A Predictive-Prescriptive Safety Framework at Intersections in a Connected Vehicle Environment

by

Shurui Zhang

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Civil Engineering)
in the University of Michigan
2022

“玉不琢, 不成器.”

– [礼记]

Shurui Zhang

shuruiz@umich.edu

ORCID iD: 0000-0003-3249-0617

DEDICATION

To mom, dad, and my grandparents.

ACKNOWLEDGEMENTS

How blessed I am to make it this far! Foremost, I would like to take a moment to express my sincere gratitude to my dear advisor, Dr. Neda Masoud. Without any exaggeration, Dr. Masoud is surely my role model as a researcher and as a friend. Being advised by Dr. Masoud is amazing, it makes me feel lucky and proud all the time. Her patience and dedication help me out every time I struggle along this journey. I do not see any barriers in front of me to share ideas and thoughts with her. She profoundly impacts me. It is a great sorrow for me realizing that I have to move forward with much less of her advisory in the future. I wish time would slow down.

I want to thank Dr. Yafeng Yin, Dr. Quentin Stout, and Dr. Gabor Orosz for serving on my dissertation committee. Their constructive instructions and feedback help me better navigate my research and see problems from different perspectives. I would like to thank Dr. Henry Liu and Dr. Tierra Bills for serving on my preliminary exam committee. Studying in the program is such a wonderful experience, and supports from faculty and staff members at Civil and Environmental Engineering make the journey even more unforgettable. I sincerely appreciate that.

I am very grateful for the supports and collaborations from Dr. Rajesh K. Malhan, Dr. Mahdi Bandegi, Joseph Lull and Dr. Wei Zhang at DENSO America Inc., which has sponsored the work of this dissertation. Working with them is full of joy. With their help we have had the chance to investigate theoretical methodologies and frameworks in real world systems, both in Ann Arbor, MI and Dublin, OH. These collaborations allow me to realize the beauty of the transportation field. It is the collaboration between academia, industry, and government that makes the field functional and impactful.

I also want to thank Dr. Shunxin Yang and Yanfeng Cui from the Southeast University, who guided me through my early stage of research in the transportation field as an undergrad student. I am fortunate enough to have friends and loved ones around me in this journey. Studying across the pacific ocean is never easy, while caring and love from my family make things less bitter. Mom and dad have kept encouraging and supporting me to pursue things I love since I was little. They are the source of my faith. World changes, and my love to them deepens. A special thank goes to my love Min, for always being there and brightening my life, the right person at the right time. Thank you to my dear friends, including, but not limited to, fellows at the next generation transportation systems program: Dr. (YEAH!) Amirmahdi Tafreshian, Dr. Sahar Hemmati, Yiyang Wang, Daniel Vignon, Mojtaba Abdolmaleki, Zhen Yang, Xiangguo Liu, Xintao Yan, Xingmin Wang, Dr. Edmond Huang, Dr. Yan Zhao, Dr. Shuo Feng, Dr Xiaotong Sun, Dr. Zhentian Xu, Dr. Zhibin Cheng, Dr. Yiheng Feng, Dr. Roger Lloret, Dr. Mehrdad Shahabi; and Benye Tang, Yabei Cao, Ziyun Xia, Shihao Chen, Yuye Zhang, Yicheng Zhou, Dr. Yongxi Li, Chenchen Ma, Dr. Lu Gan, Zhicheng Zhang, Zhehao Wang. Your company makes Ann Arbor the most memorable place to me.

During the third year of my Ph.D. study, in winter term, the COVID-19 pandemic hit fiercely, leading to a “new normal”. At the moment writing this down, global cases top 531,379,175, with 6,310,376 deceased. Thank you everybody in the fight against COVID-19.

Every milestone starts a new chapter. The future awaits!

TABLE OF CONTENTS

DEDICATION ii

ACKNOWLEDGEMENTS iii

LIST OF FIGURES viii

LIST OF TABLES x

LIST OF ALGORITHMS xi

ABSTRACT xii

CHAPTER

I. Introduction 1

1.1 Motivation 1
1.2 Challenges 3
1.3 Contribution and Dissertation Outline 5
1.3.1 Predicting Risky Driving in a Connected Vehicle Environment 5
1.3.2 Step Attention: Sequential Pedestrian Trajectory Prediction 6
1.3.3 A Learning-based Trajectory Prediction Approach for Heterogeneous Traffic Agents 7
1.3.4 Predictive Trajectory Planning for Autonomous Vehicles at Intersections Using Reinforcement Learning 7

II. Predicting Risky Driving in a Connected Vehicle Environment 9

2.1 Introduction 9
2.2 Related Work 11
2.3 Methodology 16
2.3.1 Feature Selection 16
2.3.2 Time Series K-means 18
2.3.3 iForest 21

2.3.4 Risk Probability 22
2.3.5 Real-time Assessment 22
2.4 Experiments 24
2.4.1 Pre-processing Process 25
2.4.2 Time Series K-means 26
2.4.3 iForest 28
2.4.4 Overall Framework 29
2.5 Conclusion 34

III. Step Attention: Sequential Pedestrian Trajectory Prediction 37

3.1 Introduction 37
3.2 Related Work 40
3.2.1 Pedestrian Trajectory Prediction Methods 40
3.2.2 Attention Mechanism 44
3.3 Methodology 45
3.3.1 Problem Statement 46
3.3.2 Step Attention 46
3.4 Experiments and Results 52
3.4.1 Data Collection and Preparation 53
3.4.2 Experiments on the TrajNet Dataset 54
3.4.3 Experiments at Signalized Intersections 56
3.4.4 Error Sources 61
3.4.5 Computational Efficiency 61
3.5 Conclusion 61

IV. A Learning-based Trajectory Prