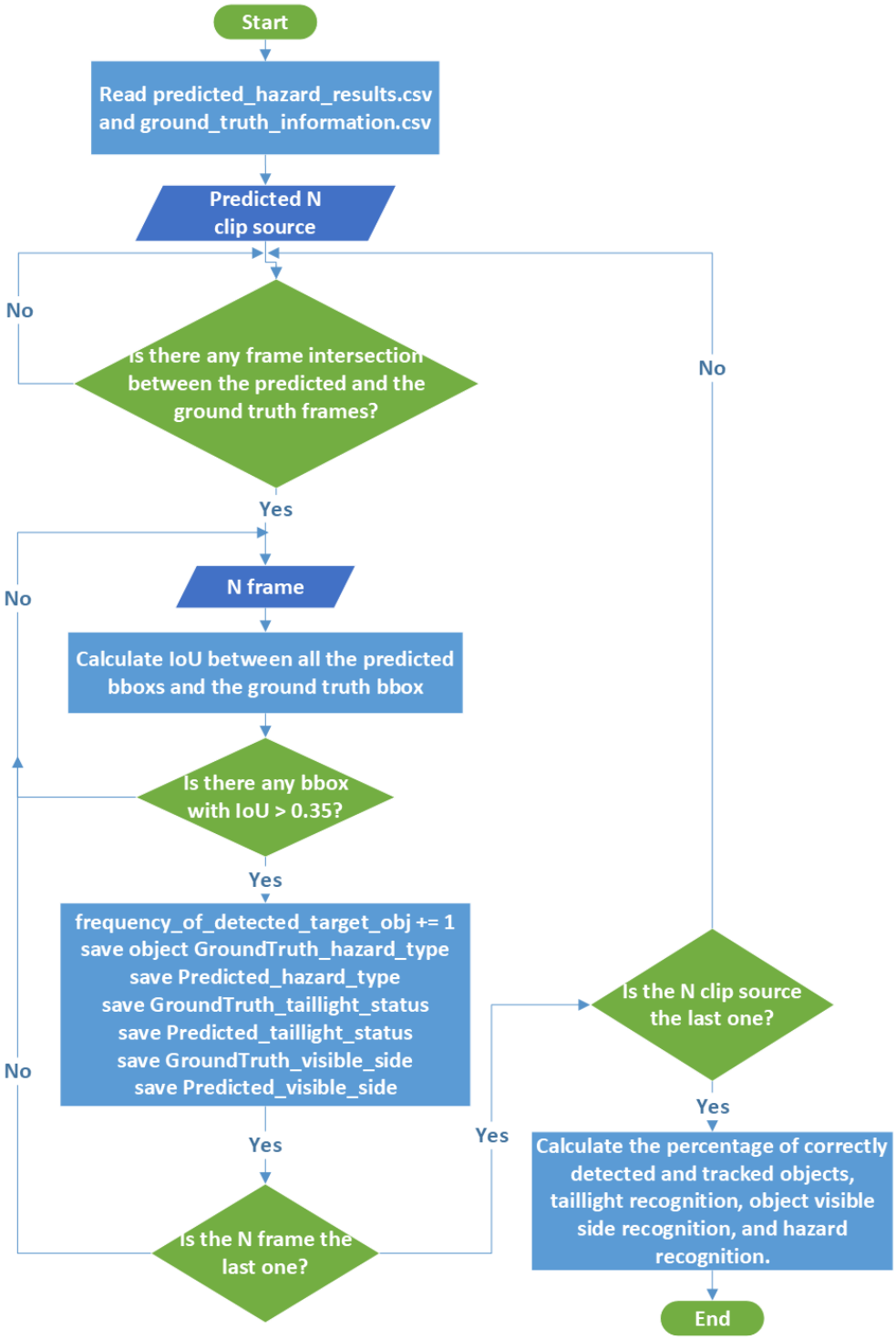


Figure 7.5: Flow diagram to calculate the corrected detection, tracking, and recognition accuracy.

size and diversity of object types. This result aligns with expectations, considering that YOLOv9e's pre-trained weights achieve an accuracy of 72.8%. While the detection ratio was reasonable, the accuracy was insufficient for deployment inIV systems, which require higher reliability due to their safety-critical nature. Furthermore, since detection and tracking modules typically serve as the initial stage in

Table 7.1: Complete Pipeline results when using different input types, models, and different numbers of classes.

Input Type	Model	N. Of. Classes	TODA[b](%)	RLSRA[c](%)	VSRA[d](%)	PTHERA[e](%)	PTHERATV[f](%)
Explicit	Embedding_TA_LSTM	10	69.87	75.0	70.0	51.0	76.5
Single Image	CNN_LSTM	10	69.87	75.0	70.0	60.0	65.8
Single Image	CNN_LSTM	14	67.19	74.0	71.0	55.0	64.1
Explicit + Single Image	Embedding_CNN_LSTM	10	69.87	75.0	70.0	54.0	79.9
Explicit + Single Image	Embedding_CNN_LSTM	14	67.19	74.0	71.0	52.0	79.7

[a]Frame Ratio, [b]Target Object Detection Accuracy, [c]Rear Light Status Recognition Accuracy, [d]Visible Side Recognition Accuracy, [e]Potential Traffic Hazard Event Recognition Accuracy (using predicted values), [f]Potential Traffic Hazard Event Recognition Accuracy with True Values.

a multi-step process, any inaccuracies at this stage can propagate and negatively impact subsequent modules.

Rear Light Status Recognition Performance

The RLSRA ranged between 74% and 75%. Datasets with more hazard classes, such as the 'motion towards' and 'all classes' sets, yielded the lowest accuracy. This outcome is consistent with expectations, given that the pre-trained model for rear light recognition has a baseline accuracy of 77%.

Object Visible Side Recognition Performance

The object VSRA achieved by the complete pipeline ranged between 70.0% and 71.0%, which is lower than expected. This is notable considering that the best-performing pre-trained model for visible side recognition achieved an accuracy of 83.0%.

Hazard Type Recognition Performance

The CNN LSTM model performed best in the potential traffic hazard event recognition task when using predicted values in the 'motion towards' and 'all classes' datasets. However, the Embedding CNN LSTM model outperformed others when using actual ground truth values. Notably, models incorporating rear light status and object visible side information—such as the Embedding LSTM and Embedding CNN LSTM—exhibited the largest discrepancies in accuracy between predicted and actual input values.

For example, the Embedding LSTM model achieved 76.5% accuracy with true values in the 'motion towards' set but only 51.0% with predicted values, resulting in a significant drop of 25.5%. Similarly, the Embedding CNN LSTM model showed a difference of 25.4%. In contrast, the CNN LSTM model exhibited a much smaller difference of 5.8%. This trend was also observed in the 'all classes' set, where the accuracy differences for the CNN LSTM and Embedding LSTM models were 9.1% and 27.7%, respectively.

These results indicate that while true rear light status and visible side values can significantly enhance the accuracy of potential traffic hazard event recognition, inaccuracies in these predictions can drastically degrade performance. This sensitivity underscores the importance of accurate rear light status and visible side recognition for effective hazard detection. On the other hand, although the CNN LSTM model may underperform compared to others when using true values, it demonstrates greater robustness to inaccuracies in predicted inputs.

7.6.1 Error Propagation, Interpretability, and Transferability

An important observation in this study is the propagation of errors across pipeline stages. Due to the modular architecture—starting with detection and tracking, followed by rear light and visible side recognition, and concluding with hazard classification—errors introduced at early stages have a cascading effect on downstream modules. For example, misdetections reduce the number of correctly identified target objects, leading to missing or erroneous inputs for the hazard classification task. This was quantitatively demonstrated by significant drops in hazard recognition accuracy (up to 27.7%) when using predicted versus ground truth inputs. These findings underscore the compounding effect of early-stage inaccuracies on final hazard prediction performance.

Regarding interpretability and explainability, interpretability refers to how easily humans can understand the internal mechanics of a model. Simple models like linear regression offer direct interpretability through their parameters. However, the deep learning models used in this pipeline—such as CNN-LSTM architectures for hazard classification—are inherently complex and non-transparent. Explainability, although related, focuses on the ability to provide understandable reasons for specific model decisions, even when the internal workings are opaque. While the current system is modular and transparent at the architectural level, it remains a *black box* in terms of understanding the rationale behind individual predictions.

In terms of transferability, the reliance on pre-trained models—particularly YOLO-v9e trained on the COCO dataset—presents challenges for generalisation. Many traffic hazard-relevant object categories are underrepresented or absent in COCO, leading to suboptimal detection performance in this context. Additionally, the rear light and visible side recognition modules may not generalise well across varied environmental conditions, such as changes in lighting, occlusion, or camera perspectives. These limitations highlight the system's susceptibility to **domain shift**, where differences between training and deployment environments can significantly degrade performance.

7.7 Conclusions

This chapter evaluated the performance of a complete potential traffic hazard event recognition pipeline, analysing the impact of each module on overall system performance.

The system achieved moderate accuracy across detection, rear light status recognition, visible side recognition, and hazard classification tasks. However, the results highlight that inaccuracies—particularly in the early detection stage—substantially degrade overall system performance. Models that leverage auxiliary cues, such as rear light status and visible side information, can improve hazard recognition accuracy but remain highly sensitive to upstream errors.

The study also identified several limitations, particularly related to the use of pre-trained models not optimized for traffic-specific contexts. Future work should focus on retraining detection models using domain-specific datasets, improving interpretability, and enhancing the robustness and generalizability of the system across diverse real-world scenarios.

To improve interpretability and explainability, techniques such as SHAP or Grad-CAM could be employed to increase transparency and trust, especially for deployment in safety-critical applications. Enhancing transferability can be achieved through methods such as domain adaptation, fine-tuning models with domain-specific data, and applying data augmentation techniques to expose models to a broader range of real-world conditions.

Chapter 8

Conclusions and Future Work

This thesis addresses the limited capability of existing IV systems to navigate safely in complex urban environments by enabling them to recognise traffic agent behaviours that may lead to potential traffic hazard events. To achieve this goal, a novel dataset was introduced, specifically designed to capture the complexity of urban traffic scenarios and to support the recognition of various discrete intention behaviours exhibited by different types of traffic agents.

In addition to the dataset, the thesis proposed and investigated two novel input features to enhance traffic agent behaviour recognition: the vehicle's rear light status and the object's visible side—one of which has not been previously studied. Both features were shown to significantly improve recognition performance. These contributions were used to train and evaluate various machine learning models for their ability to recognise potential traffic hazard events.

Finally, a complete potential hazard event recognition system was developed to assess how system performance is affected when ground truth values—such as detection, tracking, rear light status, and object visible side information—are not available.

This chapter summarises the work conducted in this thesis by evaluating each research question and objective, and by suggesting directions for future work.

8.1 Research Questions and Objectives Evaluation

This thesis addressed five research questions and outlined several objectives aimed at enhancing the ability of IV systems to navigate complex traffic scenarios. This section evaluates the extent to which these research questions and objectives have been fulfilled.

Research Question 1, Objectives 1 and 2 – Limitations of Existing Datasets

To address the research question, *"What are the limitations of existing datasets used to study complex traffic agent behaviour in complex scenarios? How can these limitations be addressed?"*, and to fulfil the objectives of *"Explore existing methods to recognise and predict the behaviour of traffic agents, along with the detection of traffic hazard events within traffic scenarios,"* and *"Create a Road Traffic Hazard*

Dataset," a comprehensive literature review was conducted and a novel traffic hazard dataset was developed.

The literature review revealed significant limitations in existing datasets, particularly their inability to represent a wide range of discrete traffic agent intention behaviours. The novel dataset introduced in this thesis offers a more diverse and detailed representation of traffic events, supporting tasks such as detection, tracking, rear light status recognition, object visible side recognition, and potential traffic hazard event recognition. However, the dataset suffers from class imbalance due to the rarity of certain behaviours. While it includes more classes than existing datasets, some behaviours—such as those caused by health impairment or intoxication—remain unrepresented. Thus, while the research question has been addressed and the objectives fulfilled, further improvements are possible.

Research Question 2, Objectives 5 and 6 – Improvement Through New Types of Input

The implementation of the proposed vehicle intention and hazard recognition systems aimed to answer the research question: *"Does traffic agent behaviour recognition algorithm performance improve with the introduction of new input features (e.g., object visible side relative to the EV, different input image representations, explicit features)?"* This also addressed the objectives of *"Develop, implement, and evaluate a Vehicle Intention Prediction algorithm,"* and *"Develop, implement, and evaluate a Potential Traffic Hazard Event Recognition algorithm."*

The results showed that incorporating overlooked cues—such as rear light status and object visible side—significantly improved the performance of behaviour recognition models, especially when these cues were accurately predicted. This was confirmed by comparing system performance using both predicted and ground truth values.

Regarding input image representations, the study found that local context images were most relevant to model accuracy. Combining multiple image types did not yield further improvements, suggesting that novel representations did not enhance performance. While relying solely on local context has limitations, integrating relevant global context remains a challenge.

The best results were achieved by combining explicit and implicit features, highlighting the importance of leveraging both in designing effective recognition algorithms.

Research Question 3, Objectives 3 and 4 – Machine Learning for Overlooked Cues Recognition

The rear light status and object visible side recognition systems were developed to address the research question: *"Can machine learning algorithms, using an appropriate dataset, recognise different types of rear light statuses and object visible sides?"* This also fulfilled the objectives of *"Develop, implement, and evaluate a Vehicle Rear Light Signal Recognition algorithm,"* and *"Develop, implement, and evaluate a Target Object Visible Side Recognition algorithm."*

The findings indicate that modern deep learning techniques can reasonably recognise rear light status and object visible sides. However, the achieved performance

is not yet sufficient for reliable use in a complete hazard recognition pipeline, as inaccuracies in these modules significantly degrade overall system performance.

Rear light misclassifications were often caused by factors such as sunlight reflection, occlusion, similar light patterns, object positioning, disproportionate light size, proximity of lights, weak brightness, and confusion between front and rear views. Visible side misclassifications were mainly due to the difficulty in distinguishing between similar classes, such as rear right and rear sides.

Research Question 4, Objective 6 – Machine Learning for Potential Traffic Hazard Event Recognition

The development of hazard recognition models addressed the research question: *"Can machine learning algorithms, with an appropriate traffic hazard dataset, recognise different types of traffic hazard events in complex urban scenarios?"* This also fulfilled the objective of *"Develop, implement, and evaluate a Potential Traffic Hazard Event Recognition algorithm."*

The results demonstrated that machine learning-based hazard recognition systems perform well with a limited number of hazard categories but show reduced accuracy as the number of categories increases. While current models can achieve high accuracy (up to 90%) in simplified scenarios, their performance declines in complex, real-world urban environments.

Qualitative analysis revealed that many misclassifications were due to poor image quality caused by lighting conditions and the distance between the target object and the EV. These issues, also noted in related studies, may be mitigated through sensor fusion techniques, such as multi-focal cameras, infrared imaging, radar, or LIDAR.

Although the proposed model implicitly considered neighbouring agents through global context images, no significant performance gains were observed. Future work should explore methods that explicitly model interactions among traffic agents.

Research Question 5, Objective 7 – Performance in a Complete Pipeline System

The complete hazard recognition pipeline was developed to address the research question: *"How is the behaviour recognition algorithm performance affected when applied to a complete pipeline system?"* and to fulfil the objective of *"Develop, implement, and evaluate a Complete Potential Traffic Hazard Event Recognition Pipeline System."*

The findings show that system performance significantly declines when using predicted values for detection, tracking, rear light status, and visible side recognition, due to error propagation across modules. For instance, if target objects are not accurately detected and tracked, subsequent modules relying on this information may fail. In this context, models like CNN LSTM, which rely solely on image inputs, demonstrated greater robustness and may be preferable.

Overall, the results highlight the challenges of enabling IVs to recognise potential traffic hazard events, particularly when multiple discrete behaviour categories are involved. The study emphasises the need for research that addresses a broader range of behaviours, as real-world scenarios are inherently diverse. Moreover, the complete pipeline evaluation shows that even well-studied tasks like object detection

and tracking require further improvement, given their critical impact on downstream modules.

8.2 Limitations

While this thesis has made significant contributions, several inherent limitations must be acknowledged, particularly regarding dataset size, causal inference, modelling performance, and system complexity.

Dataset Limitations: Despite the introduction of a novel traffic hazard dataset designed to represent a diverse range of traffic agent behaviours, certain behaviour categories remain underrepresented due to their natural rarity (e.g., driver impairment or unusual pedestrian behaviours). This class imbalance can affect model performance and generalisation, particularly for rare hazard events, and poses a challenge for achieving consistent accuracy across all classes. Moreover, although the dataset is larger than many existing ones, it may still be insufficient for training highly complex deep learning models, which typically require extensive data to avoid overfitting and to improve generalisability.

Modelling Performance: Although several machine learning models developed in this thesis demonstrated improved recognition accuracy when leveraging novel features such as rear light status and object visible side, their performance remains suboptimal for real-world deployment. For instance, error propagation through the pipeline reduced the accuracy of potential hazard event recognition by up to 27.7%. Furthermore, models with performance metrics such as recall or precision in the 70–80% range are not sufficient for safety-critical applications like autonomous driving, where near-perfect reliability is required. These results indicate that further improvements in detection, tracking, and behaviour recognition are necessary before practical implementation.

Causal Inference: The methodologies employed in this research rely primarily on observational data and machine learning techniques, which identify correlations between input features and behaviour classes. However, these models cannot establish causal relationships. As a result, while they can predict potential hazard events based on patterns in the data, they cannot explain causality with certainty. This limits the interpretability and explainability of the system, particularly in scenarios where understanding the underlying causes of specific behaviours is essential for decision-making.

Transferability: The models developed in this thesis, particularly those reliant on pre-trained detection systems such as YOLOv9e trained on the COCO dataset, face challenges in generalising to new environments. Domain shifts—such as differences in road structures, vehicle types, or lighting conditions—can degrade model performance. This limitation underscores the need for domain adaptation techniques and further data collection efforts to improve the robustness and transferability of the proposed system.

System Complexity and Computational Constraints: The modular pipeline approach, while beneficial for analysing the impact of individual components, also introduces complexity that exacerbates error propagation. Additionally, applying the behaviour recognition system to every observed traffic agent is computationally infeasible in real-world scenarios due to resource constraints.

8.3 Future Works

While this study has made significant contributions to discrete behaviour recognition, several areas remain open for future research:

1. **Dataset Enhancement**: Although the proposed novel dataset captures many potential traffic hazard events in real-world traffic scenes, some event types are underrepresented or entirely absent. Future work should focus on collecting more samples for these categories. This could involve enhancing the proposed hazard recognition system to automatically extract additional samples from other traffic datasets, followed by manual verification. Additionally, research should explore methods for collecting data on unusual behaviours, such as those exhibited by pedestrians or drivers under the influence of substances or experiencing health impairments.

2. **Improving Recognition Algorithms**: Misclassifications in rear light status and object visible side recognition are often caused by poor image quality, occlusion, object angle, distance, and class similarity. Advanced techniques should be explored to improve these recognition tasks. For instance, using segmented portions of the object instead of the full bounding box could reduce occlusion effects and focus attention on relevant features. Combining visible side information with rear light status may also help mitigate errors caused by object orientation. Additionally, investigating headlight status recognition—an overlooked cue—could provide valuable insights, as headlights may indicate the intentions of oncoming vehicles.

3. **Incorporating Traffic Agent Interactions**: The current vehicle intention and hazard recognition models do not account for interactions between traffic agents. Future research should explore techniques such as graph neural networks or Voronoi diagrams to model these interactions, which can significantly influence a target object's behaviour and intention.

4. **Expanding the Scope of Data Sources**: Most behaviour recognition studies rely on front-facing dash camera footage. However, relevant behaviours may occur outside this field of view. For example, a vehicle cutting in front of the EV may initiate its manoeuvre from the rear or side. Future work should investigate the use of additional cameras placed at the rear and sides of the vehicle to capture a more comprehensive view of traffic agent behaviour.

5. **Target Object Selection**: Applying the hazard recognition system to every traffic agent is computationally impractical, as many agents (e.g., parked vehicles, stationary bicycles, pedestrians on sidewalks) are non-hazardous. Future research should develop methods to prioritise and select relevant objects before applying behaviour recognition. For instance, in real-world scenarios, the EV's intended path is known and can be used to identify relevant target objects. However, most existing datasets lack this trajectory information.

These recommendations aim to address current limitations and enhance the capabilities of IV systems, enabling them to navigate complex urban environments more safely and efficiently.

8.4 Summary

In conclusion, this thesis has made significant contributions to advancing IV systems by improving their ability to recognise potential traffic hazards. The novel dataset and input features introduced have enhanced the recognition of various discrete intention behaviours exhibited by different types of traffic agents. However, challenges remain, particularly regarding dataset scope and the need for further improvements in recognition algorithms and complete system pipelines. Future research directions have been outlined to address these challenges, paving the way for more effective and comprehensive solutions in IV systems.

Appendix A

Rear Light Status Miss-classification Examples

This appendix provides supplementary information to support the qualitative analysis of the rear light status recognition algorithm results discussed in Chapter 4. It includes examples where the actual rear light status was misclassified into different classes. These examples represent a small sample, focusing on the classes with the highest number of misclassifications.

Figure A.1 shows instances where the OLR class was misclassified as the OOO class.

(a) PRED:OOO (b) PRED:OOO

Figure A.1: Examples of OLR misclassified as OOO.

Figure A.2 shows instances where the BOO class was misclassified as other classes.

Figure A.3 shows instances where the OLO class was misclassified as other classes.

Figure A.4 depicts instances where the OOO class has been misclassified by the other classes.

Figure A.5 shows instances where the OOR class was misclassified as other classes.

Figure A.6 shows instances where the UNK class was misclassified as other classes.

(a) PRED:OLO

(b) PRED:OLO

(c) PRED:OLR

(d) PRED:OLR

(e) PRED:OOO

(f) PRED:OOO

(g) PRED:OOO

(h) PRED:OOR

(i) PRED:OOR

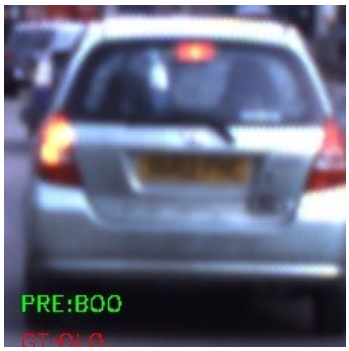
(j) PRED:OOR

(k) PRED:OOR

(l) PRED:OOR

Figure A.2: Examples of BOO misclassified as other classes.

(a) PRED:BOO



(b) PRED:BOO

(c) PRED:BOO

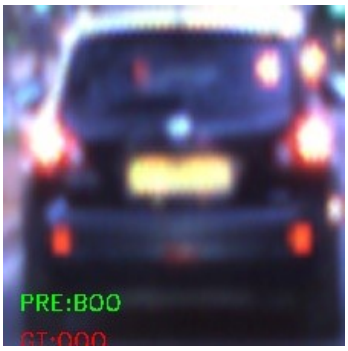
(d) PRED:BOO

(e) PRED:OOO

(f) PRED:OOO

(g) PRED:OOO

Figure A.3: Examples of OLO misclassified as other classes.



(a) PRED:BOO

(b) PRED:BOO

(c) PRED:BOO

(d) PRED:OLO

(e) PRED:OLO

(f) PRED:OLR

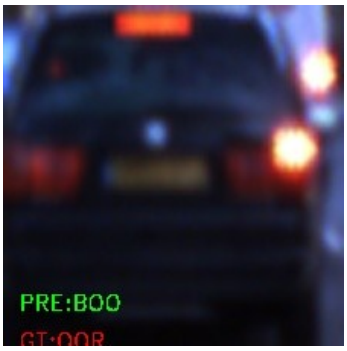
(g) PRED:OOR

(h) PRED:UNK

(i) PRED:UNK

Figure A.4: Examples of OOO misclassified as other classes.

(a) PRED:BOO



(b) PRED:BOO

(c) PRED:OOO

(d) PRED:OOO

(e) PRED:OOO

(f) PRED:OOO

Figure A.5: Examples of OOR misclassified as other classes.

(a) PRED:BOO (b) PRED:OLO (c) PRED:OLO

(d) PRED:OOO (e) PRED:OOO (f) PRED:OOO

(g) PRED:OOO (h) PRED:OOO (i) PRED:OOR

Figure A.6: Examples of UNK misclassified as other classes.

Appendix B

Object Visible Side Miss-classification Examples

This appendix provides supplementary information to support the qualitative analysis of the object visible side recognition algorithm results discussed in Chapter 4. It presents examples where the actual visible side of an object was misclassified as a different class. These instances represent a small subset of the dataset, with emphasis on the classes that exhibited the highest number of misclassifications.

Figure B.1 illustrates cases where the `front_left_side` class was incorrectly classified as other classes.

(a) PRED:front_side (b) PRED:front_side (c) PRED:front_side

(d) PRED:right_side (e) PRED:right_side (f) PRED:right_side

Figure B.1: Examples of misclassifications involving the `front_left_side` class.

Figure B.2 illustrates instances where the `front_right_side` class was misclassified as the `front_side` class.

(a) PRED:front_side

(b) PRED:front_side

Figure B.2: Examples of misclassifications involving the `front_right_side` class.

Figure B.3 illustrates instances where the `front_side` class was misclassified as other classes.

(a) PRED:left_side

(b) PRED:left_side

(c) PRED:rear_left_side

Figure B.3: Examples of misclassifications involving the `front_side` class.

Figure B.4 illustrates instances where the `left_side` class was misclassified as other classes.

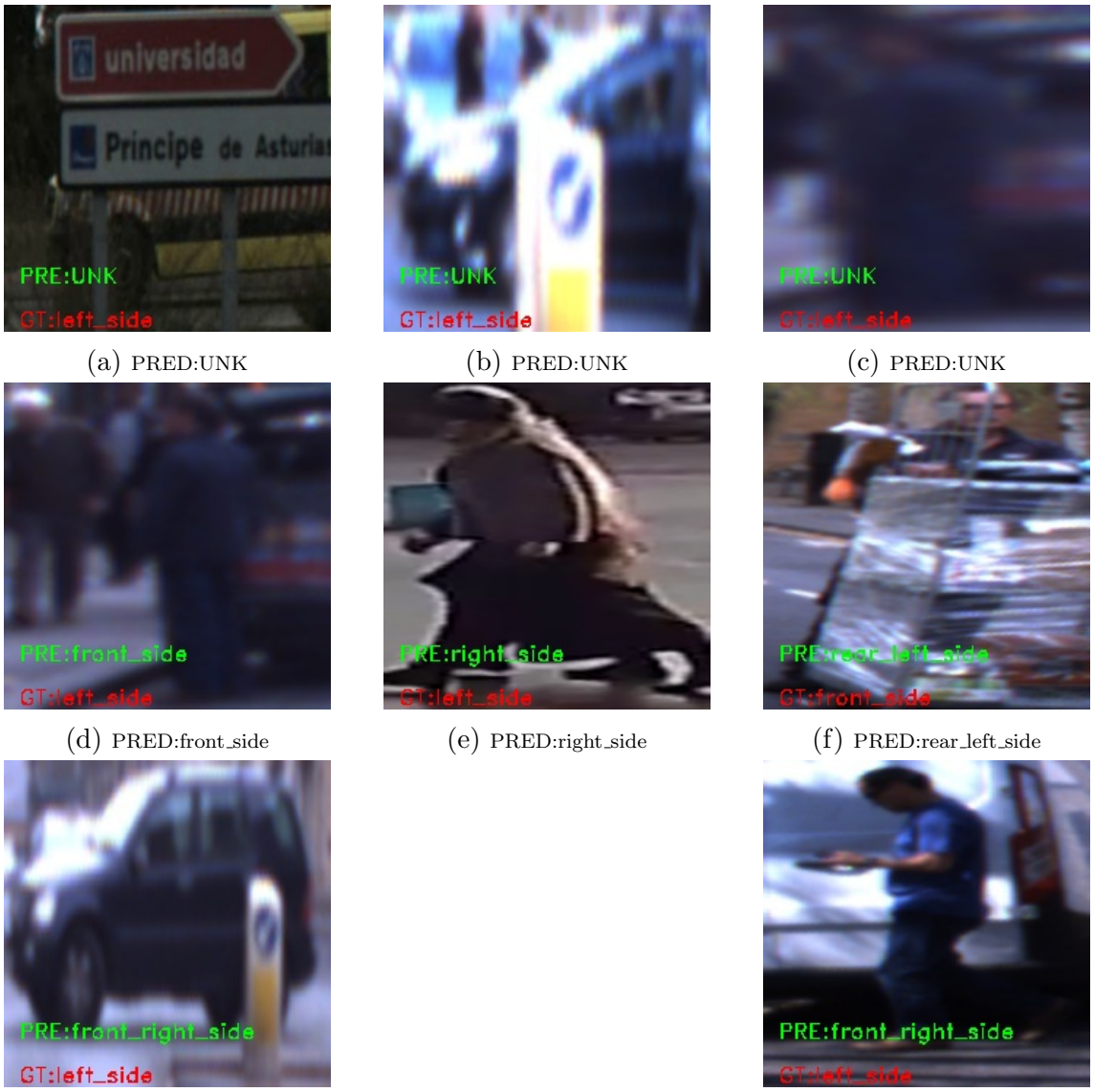
Figure B.5 illustrates instances where the `rear_left_side` class was misclassified as other classes.

Figure B.6 illustrates instances where the `rear_right_side` class was misclassified as other classes.

Figure B.7 illustrates instances where the `rear_side` class was misclassified as other classes.

Figure B.8 illustrates instances where the `right_side` class was misclassified as other classes.

Figure B.9 illustrates instances where the `UNK` class was misclassified as other classes.



(a) PRED:UNK (b) PRED:UNK (c) PRED:UNK

(d) PRED:front_side (e) PRED:right_side (f) PRED:rear_left_side

(g) PRED:front_right_side (h) PRED:front_right_side

Figure B.4: Examples of misclassifications involving the `left_side` class.

(a) PRED:rear_side (b) PRED:rear_side (c) PRED:rear_side

(d) PRED:rear_side

Figure B.5: Examples of misclassifications involving the `rear_left_side` class.

(a) PRED:rear_side (b) PRED:rear_side (c) PRED:rear_side

(d) PRED:rear_side (e) PRED:rear_side (f) PRED:right_side

(g) PRED:right_side (h) PRED:right_side (i) PRED:right_side

Figure B.6: Examples of misclassifications involving the `rear_right_side` class.

(a) PRED:rear_left_side (b) PRED:rear_left_side (c) PRED:rear_left_side

(d) PRED:rear_left_side (e) PRED:rear_left_side (f) PRED:rear_right_side

(g) PRED:rear_right_side (h) PRED:rear_right_side (i) PRED:rear_right_side

Figure B.7: Examples of misclassifications involving the `rear_side` class.

(a) PRED:left_side

(b) PRED:left_side

(c) PRED:left_side

Figure B.8: Examples of misclassifications involving the `right_side` class.

(a) PRED:rear_side

(b) PRED:right_side

(c) PRED:right_side

(d) PRED:rear_right_side

(e) PRED:rear_right_side

(f) PRED:rear_right_side

Figure B.9: Examples of misclassifications involving the `UNK` class.

Bibliography

[1] Geneva WHO. *Global status report on road safety 2023*. en. 2023. URL: https://www.who.int/publications-detail-redirect/9789240086517 (visited on 12/13/2023).

[2] GOVUK GOVUK. *Reported road casualties Great Britain, annual report: 2020*. en. Publication Title: GOV.UK. 2020. URL: https://www.gov.uk/government/statistics/reported-road-casualties-great-britain-annual-report-2020/reported-road-casualties-great-britain-annual-report-2020 (visited on 03/29/2022).

[3] GOVUK GOVUK. *Reported road casualties in Great Britain, provisional estimates: year ending June 2021*. en. Publication Title: GOV.UK. 2021. URL: https://www.gov.uk/government/statistics/reported-road-casualties-in-great-britain-provisional-estimates-year-ending-june-2021/reported-road-casualties-in-great-britain-provisional-estimates-year-ending-june-2021 (visited on 03/29/2022).

[4] General Directorate. *Road safety: 20,640 people died in a road crash last year – progress remains too slow*. en. 2023. URL: https://transport.ec.europa.eu/news-events/news/road-safety-20640-people-died-road-crash-last-year-progress-remains-too-slow-2023-10-19_en (visited on 12/13/2023).

[5] Claire Perry. *The Pathway to Driverless Cars: Summary report and action plan*. en. United Kingdom Department for Transport, 2015. ISBN: 978-1-84864-153-2. URL: https://nls.ldls.org.uk/welcome.html?ark:/81055/vdc%5C_100063396695.0x000001 (visited on 10/08/2020).

[6] Roger L. McCarthy. "Autonomous vehicle accident data analysis: California ol 316 reports: 2015–2020". In: *ASCE-ASME Journal of Risk and Uncertainty in Engineering Systems, Part B: Mechanical Engineering* 8.3 (2022). Publisher: American Society of Mechanical Engineers, p. 034502. URL: https://asmedigitalcollection.asme.org/risk/article-abstract/8/3/034502/1114602 (visited on 12/14/2023).

[7] Trevor Reed. "INRIX Global Traffic Scorecard". In: (2019).

[8] Arun Chand, S. Jayesh, and A. B. Bhasi. "Road traffic accidents: An overview of data sources, analysis techniques and contributing factors". In: *Materials Today: Proceedings* 47 (2021). Publisher: Elsevier, pp. 5135–5141. URL: https://www.sciencedirect.com/science/article/pii/S2214785321040153 (visited on 12/12/2023).

[9] World Health Organization WHO. *Global status report on road safety 2018: Summary*. Tech. rep. World Health Organization, 2018.

[10] Anna Goodman and James Cheshire. "Inequalities in the London bicycle sharing system revisited: impacts of extending the scheme to poorer areas but then doubling prices". In: *Journal of Transport Geography* 41 (Dec. 2014), pp. 272–279. ISSN: 0966-6923. DOI: 10.1016/j.jtrangeo.2014.04.004. URL: https://www.sciencedirect.com/science/article/pii/S0966692314000659 (visited on 02/22/2024).

[11] Ezgi Eren and Volkan Emre Uz. "A review on bike-sharing: The factors affecting bike-sharing demand". In: *Sustainable cities and society* 54 (2020). Publisher: Elsevier, p. 101882. URL: https://www.sciencedirect.com/science/article/pii/S2210670719312387 (visited on 02/22/2024).

[12] Tomasz Bieliński and Agnieszka Ważna. "Electric scooter sharing and bike sharing user behaviour and characteristics". In: *Sustainability* 12.22 (2020). Publisher: MDPI, p. 9640. URL: https://www.mdpi.com/2071-1050/12/22/9640 (visited on 02/22/2024).

[13] Jan CT Bieser et al. "Impacts of telecommuting on time use and travel: A case study of a neighborhood telecommuting center in Stockholm". In: *Travel Behaviour and Society* 23 (2021). Publisher: Elsevier, pp. 157–165. URL: https://www.sciencedirect.com/science/article/pii/S2214367X20302441 (visited on 02/22/2024).

[14] Ugo Lachapelle, Georges A Tanguay, and Léa Neumark-Gaudet. "Telecommuting and sustainable travel: Reduction of overall travel time, increases in non-motorised travel and congestion relief?" en. In: *Urban Studies* 55.10 (Aug. 2018), pp. 2226–2244. ISSN: 0042-0980, 1360-063X. DOI: 10.1177/0042098017708985. URL: http://journals.sagepub.com/doi/10.1177/0042098017708985 (visited on 02/22/2024).

[15] Andrea Pompigna and Raffaele Mauro. "Smart roads: A state of the art of highways innovations in the Smart Age". In: *Engineering Science and Technology, an International Journal* 25 (2022). Publisher: Elsevier, p. 100986. URL: https://www.sciencedirect.com/science/article/pii/S2215098621000872 (visited on 02/22/2024).

[16] Chenglong Liu et al. "New generation of smart highway: Framework and insights". In: *Journal of Advanced Transportation* 2021 (2021). Publisher: Hindawi Limited, pp. 1–12. URL: https://www.hindawi.com/journals/jat/2021/9445070/ (visited on 02/22/2024).

[17] Jonas Eliasson. *The Stockholm congestion charges: an overview*. Centre for Transport Studies Stockholm, Sweden, 2014. URL: https://www.transportportal.se/swopec/CTS2014-7.pdf (visited on 02/22/2024).

[18] Colin P. Green, John S. Heywood, and Maria Navarro Paniagua. "Did the London congestion charge reduce pollution?" In: *Regional Science and Urban Economics* 84 (2020). Publisher: Elsevier, p. 103573. URL: https://www.sciencedirect.com/science/article/pii/S0166046220302581 (visited on 02/22/2024).

[19] Maria Börjesson and Ida Kristoffersson. "The Swedish congestion charges: Ten years on". In: *Transportation Research Part A: Policy and Practice* 107 (2018). Publisher: Elsevier, pp. 35–51. URL: https://www.sciencedirect.com/science/article/pii/S0965856417300782 (visited on 02/22/2024).

[20] Marianna COLONNA. *Urbanisation worldwide*. en. Publication Title: Knowledge for policy - European Commission Type: Text. Oct. 2018. URL: https://ec.europa.eu/knowledge4policy/foresight/topic/continuing-urbanisation/urbanisation-worldwide%5C_en (visited on 10/06/2020).

[21] Andrew Hart and Carrie Cox. *How autonomous vehicles could relive or worsen traffic congestion*. English. Tech. rep. Berlin: SBD HERE, 2017, p. 18. URL: https://www.here.com/sites/g/files/odxslz166/files/2018-12/HERE%5C_How%5C_autonomous%5C_vehicles%5C_could%5C_relieve%5C_or%5C_worsen%5C_traffic%5C_congestion%5C_white%5C_paper.pdf (visited on 10/05/2020).

[22] Bruce Mehler et al. "Evaluating technologies relevant to the enhancement of driver safety". In: (2014).

[23] Tesla Tesla. *Autopilot and Full Self-Driving Capability \textbar Tesla Support*. en. Publication Title: Tesla. 2023. URL: https://www.tesla.com/support/autopilot (visited on 12/13/2023).

[24] Waymo Waymo. *Waymo Safety Report*. en. Publication Title: Waymo. Sept. 2020. URL: https://waymo.com/safety/ (visited on 10/30/2020).

[25] Baidu. *Baidu Research*. 2023. URL: http://research.baidu.com/Research_Areas/index-view?id=58 (visited on 12/13/2023).

[26] Tecent Tecent. *Urban Mobility Reinvented: Tencent's Autonomous Driving Road Paves the Way to Greener Cities*. en-US. 2023. URL: https://www.tencent.com/en-us/articles/2201669.html (visited on 12/13/2023).

[27] Mercedes-Benz Group. *Autonomous Driving*. en. Publication Title: Mercedes-Benz Group. 2023. URL: https://group.mercedes-benz.com/innovation/product-innovation/autonomous-driving/ (visited on 12/13/2023).

[28] BMW BMW. *Autonomous Driving – five steps to the self-driving car*. en. 2023. URL: https://www.bmw.com/en/automotive-life/autonomous-driving.html (visited on 12/13/2023).

[29] VW VW. *Autonomous Driving \textbar Electric Car Tech \textbar Volkswagen UK*. en. 2023. URL: https://www.volkswagen.co.uk/en/electric-and-hybrid/software-and-technology/driving-technology/autonomous-driving.html (visited on 12/13/2023).

[30] Toyota Toyota. *Automated Driving — Toyota Europe*. en. URL: https://www.toyota-europe.com/innovation/zero-accidents/automated-driving (visited on 12/13/2023).

[31] Honda. *Honda Global \textbar October 19, 2023 Honda, GM and Cruise Plan to Begin Driverless Ridehail Service in early 2026 – The Three Companies Enter Memorandum of Understanding Aiming to Establish a JV Company to Provide Driverless Ridehail Service in Japan –*. en. 2023. URL: https://global.honda/en/newsroom/news/2023/c231019aeng.html (visited on 12/13/2023).

[32] Hyundai Hyundai. *Autonomous \textbar Why Hyundai \textbar Hyundai Australia*. 2023. URL: https://www.hyundai.com/au/en/why-hyundai/autonomous-driving (visited on 12/13/2023).

[33] KIA KIA. *Autonomous \textbar Mobility ACE \textbar Kia Australia*. 2023. URL: https://www.kia.com/au/discover-kia/innovation/mobility-ace/autonomous.html (visited on 12/13/2023).

[34] Y Huang and Y Chen. "Autonomous driving with deep learning: A survey of state-of-art technologies. arXiv 2020". In: *arXiv preprint arXiv:2006.06091* (2006).

[35] Jessica Van Brummelen et al. "Autonomous vehicle perception: The technology of today and tomorrow". In: *Transportation research part C: emerging technologies* 89 (2018), pp. 384–406.

[36] Abdul Sajeed Mohammed et al. "The perception system of intelligent ground vehicles in all weather conditions: A systematic literature review". In: *Sensors* 20.22 (2020), p. 6532.

[37] Brad Templeton. *Former Head Of Tesla AI Explains Why They've Removed Sensors; Others Differ*. 2022. URL: https://www.forbes.com/sites/bradtempleton/2022/10/31/former-head-of-tesla-ai-explains-why-theyve-removed-sensors-others-differ/ (visited on 02/05/2025).

[38] Mohammed Chghaf, Sergio Rodriguez, and Abdelhafid El Ouardi. "Camera, LiDAR and multi-modal SLAM systems for autonomous ground vehicles: a survey". In: *Journal of Intelligent & Robotic Systems* 105.1 (2022), p. 2.

[39] Fei Liu, Zihao Lu, and Xianke Lin. "Vision-based environmental perception for autonomous driving". In: *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering* (2022), p. 09544070231203059.

[40] Khan Muhammad et al. "Vision-based semantic segmentation in scene understanding for autonomous driving: Recent achievements, challenges, and outlooks". In: *IEEE Transactions on Intelligent Transportation Systems* 23.12 (2022), pp. 22694–22715.

[41] Monirul Islam Pavel, Siok Yee Tan, and Azizi Abdullah. "Vision-based autonomous vehicle systems based on deep learning: A systematic literature review". In: *Applied Sciences* 12.14 (2022), p. 6831.

[42] Support Tesla. *Tesla Vision Update: Replacing Ultrasonic Sensors with Tesla Vision — Tesla Support*. 2024. URL: https://www.tesla.com/support/transitioning-tesla-vision (visited on 02/05/2025).

[43] Driver Waymo. *Waymo Driver*. en. Publication Title: Waymo. 2023. URL: https://waymo.com/waymo-driver/ (visited on 05/25/2023).

[44] Hisham Y Makahleh, Emma Jayne Sakamoto Ferranti, and Dilum Dissanayake. "Assessing the Role of Autonomous Vehicles in Urban Areas: A Systematic Review of Literature". In: *Future Transportation* 4.2 (2024), pp. 321–348.

[45] Kareem Othman. "Exploring the implications of autonomous vehicles: A comprehensive review". In: *Innovative Infrastructure Solutions* 7.2 (2022), p. 165.

[46] Dalma Zilahy and Gyula Mester. "Factors Affecting the Adoption of Self-Driving Cars". In: *Interdisciplinary Description of Complex Systems: INDECS* 22.6 (2024), pp. 732–737.

[47] Vikas Vyas and Zheyuan Xu. "Key Safety Design Overview in AI-driven Autonomous Vehicles". In: *arXiv preprint arXiv:2412.08862* (2024).

[48] Divya Garikapati and Sneha Sudhir Shetiya. "Autonomous Vehicles: Evolution of Artificial Intelligence and the Current Industry Landscape". In: *Big Data and Cognitive Computing* 8.4 (2024), p. 42.

[49] Wenhao Ding et al. "A survey on safety-critical driving scenario generation—A methodological perspective". In: *IEEE Transactions on Intelligent Transportation Systems* 24.7 (2023), pp. 6971–6988.

[50] Tianyue Feng et al. "Multimodal critical-scenarios search method for test of autonomous vehicles". In: *Journal of intelligent and connected vehicles* 5.3 (2022), pp. 167–176.

[51] Anis Boubakri et al. "High definition map update for autonomous and connected vehicles: A survey". In: *2022 International Wireless Communications and Mobile Computing (IWCMC)*. IEEE. 2022, pp. 1148–1153.

[52] Anastasios Giannaros et al. "Autonomous vehicles: Sophisticated attacks, safety issues, challenges, open topics, blockchain, and future directions". In: *Journal of Cybersecurity and Privacy* 3.3 (2023), pp. 493–543.

[53] Hussam Jaafar Kadhim and Amel Hussein Abbas. "A review: The Self-Driving Car's Requirements and The Challenges it Faces". In: *Mustansiriyah Journal of Pure and Applied Sciences* 2.1 (2024), pp. 135–150.

[54] Frank Kargl et al. "Privacy protection of automated and self-driving vehicles (dagstuhl seminar 22042)". In: (2022).

[55] Sakib Mahmud Khan et al. "Autonomous Vehicles for All?" In: *Journal on Autonomous Transportation Systems* 1.1 (2024), pp. 1–8.

[56] James M. Anderson et al. *Autonomous vehicle technology: A guide for policymakers*. Rand Corporation, 2014. URL: https://books.google.co.uk/books?hl=pt-BR&lr=&id=y0WrAgAAQBAJ&oi=fnd&pg=PP1&dq=current+state+of+autonomous+vehicle+technology&ots=-9E9d5FDOL&sig=Gy1dsdSNKBqiHsx6bwavh2skID0 (visited on 12/13/2023).

[57] Zongwei Liu et al. "An overview of the latest progress and core challenge of autonomous vehicle technologies". In: *MATEC Web of Conferences*. Vol. 308. EDP Sciences, 2020, p. 06002. URL: https://www.matec-conferences.org/articles/matecconf/abs/2020/04/matecconf_ictte2019_06002/matecconf_ictte2019_06002.html (visited on 12/13/2023).

[58] Matthew Schwall et al. "Waymo public road safety performance data". In: *arXiv preprint arXiv:2011.00038* (2020).

[59] Francesca M. Favarò et al. "Examining accident reports involving autonomous vehicles in California". In: *PLoS one* 12.9 (2017). Publisher: Public Library of Science San Francisco, CA USA, e0184952. URL: https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0184952 (visited on 12/15/2023).

[60] Junyao Guo, Unmesh Kurup, and Mohak Shah. "Is it safe to drive? An overview of factors, metrics, and datasets for driveability assessment in autonomous driving". In: *IEEE Transactions on Intelligent Transportation Systems* 21.8 (2019), pp. 3135–3151. URL: https://ieeexplore.ieee.org/abstract/document/8760560/?casa_token=OFejeVhcZPkAAAAA:SOPZvZg81bKhN1tvt9PnXVW3e_u7MTFcpfrCk7Ns75HN8JRZ4bhmTpmXpN6ll6vP5nNfEL07 (visited on 12/22/2023).

[61] Dannier Xiao et al. "Review of graph-based hazardous event detection methods for autonomous driving systems". In: *IEEE Transactions on Intelligent Transportation Systems* (2023). Publisher: IEEE. URL: https://ieeexplore.ieee.org/abstract/document/10038646/ (visited on 12/15/2023).

[62] John Dahl et al. "Collision avoidance: A literature review on threat-assessment techniques". In: *IEEE Transactions on Intelligent Vehicles* 4.1 (2018), pp. 101–113. URL: https://ieeexplore.ieee.org/abstract/document/8574961/?casa_token=5hJrY_zZDmkAAAAA:1N5qt38WiaYKn1EhPW36J86k1R2HbCVTn4VS2t-WtqfOeCNyfQCim8sVZhsbpoa7LY5BfY7n (visited on 12/22/2023).

[63] Francisca Rosique et al. "A systematic review of perception system and simulators for autonomous vehicles research". In: *Sensors* 19.3 (2019), p. 648.

[64] Sajjad Mozaffari et al. "Deep learning-based vehicle behavior prediction for autonomous driving applications: A review". In: *IEEE Transactions on Intelligent Transportation Systems* 23.1 (2020). Publisher: IEEE, pp. 33–47.

[65] Alberto Broggi et al. "Intelligent Vehicles". In: *Springer Handbook of Robotics*. Ed. by Bruno Siciliano and Oussama Khatib. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 1175–1198. ISBN: 978-3-540-30301-5. DOI: 10.1007/978-3-540-30301-5_52. URL: https://doi.org/10.1007/978-3-540-30301-5_52.

[66] Felipe Jiménez. *Intelligent Vehicles: Enabling technologies and future developments*. Butterworth-Heinemann, 2017.

[67] Phillip Czech et al. "On-Board Pedestrian Trajectory Prediction Using Behavioral Features". In: *arXiv preprint arXiv:2210.11999* (2022).

[68] Julian Francisco Pieter Kooij et al. "Context-based pedestrian path prediction". In: *European Conference on Computer Vision*. Springer, 2014, pp. 618–633.

[69] Mahdi Biparva et al. "Video action recognition for lane-change classification and prediction of surrounding vehicles". In: *arXiv preprint arXiv:2101.05043* (2021).

[70] Luiz G Galvão and M Nazmul Huda. "Pedestrian and vehicle behaviour prediction in autonomous vehicle system—A review". In: *Expert Systems with Applications* 238 (2024), p. 121983.

[71] Rubén Izquierdo et al. "The prevention dataset: a novel benchmark for prediction of vehicles intentions". In: *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*. IEEE, 2019, pp. 3114–3121.

[72] Wael Farag and Zakaria Saleh. "An advanced vehicle detection and tracking scheme for self-driving cars". In: *2nd Smart Cities Symposium (SCS 2019)*. IET. 2019, pp. 1–6.

[73] Namhoon Lee et al. "Desire: Distant future prediction in dynamic scenes with interacting agents". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 336–345.

[74] Mahir Gulzar, Yar Muhammad, and Naveed Muhammad. "A survey on motion prediction of pedestrians and vehicles for autonomous driving". In: *IEEE Access* 9 (2021). Publisher: IEEE, pp. 137957–137969.

[75] J. Joseph Antony and M. Suchetha. "Vision based vehicle detection: a literature review". In: *International Journal of Applied Engineering Research* 11.5 (2016), pp. 3128–3133.

[76] Iuliia Kotseruba, Amir Rasouli, and John K. Tsotsos. "Benchmark for evaluating pedestrian action prediction". In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 2021, pp. 1258–1268.

[77] Chien-Yao Wang, I-Hau Yeh, and Hong-Yuan Mark Liao. "You only learn one representation: Unified network for multiple tasks". In: *arXiv preprint arXiv:2105.04206* (2021).

[78] Mohammad Sadegh Aliakbarian et al. "VIENA \ˆ2\ : A Driving Anticipation Dataset". en. In: *Computer Vision – ACCV 2018*. Ed. by C. V. Jawahar et al. Vol. 11361. Cham: Springer International Publishing, 2019, pp. 449–466. DOI: 10.1007/978-3-030-20887-5_28. URL: http://link.springer.com/10.1007/978-3-030-20887-5_28 (visited on 01/08/2024).

[79] Shaozheng Yan et al. "Research on recognition algorithm of brake tail light based on monocular vision". In: *2019 Chinese Automation Congress (CAC)*. IEEE, 2019, pp. 2889–2893.

[80] Wenjie Song et al. "Action-State Joint Learning-Based Vehicle Taillight Recognition in Diverse Actual Traffic Scenes". In: *IEEE Transactions on Intelligent Transportation Systems* 23.10 (2022), pp. 18088–18099. DOI: 10.1109/TITS.2022.3160501.

[81] Han-Kai Hsu et al. "Learning to tell brake and turn signals in videos using cnn-lstm structure". In: *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2017, pp. 1–6.

[82] Mauricio Casares, Akhan Almagambetov, and Senem Velipasalar. "A robust algorithm for the detection of vehicle turn signals and brake lights". In: *2012 IEEE Ninth International Conference on Advanced Video and Signal-Based Surveillance*. IEEE, 2012, pp. 386–391.

[83] Zhiyong Cui, Shao-Wen Yang, and Hsin-Mu Tsai. "A vision-based hierarchical framework for autonomous front-vehicle taillights detection and signal recognition". In: *2015 IEEE 18th International Conference on Intelligent Transportation Systems*. IEEE, 2015, pp. 931–937.

[84] Jian-Gang Wang et al. "Appearance-based brake-lights recognition using deep learning and vehicle detection". In: *2016 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2016, pp. 815–820.

[85] Adrienne LaFrance. *Your Grandmother's Driverless Car*. en-US. Publication Title: The Atlantic. June 2016. URL: https://www.theatlantic.com/technology/archive/2016/06/beep-beep/489029/ (visited on 10/07/2020).

[86] Dean A. Pomerleau. "Alvinn: An autonomous land vehicle in a neural network". In: *Advances in neural information processing systems*. 1989, pp. 305–313.

[87] Qi Chen, Umit Ozguner, and Keith Redmill. "Ohio state university at the 2004 darpa grand challenge: developing a completely autonomous vehicle". In: *IEEE Intelligent Systems* 19.5 (2004), pp. 8–11.

[88] Rio Tinto. *Automated-truck-expansion-Pilbara*. en. Mar. 2018. URL: https://www.riotinto.com/news/releases/Automated-truck-expansion-Pilbara (visited on 10/07/2020).

[89] Mark Harris. *How Google's Autonomous Car Passed the First U.S. State Self-Driving Test - IEEE Spectrum*. en. Publication Title: IEEE Spectrum: Technology, Engineering, and Science News. 2014. URL: https://spectrum.ieee.org/transportation/advanced-cars/how-googles-autonomous-car-passed-the-first-us-state-selfdriving-test (visited on 10/07/2020).

[90] Arun Agarwal et al. "Driverless Car for Next Generation Commuters-Key Factors and Future Issues". In: *American Journal of Electrical and Electronic Engineering* 7.3 (2019), pp. 62–68.

[91] Saeed Asadi Bagloee et al. "Autonomous vehicles: challenges, opportunities, and future implications for transportation policies". In: *Journal of modern transportation* 24.4 (2016), pp. 284–303.

[92] EUR-Lex EUR-Lex. *EUR-Lex - 32019R2144 - EN - EUR-Lex*. en. 2014. URL: https://eur-lex.europa.eu/legal-content/en/ALL/?uri=CELEX%5C%3A32019R2144 (visited on 10/08/2020).

[93] Daniel J. Fagnant and Kara Kockelman. "Preparing a nation for autonomous vehicles: opportunities, barriers and policy recommendations". In: *Transportation Research Part A: Policy and Practice* 77 (2015), pp. 167–181.

[94] Hana Creger, Joel Espino, and Alvaro S. Sanchez. "Autonomous vehicle heaven or hell? Creating a transportation revolution that benefits all". In: (2019).

[95] Kum Fai Yuen et al. "Understanding public acceptance of autonomous vehicles using the Theory of Planned Behaviour". In: *International journal of environmental research and public health* 17.12 (2020), p. 4419.

[96] Ali Nasseri and Hacohen Shlomit. *2020 Autonomous Vehicle Technology Report*. en. Publication Title: Wevolver. 2020. URL: https://www.wevolver.com/article/2020.autonomous.vehicle.technology.report/ (visited on 10/12/2020).

[97] International SAE. *J3016B: Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles - SAE International*. 2018. URL: https://www.sae.org/standards/content/j3016%5C_201806/ (visited on 10/12/2020).

[98] Scott Drew Pendleton et al. "Perception, planning, control, and coordination for autonomous vehicles". In: *Machines* 5.1 (2017), p. 6.

[99] Roland Siegwart, Illah Reza Nourbakhsh, and Davide Scaramuzza. *Introduction to autonomous mobile robots*. MIT press, 2011.

[100] Hugh Durrant-Whyte. "A critical review of the state-of-the-art in autonomous land vehicle systems and technology". In: *Albuquerque (NM) andLivermore (CA), USA: SandiaNationalLaboratories* 41 (2001), p. 242.

[101] Luiz G. Galvao et al. "Pedestrian and vehicle detection in autonomous vehicle perception systems—A review". In: *Sensors* 21.21 (2021). Publisher: MDPI, p. 7267.

[102] Dickson Ben. *Tesla AI chief explains why self-driving cars don't need lidar*. en-US. Publication Title: VentureBeat. July 2021. URL: `https://venturebeat.com/2021/07/03/tesla-ai-chief-explains-why-self-driving-cars-dont-need-lidar/` (visited on 07/09/2021).

[103] Stéphanie Lefèvre, Dizan Vasquez, and Christian Laugier. "A survey on motion prediction and risk assessment for intelligent vehicles". In: *ROBOMECH journal* 1.1 (2014), pp. 1–14.

[104] Mohammad Shokrolah Shirazi and Brendan Tran Morris. "Looking at intersections: a survey of intersection monitoring, behavior and safety analysis of recent studies". In: *IEEE Transactions on Intelligent Transportation Systems* 18.1 (2016), pp. 4–24.

[105] Florin Leon and Marius Gavrilescu. "A review of tracking, prediction and decision making methods for autonomous driving". In: *arXiv preprint arXiv:1909.07707* (2019).

[106] Sayanan Sivaraman and Mohan Manubhai Trivedi. "Looking at vehicles on the road: A survey of vision-based vehicle detection, tracking, and behavior analysis". In: *IEEE transactions on intelligent transportation systems* 14.4 (2013), pp. 1773–1795.

[107] Yu Kong and Yun Fu. "Human action recognition and prediction: A survey". In: *arXiv preprint arXiv:1806.11230* (2018).

[108] Daniela Ridel et al. "A literature review on the prediction of pedestrian behavior in urban scenarios". In: *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2018, pp. 3105–3112.

[109] Andrey Rudenko et al. "Human motion trajectory prediction: A survey". In: *The International Journal of Robotics Research* 39.8 (2020). Publisher: Sage Publications Sage UK: London, England, pp. 895–935.

[110] Long Chen et al. "DenseLightNet: a light-weight vehicle detection network for autonomous driving". In: *IEEE Transactions on Industrial Electronics* 67.12 (2020), pp. 10600–10609.

[111] Neha Sharma, Chhavi Dhiman, and S. Indu. "Pedestrian Intention Prediction for Autonomous Vehicles: A Comprehensive Survey". In: *Neurocomputing* (2022). Publisher: Elsevier.

[112] Sarfraz Ahmed et al. "Pedestrian and cyclist detection and intent estimation for autonomous vehicles: A survey". In: *Applied Sciences* 9.11 (2019). Publisher: MDPI, p. 2335.

[113] Li Chen et al. “Survey of pedestrian action recognition techniques for autonomous driving”. In: *Tsinghua Science and Technology* 25.4 (2020). Publisher: TUP, pp. 458–470.

[114] Juan Alberto Antonio and Marcelo Romero. “Pedestrians’ Detection Methods in Video Images: A Literature Review”. In: *2018 International Conference on Computational Science and Computational Intelligence (CSCI)*. IEEE, 2018, pp. 354–360.

[115] N. K. Ragesh and Reghunadhan Rajesh. “Pedestrian detection in automotive safety: understanding state-of-the-art”. In: *IEEE Access* 7 (2019), pp. 47864–47890.

[116] B. S. Shobha and R. Deepu. “A review on video based vehicle detection, recognition and tracking”. In: *2018 3rd International Conference on Computational Systems and Information Technology for Sustainable Solutions (CSITSS)*. IEEE, 2018, pp. 183–186.

[117] Aymen Fadhil Abbas et al. “A comprehensive review of vehicle detection using computer vision.” In: *Telkomnika* 19.3 (2021).

[118] Patrick Dendorfer et al. “Motchallenge: A benchmark for single-camera multiple target tracking”. In: *International Journal of Computer Vision* 129 (2021). Publisher: Springer, pp. 845–881.

[119] Sarfraz Ahmed et al. “Visual and thermal data for pedestrian and cyclist detection”. In: *Towards Autonomous Robotic Systems: 20th Annual Conference, TAROS 2019, London, UK, July 3–5, 2019, Proceedings, Part II 20*. Springer, 2019, pp. 223–234.

[120] David Fernández-Llorca et al. “Two-Stream Networks for Lane-Change Prediction of Surrounding Vehicles”. In: *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2020, pp. 1–6.

[121] Rubén Izquierdo et al. “Vehicle Lane Change Prediction on Highways Using Efficient Environment Representation and Deep Learning”. In: *IEEE Access* 9 (2021), pp. 119454–119465.

[122] Yu Yao et al. “Coupling intent and action for pedestrian crossing behavior prediction”. In: *arXiv preprint arXiv:2105.04133* (2021).

[123] Lina Achaji et al. “Is attention to bounding boxes all you need for pedestrian action prediction?” In: *2022 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2022, pp. 895–902.

[124] Xingchen Zhang, Panagiotis Angeloudis, and Yiannis Demiris. “ST Crossing-Pose: A Spatial-Temporal Graph Convolutional Network for Skeleton-Based Pedestrian Crossing Intention Prediction”. In: *IEEE Transactions on Intelligent Transportation Systems* (2022). Publisher: IEEE.

[125] Zhuoran Zeng. “High Efficiency Pedestrian Crossing Prediction”. In: *arXiv preprint arXiv:2204.01862* (2022).

[126] Dongfang Yang et al. “Predicting pedestrian crossing intention with feature fusion and spatio-temporal attention”. In: *IEEE Transactions on Intelligent Vehicles* 7.2 (2022). Publisher: IEEE, pp. 221–230.

[127] Abhilash Y. Naik et al. “Scene Spatio-Temporal Graph Convolutional Network for Pedestrian Intention Estimation”. In: *2022 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2022, pp. 874–881.

[128] Amir Rasouli et al. “Pie: A large-scale dataset and models for pedestrian intention estimation and trajectory prediction”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019, pp. 6262–6271.

[129] Dijana Vitas, Martina Tomic, and Matko Burul. “Traffic Light Detection in Autonomous Driving Systems”. In: *IEEE Consumer Electronics Magazine* 9.4 (2020), pp. 90–96.

[130] Iuliia Kotseruba, Amir Rasouli, and John K. Tsotsos. “Do they want to cross? understanding pedestrian intention for behavior prediction”. In: *2020 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2020, pp. 1688–1693.

[131] Smail Ait Bouhsain, Saeed Saadatnejad, and Alexandre Alahi. “Pedestrian intention prediction: A multi-task perspective”. In: *arXiv preprint arXiv:2010.10270* (2020).

[132] Amir Rasouli, Iuliia Kotseruba, and John K. Tsotsos. “Pedestrian action anticipation using contextual feature fusion in stacked rnns”. In: *arXiv preprint arXiv:2005.06582* (2020).

[133] Francesco Piccoli et al. “Fussi-net: Fusion of spatio-temporal skeletons for intention prediction network”. In: *2020 54th Asilomar Conference on Signals, Systems, and Computers*. IEEE, 2020, pp. 68–72.

[134] Soulayma Gazzeh and Ali Douik. “Deep learning for pedestrian behavior understanding”. In: *2022 6th International Conference on Advanced Technologies for Signal and Image Processing (ATSIP)*. IEEE, 2022, pp. 1–5.

[135] Hao Xue, Du Q. Huynh, and Mark Reynolds. “SS-LSTM: A hierarchical LSTM model for pedestrian trajectory prediction”. In: *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, 2018, pp. 1186–1194.

[136] Shengzhe Dai, Li Li, and Zhiheng Li. “Modeling vehicle interactions via modified LSTM models for trajectory prediction”. In: *IEEE Access* 7 (2019), pp. 38287–38296.

[137] Long Xin et al. “Intention-aware long horizon trajectory prediction of surrounding vehicles using dual lstm networks”. In: *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2018, pp. 1441–1446.

[138] Xin Li, Xiaowen Ying, and Mooi Choo Chuah. “Grip++: Enhanced graph-based interaction-aware trajectory prediction for autonomous driving”. In: *arXiv preprint arXiv:1907.07792* (2019).

[139] Nachiket Deo and Mohan M. Trivedi. “Convolutional social pooling for vehicle trajectory prediction”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*. 2018, pp. 1468–1476.

[140] Nachiket Deo and Mohan M. Trivedi. “Multi-modal trajectory prediction of surrounding vehicles with maneuver based lstms”. In: *2018 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2018, pp. 1179–1184.

[141] Kaouther Messaoud et al. “Non-local social pooling for vehicle trajectory prediction”. In: *2019 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2019, pp. 975–980.

[142] Florent Altché and Arnaud de La Fortelle. “An LSTM network for highway trajectory prediction”. In: *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2017, pp. 353–359.

[143] ByeoungDo Kim et al. “Probabilistic vehicle trajectory prediction over occupancy grid map via recurrent neural network”. In: *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2017, pp. 399–404.

[144] Xin Li, Xiaowen Ying, and Mooi Choo Chuah. “Grip: Graph-based interaction-aware trajectory prediction”. In: *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*. IEEE, 2019, pp. 3960–3966.

[145] Pu Zhang et al. “Sr-lstm: State refinement for lstm towards pedestrian trajectory prediction”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 12085–12094.

[146] Karttikeya Mangalam et al. “It is not the journey but the destination: Endpoint conditioned trajectory prediction”. In: *European conference on computer vision*. Springer, 2020, pp. 759–776.

[147] Abduallah Mohamed et al. “Social-stgcnn: A social spatio-temporal graph convolutional neural network for human trajectory prediction”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 14424–14432.

[148] Jianhua Sun, Qinhong Jiang, and Cewu Lu. “Recursive social behavior graph for trajectory prediction”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 660–669.

[149] Yanliang Zhu et al. “Starnet: Pedestrian trajectory prediction using deep neural network in star topology”. In: *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 8075–8080.

[150] Anirudh Vemula, Katharina Muelling, and Jean Oh. “Social attention: Modeling attention in human crowds”. In: *2018 IEEE international Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 4601–4607.

[151] Yanyu Xu, Zhixin Piao, and Shenghua Gao. “Encoding crowd interaction with deep neural network for pedestrian trajectory prediction”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 5275–5284.

[152] Amir Sadeghian et al. “Sophie: An attentive gan for predicting paths compliant to social and physical constraints”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019, pp. 1349–1358.

[153] Agrim Gupta et al. “Social gan: Socially acceptable trajectories with generative adversarial networks”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 2255–2264.

[154] Abenezer Girma et al. “Deep learning with attention mechanism for predicting driver intention at intersection”. In: *2020 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2020, pp. 1183–1188.

[155] Holger Berndt and Klaus Dietmayer. “Driver intention inference with vehicle onboard sensors”. In: *2009 IEEE international conference on vehicular electronics and safety (ICVES)*. IEEE, 2009, pp. 102–107.

[156] Yang Xing et al. *Recognizing driver braking intention with vehicle data using unsupervised learning methods*. Tech. rep. SAE Technical Paper, 2017.

[157] Vazquez Raimundo and Mason Favio. “Driver intention prediction at roundabouts”. In: *2021 XIX Workshop on Information Processing and Control (RPIC)*. IEEE, 2021, pp. 1–5.

[158] Wei Zhou et al. “Developing and testing robust autonomy: The university of sydney campus data set”. In: *IEEE Intelligent Transportation Systems Magazine* 12.4 (2020), pp. 23–40.

[159] Generation SIMulation. “US Highway 101 Dataset”. In: (2007).

[160] Alex Zyner, Stewart Worrall, and Eduardo M. Nebot. “ACFR five roundabouts dataset: Naturalistic driving at unsignalized intersections”. In: *IEEE Intelligent Transportation Systems Magazine* 11.4 (2019), pp. 8–18.

[161] Christoph Hermes et al. “Long-term vehicle motion prediction”. In: *2009 IEEE intelligent vehicles symposium*. IEEE, 2009, pp. 652–657.

[162] Young-Chul Lim et al. “Improvement of stereo vision-based position and velocity estimation and tracking using a stripe-based disparity estimation and inverse perspective map-based extended Kalman filter”. In: *Optics and Lasers in Engineering* 48.9 (2010), pp. 859–868.

[163] Dietmar Kasper et al. “Object-oriented Bayesian networks for detection of lane change maneuvers”. In: *IEEE Intelligent Transportation Systems Magazine* 4.3 (2012), pp. 19–31.

[164] Puneet Kumar et al. “Learning-based approach for online lane change intention prediction”. In: *2013 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2013, pp. 797–802.

[165] Seungje Yoon and Dongsuk Kum. “The multilayer perceptron approach to lateral motion prediction of surrounding vehicles for autonomous vehicles”. In: *2016 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2016, pp. 1307–1312.

[166] Aida Khosroshahi, Eshed Ohn-Bar, and Mohan Manubhai Trivedi. “Surround vehicles trajectory analysis with recurrent neural networks”. In: *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2016, pp. 2267–2272.

[167] Jacob Velling Dueholm et al. “Trajectories and maneuvers of surrounding vehicles with panoramic camera arrays”. In: *IEEE Transactions on Intelligent Vehicles* 1.2 (2016), pp. 203–214.

[168] Donghan Lee et al. “Convolution neural network-based lane change intention prediction of surrounding vehicles for acc”. In: *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2017, pp. 1–6.

[169] Seong Hyeon Park et al. “Sequence-to-sequence prediction of vehicle trajectory via LSTM encoder-decoder architecture”. In: *2018 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2018, pp. 1672–1678.

[170] Nachiket Deo, Akshay Rangesh, and Mohan M. Trivedi. “How would surround vehicles move? a unified framework for maneuver classification and motion prediction”. In: *IEEE Transactions on Intelligent Vehicles* 3.2 (2018), pp. 129–140.

[171] Shuang Su et al. “Learning vehicle surrounding-aware lane-changing behavior from observed trajectories”. In: *2018 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2018, pp. 1412–1417.

[172] Tianyang Zhao et al. “Multi-agent tensor fusion for contextual trajectory prediction”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 12126–12134.

[173] Abdelmoudjib Benterki et al. “Prediction of surrounding vehicles lane change intention using machine learning”. In: *2019 10th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS)*. Vol. 2. IEEE, 2019, pp. 839–843.

[174] Abdelmoudjib Benterki et al. “Artificial intelligence for vehicle behavior anticipation: Hybrid approach based on maneuver classification and trajectory prediction”. In: *IEEE Access* 8 (2020), pp. 56992–57002.

[175] Hailun Zhang and Rui Fu. “A Hybrid Approach for Turning Intention Prediction Based on Time Series Forecasting and Deep Learning”. In: *Sensors* 20.17 (2020), p. 4887.

[176] He Huang et al. “Spatial-temporal ConvLSTM for vehicle driving intention prediction”. In: *Tsinghua Science and Technology* 27.3 (2021), pp. 599–609.

[177] Ting Zhang et al. “Vehicle motion prediction at intersections based on the turning intention and prior trajectories model”. In: *IEEE/CAA Journal of Automatica Sinica* 8.10 (2021), pp. 1657–1666.

[178] Jyun-Hong He et al. “Vehicle Turning Intention Prediction Based on Data-Driven Method with Roadside Radar and Vision Sensor”. In: *2021 IEEE International Conference on Consumer Electronics-Taiwan (ICCE-TW)*. IEEE, 2021, pp. 1–2.

[179] Zhongkai Luan et al. “A comprehensive lateral motion prediction method of surrounding vehicles integrating driver intention prediction and vehicle behavior recognition”. In: *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering* (2022), p. 09544070221078636.

[180] Kunpeng Zhang et al. “AI-TP: Attention-based Interaction-aware Trajectory Prediction for Autonomous Driving”. In: *IEEE Transactions on Intelligent Vehicles* (2022). Publisher: IEEE.

[181] Alexandre Alahi et al. “Social lstm: Human trajectory prediction in crowded spaces”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 961–971.

[182] Alex Kuefler et al. “Imitating driver behavior with generative adversarial networks”. In: *2017 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2017, pp. 204–211.

[183] Zhijie Fang and Antonio M. López. "Is the pedestrian going to cross? answering by 2d pose estimation". In: *2018 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2018, pp. 1271–1276.

[184] Nicolas Schneider and Dariu M. Gavrila. "Pedestrian path prediction with recursive bayesian filters: A comparative study". In: *German Conference on Pattern Recognition*. Springer, 2013, pp. 174–183.

[185] Christoph G. Keller and Dariu M. Gavrila. "Will the pedestrian cross? a study on pedestrian path prediction". In: *IEEE Transactions on Intelligent Transportation Systems* 15.2 (2013), pp. 494–506.

[186] Vasiliy Karasev et al. "Intent-aware long-term prediction of pedestrian motion". In: *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 2543–2549.

[187] Eike Rehder et al. "Pedestrian prediction by planning using deep neural networks". In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 1–5.

[188] Xiaowei Zhang et al. "Too far to see? Not really!—Pedestrian detection with scale-aware localization policy". In: *IEEE transactions on image processing* 27.8 (2018), pp. 3703–3715.

[189] Apratim Bhattacharyya, Mario Fritz, and Bernt Schiele. "Long-term on-board prediction of people in traffic scenes under uncertainty". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 4194–4202.

[190] Fabian Flohr Flohr et al. "Context-Based Path Prediction for Targets with Switching Dynamics". In: (2018).

[191] Irtiza Hasan et al. "Mx-lstm: mixing tracklets and vislets to jointly forecast trajectories and head poses". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 6067–6076.

[192] Huynh Manh and Gita Alaghband. "Scene-lstm: A model for human trajectory prediction". In: *arXiv preprint arXiv:1808.04018* (2018).

[193] Hao Xue, Du Q. Huynh, and Mark Reynolds. "A location-velocity-temporal attention LSTM model for pedestrian trajectory prediction". In: *IEEE Access* 8 (2020). Publisher: IEEE, pp. 44576–44589.

[194] Ruijie Quan et al. "Holistic LSTM for pedestrian trajectory prediction". In: *IEEE transactions on image processing* 30 (2021). Publisher: IEEE, pp. 3229–3239.

[195] Yu Yao et al. "Bitrap: Bi-directional pedestrian trajectory prediction with multi-modal goal estimation". In: *IEEE Robotics and Automation Letters* 6.2 (2021). Publisher: IEEE, pp. 1463–1470.

[196] Chuhua Wang et al. "Stepwise goal-driven networks for trajectory prediction". In: *IEEE Robotics and Automation Letters* 7.2 (2022). Publisher: IEEE, pp. 2716–2723.

[197] Juan Yang et al. "PTPGC: Pedestrian trajectory prediction by graph attention network with ConvLSTM". In: *Robotics and Autonomous Systems* 148 (2022). Publisher: Elsevier, p. 103931.

[198] Zhijie Fang, David Vázquez, and Antonio M. López. "On-board detection of pedestrian intentions". In: *Sensors* 17.10 (2017). Publisher: MDPI, p. 2193.

[199] Khairi Abdulrahim and Rosalina Abdul Salam. "Traffic surveillance: A review of vision based vehicle detection, recognition and tracking". In: *International journal of applied engineering research* 11.1 (2016), pp. 713–726.

[200] Biao Yang et al. "Crossing or not? Context-based recognition of pedestrian crossing intention in the urban environment". In: *IEEE Transactions on Intelligent Transportation Systems* 23.6 (2021). Publisher: IEEE, pp. 5338–5349.

[201] Jing Lian et al. "Early intention prediction of pedestrians using contextual attention-based LSTM". In: *Multimedia Tools and Applications* (2022). Publisher: Springer, pp. 1–17.

[202] Junyoung Chung et al. "Empirical evaluation of gated recurrent neural networks on sequence modeling". In: *arXiv preprint arXiv:1412.3555* (2014).

[203] Sarfraz Ahmed et al. "Multi-scale pedestrian intent prediction using 3D joint information as spatio-temporal representation". In: *Expert Systems with Applications* 225 (2023). Publisher: Elsevier, p. 120077.

[204] Sarah Bonnin et al. "Pedestrian crossing prediction using multiple context-based models". In: *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2014, pp. 378–385.

[205] Satyajit Neogi et al. "Context based pedestrian intention prediction using factored latent dynamic conditional random fields". In: *2017 IEEE Symposium Series on Computational Intelligence (SSCI)*. IEEE, 2017, pp. 1–8.

[206] Raul Quintero Minguez et al. "Pedestrian path, pose, and intention prediction through gaussian process dynamical models and pedestrian activity recognition". In: *IEEE Transactions on Intelligent Transportation Systems* 20.5 (2018), pp. 1803–1814.

[207] Bingbin Liu et al. "Spatiotemporal relationship reasoning for pedestrian intent prediction". In: *IEEE Robotics and Automation Letters* 5.2 (2020). Publisher: IEEE, pp. 3485–3492.

[208] Karam M. Abughalieh and Shadi G. Alawneh. "Predicting pedestrian intention to cross the road". In: *IEEE Access* 8 (2020). Publisher: IEEE, pp. 72558–72569.

[209] Haziq Razali, Taylor Mordan, and Alexandre Alahi. "Pedestrian intention prediction: A convolutional bottom-up multi-task approach". In: *Transportation research part C: emerging technologies* 130 (2021). Publisher: Elsevier, p. 103259.

[210] Shile Zhang et al. "Pedestrian crossing intention prediction at red-light using pose estimation". In: *IEEE Transactions on Intelligent Transportation Systems* 23.3 (2021). Publisher: IEEE, pp. 2331–2339.

[211] Tina Chen, Renran Tian, and Zhengming Ding. "Visual reasoning using graph convolutional networks for predicting pedestrian crossing intention". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 3103–3109.

[212] Pablo Rodrigo Gantier Cadena et al. "Pedestrian Graph+: A Fast Pedestrian Crossing Prediction Model Based on Graph Convolutional Networks". In: *IEEE Transactions on Intelligent Transportation Systems* (2022). Publisher: IEEE.

[213] Jun Ma and Wenhui Rong. "Pedestrian Crossing Intention Prediction Method Based on Multi-Feature Fusion". In: *World Electric Vehicle Journal* 13.8 (2022). Publisher: MDPI, p. 158.

[214] Yuexin Ma et al. "Trafficpredict: Trajectory prediction for heterogeneous traffic-agents". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 33. 2019, pp. 6120–6127.

[215] Rohan Chandra et al. *Deeptagent: Realtime tracking of dense traffic agents using heterogeneous interaction*. Technical Report, 2018.[Online]. Available: http://gamma. cs. unc. edu/HTI ..., 2019.

[216] Rohan Chandra et al. "RoadTrack: Realtime Tracking of Road Agents in Dense and Heterogeneous Environments". In: *arXiv* (2019), arXiv–1906.

[217] Jiachen Li et al. "Evolvegraph: Multi-agent trajectory prediction with dynamic relational reasoning". In: *Proceedings of the Neural Information Processing Systems (NeurIPS)* (2020).

[218] Xiaoyu Mo et al. "Multi-agent trajectory prediction with heterogeneous edge-enhanced graph attention network". In: *IEEE Transactions on Intelligent Transportation Systems* (2022). Publisher: IEEE.

[219] Yaran Chen et al. "Multi-task learning for dangerous object detection in autonomous driving". In: *Information Sciences* 432 (2018), pp. 559–571.

[220] Yanfen Li et al. "A deep learning-based hybrid framework for object detection and recognition in autonomous driving". In: *IEEE Access* 8 (2020), pp. 194228–194239.

[221] Ashesh Jain et al. "Car that knows before you do: Anticipating maneuvers via learning temporal driving models". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2015, pp. 3182–3190. URL: http://openaccess.thecvf.com/content_iccv_2015/html/Jain_Car_That_Knows_ICCV_2015_paper.html (visited on 01/05/2024).

[222] Nicolas Pugeault and Richard Bowden. "How much of driving is preattentive?" In: *IEEE Transactions on Vehicular Technology* 64.12 (2015), pp. 5424–5438. URL: https://ieeexplore.ieee.org/abstract/document/7293673/?casa_token=BN5hMy_adXcAAAAA:Zj7NZYht3DnwQ4ofG8YjMH29pQCHSYHkCpcl1rzsQfsz1pZLt1Oh6musFSICo2YdmnBKWe6R (visited on 01/05/2024).

[223] Andrea Palazzi et al. "Predicting the Driver's Focus of Attention: the DR (eye) VE Project". In: *IEEE transactions on pattern analysis and machine intelligence* 41.7 (2018), pp. 1720–1733. URL: https://ieeexplore.ieee.org/abstract/document/8375682/?casa_token=hFROQFkUwzYAAAAA:TXXqr7Xrt7IIMO7bbKajYAqHJgo2YOmjD34kZJyV37qAYaiW0whOlYB1Fd4sSFzN5IsS50Gv (visited on 01/05/2024).

[224] Amir Rasouli, Iuliia Kotseruba, and John K. Tsotsos. "Are they going to cross? a benchmark dataset and baseline for pedestrian crosswalk behavior". In: *Proceedings of the IEEE International Conference on Computer Vision Workshops*. 2017, pp. 206–213.

[225] Vasili Ramanishka et al. "Toward driving scene understanding: A dataset for learning driver behavior and causal reasoning". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 7699–7707. URL: http://openaccess.thecvf.com/content_cvpr_2018/html/Ramanishka_Toward_Driving_Scene_CVPR_2018_paper.html (visited on 01/12/2024).

[226] Parth Bhavsar et al. "Risk analysis of autonomous vehicles in mixed traffic streams". In: *Transportation Research Record* 2625.1 (2017). Publisher: SAGE Publications Sage CA: Los Angeles, CA, pp. 51–61.

[227] Liudong Xing and Suprasad V. Amari. "Fault tree analysis". In: *Handbook of performability engineering* (2008). Publisher: Springer, pp. 595–620.

[228] Enno Ruijters and Mariëlle Stoelinga. "Fault tree analysis: A survey of the state-of-the-art in modeling, analysis and tools". In: *Computer science review* 15 (2015). Publisher: Elsevier, pp. 29–62.

[229] Jiaxin Liu et al. "PNNUAD: Perception Neural Networks Uncertainty Aware Decision-Making for Autonomous Vehicle". In: *IEEE Transactions on Intelligent Transportation Systems* 23.12 (2022). Publisher: IEEE, pp. 24355–24368.

[230] Olga Russakovsky et al. "Imagenet large scale visual recognition challenge". In: *International journal of computer vision* 115 (2015). Publisher: Springer, pp. 211–252.

[231] Tsung-Yi Lin et al. "Microsoft coco: Common objects in context". In: *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*. Springer, 2014, pp. 740–755.

[232] Jason Zhang. *Deep Understanding Tesla FSD Part 1: HydraNet*. en. Dec. 2021. URL: https://saneryee-studio.medium.com/deep-understanding-tesla-fsd-part-1-hydranet-1b46106d57 (visited on 05/25/2023).

[233] Jean-Nicola Russo et al. "Risk level assessment for rear-end collision with Bayesian network". In: *IFAC-PapersOnLine* 50.1 (2017). Publisher: Elsevier, pp. 12514–12519. URL: https://www.sciencedirect.com/science/article/pii/S2405896317327040 (visited on 01/11/2024).

[234] Roei Herzig et al. "Spatio-temporal action graph networks". In: *Proceedings of the IEEE/CVF international conference on computer vision workshops*. 2019. URL: http://openaccess.thecvf.com/content_ICCVW_2019/html/ADW/Herzig_Spatio-Temporal_Action_Graph_Networks_ICCVW_2019_paper.html (visited on 01/11/2024).

[235] Hirokatsu Kataoka et al. "Drive video analysis for the detection of traffic near-miss incidents". In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 3421–3428. URL: https://ieeexplore.ieee.org/abstract/document/8460812/?casa_token=7rVkZc63YXIAAAAA:U5oTXNo0QUjb1smwqiF67-pmWlOJtp2-GIwpF8weSFBY-QTK28N6p9eD5dw6OO54NMF_4rX9 (visited on 01/08/2024).

[236] Hoon Kim et al. "Crash to not crash: Learn to identify dangerous vehicles using a simulator". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 33. 2019, pp. 978–985. URL: https://ojs.aaai.org/index.php/AAAI/article/view/3887 (visited on 01/08/2024).

[237] Yu Yao et al. "Unsupervised traffic accident detection in first-person videos". In: *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 273–280. URL: https://ieeexplore.ieee.org/abstract/document/8967556/ (visited on 01/08/2024).

[238] Ekim Yurtsever et al. "Risky action recognition in lane change video clips using deep spatiotemporal networks with segmentation mask transfer". In: *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*. IEEE, 2019, pp. 3100–3107. URL: https://ieeexplore.ieee.org/abstract/document/8917362/ (visited on 01/11/2024).

[239] Fu-Hsiang Chan et al. "Anticipating Accidents in Dashcam Videos". In: *Computer Vision – ACCV 2016*. Ed. by Shang-Hong Lai et al. Vol. 10114. Cham: Springer International Publishing, 2017, pp. 136–153. DOI: 10.1007/978-3-319-54190-7_9. URL: http://link.springer.com/10.1007/978-3-319-54190-7_9 (visited on 01/08/2024).

[240] Wentao Bao, Qi Yu, and Yu Kong. "Uncertainty-based Traffic Accident Anticipation with Spatio-Temporal Relational Learning". In: *ACM Multimedia Conference*. May 2020.

[241] Arnav Vaibhav Malawade et al. "Spatiotemporal scene-graph embedding for autonomous vehicle collision prediction". In: *IEEE Internet of Things Journal* 9.12 (2022). Publisher: IEEE, pp. 9379–9388. URL: https://ieeexplore.ieee.org/abstract/document/9672160/ (visited on 01/11/2024).

[242] Yu Yao et al. "DoTA: unsupervised detection of traffic anomaly in driving videos". In: *IEEE transactions on pattern analysis and machine intelligence* (2022). Publisher: IEEE.

[243] Tackgeun You and Bohyung Han. "Traffic Accident Benchmark for Causality Recognition". en. In: *Computer Vision – ECCV 2020*. Ed. by Andrea Vedaldi et al. Vol. 12352. Cham: Springer International Publishing, 2020, pp. 540–556. DOI: 10.1007/978-3-030-58571-6_32. URL: https://link.springer.com/10.1007/978-3-030-58571-6_32 (visited on 01/08/2024).

[244] Kuan-Hui Lee et al. "An attention-based recurrent convolutional network for vehicle taillight recognition". In: *2019 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2019, pp. 2365–2370.

[245] Santhosh Kelathodi Kumaran, Debi Prosad Dogra, and Partha Pratim Roy. "Anomaly detection in road traffic using visual surveillance: A survey". In: *arXiv preprint arXiv:1901.08292* (2019).

[246] Jian-Gang Wang et al. "Real-time vehicle signal lights recognition with HDR camera". In: *2016 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*. IEEE, 2016, pp. 355–358.

[247] Yi Li, Zi-xing Cai, and Jin Tang. "Recognition algorithm for turn light of front vehicle". In: *Journal of Central South University* 19.2 (2012). Publisher: Springer, pp. 522–526.

[248] Björn Fröhlich, Markus Enzweiler, and Uwe Franke. "Will this car change the lane?-turn signal recognition in the frequency domain". In: *2014 IEEE Intelligent Vehicles Symposium Proceedings*. IEEE, 2014, pp. 37–42.

[249] Li-Chih Chen et al. "Robust rear light status recognition using symmetrical surfs". In: *2015 IEEE 18th International Conference on Intelligent Transportation Systems*. IEEE, 2015, pp. 2053–2058.

[250] Ravi Kumar Satzoda and Mohan Manubhai Trivedi. "Looking at vehicles in the night: Detection and dynamics of rear lights". In: *IEEE transactions on intelligent transportation systems* 20.12 (2016). Publisher: IEEE, pp. 4297–4307.

[251] Guangyu Zhong et al. "Learning to tell brake lights with convolutional features". In: *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2016, pp. 1558–1563.

[252] Zhenzhou Wang et al. "Research on vehicle taillight detection and semantic recognition based on Internet of vehicle". In: *2018 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC)*. IEEE, 2018, pp. 147–1473.

[253] Zhenzhou Wang et al. "Performance evaluation of region-based convolutional neural networks toward improved vehicle taillight detection". In: *Applied Sciences* 9.18 (2019). Publisher: MDPI, p. 3753.

[254] Dario Nava, Giulio Panzani, and Sergio M. Savaresi. "A collision warning oriented brake lights detection and classification algorithm based on a mono camera sensor". In: *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*. IEEE, 2019, pp. 319–324.

[255] Long Chen et al. "Turn signal detection during nighttime by CNN detector and perceptual hashing tracking". In: *IEEE Transactions on Intelligent Transportation Systems* 18.12 (2017). Publisher: IEEE, pp. 3303–3314.

[256] Qiaohong Li et al. "A highly efficient vehicle taillight detection approach based on deep learning". In: *IEEE transactions on intelligent transportation systems* 22.7 (2020). Publisher: IEEE, pp. 4716–4726.

[257] Ming Liu et al. "Real-Time Vehicle Taillight Recognition Based on Siamese Recurrent Neural Network". In: *Journal of Physics: Conference Series*. Vol. 1673. Issue: 1. IOP Publishing, 2020, p. 012056.

[258] Le Chang and Chongyang Zhang. "Vehicle Taillight Detection Based on Semantic Information Fusion". In: *International Conference on Neural Information Processing*. Springer, 2021, pp. 528–536.

[259] Gobind Kumar, Medipelly Rampavan, and Earnest Paul Ijjina. "Deep Learning based Brake Light Detection for Two Wheelers". In: *2021 12th International Conference on Computing Communication and Networking Technologies (ICCCNT)*. IEEE, 2021, pp. 1–4.

[260] Hyung-Joon Jeon et al. "A Deep Learning Framework for Robust and Real-Time Taillight Detection Under Various Road Conditions". In: *IEEE Transactions on Intelligent Transportation Systems* (2022). Publisher: IEEE.

[261] Wei Liu et al. "Vision-based method for forward vehicle brake lights recognition". In: *International Journal of Signal Processing, Image Processing and Pattern Recognition* 8.6 (2015), pp. 167–180.

[262] Jesse Pirhonen et al. "Brake light detection algorithm for predictive braking". In: *Applied Sciences* 12.6 (2022). Publisher: MDPI, p. 2804.

[263] Kotcharat Kitchat et al. "Taillight Signal Recognition via Sequential Learning". en. In: *Proceedings of the 52nd International Conference on Parallel Processing Workshops*. Salt Lake City UT USA: ACM, Aug. 2023, pp. 1–7. DOI: 10.1145/3605731.3605872. URL: https://dl.acm.org/doi/10.1145/3605731.3605872 (visited on 07/15/2024).

[264] Aaron van den Oord et al. *WaveNet: A Generative Model for Raw Audio*. arXiv:1609.03499 [cs]. Sept. 2016. URL: http://arxiv.org/abs/1609.03499 (visited on 07/15/2024).

[265] GOVUK GOVUK. *The Highway Code - Signals to other road users - Guidance - GOV.UK*. en. 2023. URL: https://www.gov.uk/guidance/the-highway-code/signals-to-other-road-users (visited on 12/15/2023).

[266] Creative Commons Creative Commons. *CC BY-NC-SA 4.0 Legal Code — Attribution-NonCommercial-ShareAlike 4.0 International — Creative Commons*. URL: https://creativecommons.org/licenses/by-nc-sa/4.0/legalcode.en (visited on 02/06/2024).

[267] Yonghye Kwon. *developer0hye/Yolo_Label*. original-date: 2018-10-29T04:24:17Z. Jan. 2024. URL: https://github.com/developer0hye/Yolo_Label (visited on 01/16/2024).

[268] Shuwei Shao et al. "IEBins: Iterative Elastic Bins for Monocular Depth Estimation". In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2023.

[269] Dong Wu et al. "Yolop: You only look once for panoptic driving perception". In: *Machine Intelligence Research* (2022). Publisher: Springer, pp. 1–13.

[270] Alexey Dosovitskiy et al. *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale*. arXiv:2010.11929 [cs]. June 2021. URL: http://arxiv.org/abs/2010.11929 (visited on 07/16/2024).

[271] Ze Liu et al. "Swin transformer: Hierarchical vision transformer using shifted windows". In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2021, pp. 10012–10022.

[272] Kaiming He et al. "Deep residual learning for image recognition". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.

[273] Mingxing Tan and Quoc Le. "Efficientnetv2: Smaller models and faster training". In: *International conference on machine learning*. PMLR. 2021, pp. 10096–10106.

[274] Zhuang Liu et al. "A convnet for the 2020s". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2022, pp. 11976–11986.

[275] Ilija Radosavovic et al. "Designing network design spaces". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 10428–10436.

[276] Haşim Sak, Andrew Senior, and Françoise Beaufays. "Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition". In: *arXiv preprint arXiv:1402.1128* (2014).

[277] *ReLU, Leaky ReLU, Sigmoid, Tanh and Softmax - MLK - Machine Learning Knowledge*. Accessed: 2025-06-02. URL: https://machinelearningknowledge.ai/pytorch-activation-functions-relu-leaky-relu-sigmoid-tanh-and-softmax/.

[278] Dosovitskiy Alexey. "An image is worth 16x16 words: Transformers for image recognition at scale". In: *arXiv preprint arXiv: 2010.11929* (2020).

[279] A Vaswani. "Attention is all you need". In: *Advances in Neural Information Processing Systems* (2017).

[280] Srinath NS, Maria Anu Vensuslaus, Joshua Thomas John Victor, et al. "Lane Change Prediction of Surrounding Vehicles using Video Vision Transformers". In: *International Journal of Computing and Digital Systems* 16.1 (2024), pp. 1–15.

[281] Ekin D. Cubuk et al. "Randaugment: Practical automated data augmentation with a reduced search space". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*. 2020, pp. 702–703. URL: http://openaccess.thecvf.com/content_CVPRW_2020/html/w40/Cubuk_Randaugment_Practical_Automated_Data_Augmentation_With_a_Reduced_Search_Space_CVPRW_2020_paper.html (visited on 01/25/2024).

[282] Ekin D. Cubuk et al. "Autoaugment: Learning augmentation strategies from data". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019, pp. 113–123. URL: http://openaccess.thecvf.com/content_CVPR_2019/html/Cubuk_AutoAugment_Learning_Augmentation_Strategies_From_Data_CVPR_2019_paper.html (visited on 01/25/2024).

[283] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. *UCF101: A Dataset of 101 Human Actions Classes From Videos in The Wild*. arXiv:1212.0402 [cs]. Dec. 2012. URL: http://arxiv.org/abs/1212.0402 (visited on 03/15/2024).

[284] Zhijie Fang and Antonio M López. "Intention recognition of pedestrians and cyclists by 2d pose estimation". In: *IEEE Transactions on Intelligent Transportation Systems* 21.11 (2019), pp. 4773–4783.

[285] Dong Cao and Yunbin Fu. "Using graph convolutional networks skeleton-based pedestrian intention estimation models for trajectory prediction". In: *Journal of Physics: Conference Series*. Vol. 1621. IOP Publishing. 2020, p. 012047.

[286] Amir Rasouli, Mohsen Rohani, and Jun Luo. "Bifold and semantic reasoning for pedestrian behavior prediction". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 15600–15610.

[287] Javier Lorenzo, Ignacio Parra, and MA Sotelo. "Intformer: Predicting pedestrian intention with the aid of the transformer architecture". In: *arXiv preprint arXiv:2105.08647* (2021).

[288] Ankur Singh and Upendra Suddamalla. "Multi-input fusion for practical pedestrian intention prediction". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 2304–2311.

[289] Tiffany Yau et al. "Graph-sim: A graph-based spatiotemporal interaction modelling for pedestrian action prediction". In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2021, pp. 8580–8586.

[290] Jibran Ali Abbasi et al. "Watchped: Pedestrian crossing intention prediction using embedded sensors of smartwatch". In: *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2023, pp. 9574–9581.

[291] Shengzhe Zhao et al. "Action-vit: Pedestrian intent prediction in traffic scenes". In: *IEEE Signal Processing Letters* 29 (2021), pp. 324–328.

[292] Rongrong Ni et al. "Pedestrians crossing intention anticipation based on dual-channel action recognition and hierarchical environmental context". In: *IET Intelligent Transport Systems* 17.2 (2023), pp. 255–269.

[293] Je-Seok Ham et al. "Cipf: Crossing intention prediction network based on feature fusion modules for improving pedestrian safety". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 3666–3675.

[294] Zhengming Zhang, Renran Tian, and Zhengming Ding. "Trep: Transformer-based evidential prediction for pedestrian intention with uncertainty". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 37. 2023, pp. 3534–3542.

[295] Mohsen Azarmi et al. "Local and global contextual features fusion for pedestrian intention prediction". In: *International Conference on Artificial Intelligence and Smart Vehicles*. Springer. 2023, pp. 1–13.

[296] Chi Zhang et al. "Cross or wait? Predicting pedestrian interaction outcomes at unsignalized crossings". In: *2023 IEEE Intelligent Vehicles Symposium (IV)*. IEEE. 2023, pp. 1–8.

[297] Dongxu Guo, Taylor Mordan, and Alexandre Alahi. "Pedestrian stop and go forecasting with hybrid feature fusion". In: *2022 International Conference on Robotics and Automation (ICRA)*. IEEE. 2022, pp. 940–947.

[298] Yuchen Zhou et al. "Pit: Progressive interaction transformer for pedestrian crossing intention prediction". In: *IEEE Transactions on Intelligent Transportation Systems* (2023).

[299] Jian Zheng, Koji Suzuki, and Motohiro Fujita. "Predicting driver's lane-changing decisions using a neural network model". In: *Simulation Modelling Practice and Theory* 42 (2014), pp. 73–83.

[300] Ting Xu et al. “The hybrid model for lane-changing detection at freeway off-ramps using naturalistic driving trajectories”. In: *IEEE Access* 7 (2019), pp. 103716–103726.

[301] Rubén Izquierdo et al. “Experimental validation of lane-change intention prediction methodologies based on CNN and LSTM”. In: *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*. IEEE. 2019, pp. 3657–3662.

[302] Christina Ng et al. “Development of a binary logistic lane change model and its validation using empirical freeway data”. In: *Transportmetrica B: Transport Dynamics* 8.1 (2020), pp. 49–71.

[303] Abenezer Girma et al. “Deep learning with attention mechanism for predicting driver intention at intersection”. In: *2020 IEEE intelligent vehicles symposium (IV)*. IEEE. 2020, pp. 1183–1188.

[304] Vishal Mahajan, Christos Katrakazas, and Constantinos Antoniou. “Prediction of lane-changing maneuvers with automatic labeling and deep learning”. In: *Transportation research record* 2674.7 (2020), pp. 336–347.

[305] Weida Wang et al. “An intelligent lane-changing behavior prediction and decision-making strategy for an autonomous vehicle”. In: *IEEE transactions on industrial electronics* 69.3 (2021), pp. 2927–2937.

[306] Lin Li et al. “Lane-change intention inference based on RNN for autonomous driving on highways”. In: *IEEE Transactions on Vehicular Technology* 70.6 (2021), pp. 5499–5510.

[307] Qian Shi and Hui Zhang. “An improved learning-based LSTM approach for lane change intention prediction subject to imbalanced data”. In: *Transportation research part C: emerging technologies* 133 (2021), p. 103414.

[308] Ting Xu et al. “Recognition of lane-changing behaviour with machine learning methods at freeway off-ramps”. In: *Physica A: Statistical Mechanics and its Applications* 567 (2021), p. 125691.

[309] Yasir Ali et al. “Predicting and explaining lane-changing behaviour using machine learning: A comparative study”. In: *Transportation research part C: emerging technologies* 145 (2022), p. 103931.

[310] Neha Konakalla, Avrum Noor, and Josh Singh. *Cnn, cnn encoder-rnn decoder, and pretrained vision transformers for surrounding vehicle lane change classification at future time steps*. 2022.

[311] Yeping Hu et al. “Causal-based time series domain generalization for vehicle intention prediction”. In: *2022 International Conference on Robotics and Automation (ICRA)*. IEEE. 2022, pp. 7806–7813.

[312] Yuming Wu et al. “Recognition of Lane Changing Maneuvers for Vehicle Driving Safety”. In: *Electronics* 12.6 (2023), p. 1456.

[313] Renteng Yuan et al. “A unified modeling framework for lane change intention recognition and vehicle status prediction”. In: *Physica A: Statistical Mechanics and its Applications* 632 (2023), p. 129332.

[314] Kai Gao et al. “Dual transformer based prediction for lane change intentions and trajectories in mixed traffic environment”. In: *IEEE Transactions on Intelligent Transportation Systems* 24.6 (2023), pp. 6203–6216.

[315] Titong Jiang et al. “Intention-aware interactive transformer for real-time vehicle trajectory prediction in dense traffic”. In: *Transportation research record* 2677.3 (2023), pp. 946–960.

[316] Yufeng Chen et al. “Vehicle lane-change intention recognition based on BiLSTM Attention model for the Internet of vehicles”. In: *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering* (2024), p. 09544070241240225.

[317] Nir Aharon, Roy Orfaig, and Ben-Zion Bobrovsky. *BoT-SORT: Robust Associations Multi-Pedestrian Tracking*. 2022. arXiv: 2206.14651 [cs.CV]. URL: https://arxiv.org/abs/2206.14651.

[318] Yifu Zhang et al. *ByteTrack: Multi-Object Tracking by Associating Every Detection Box*. 2022. arXiv: 2110.06864 [cs.CV]. URL: https://arxiv.org/abs/2110.06864.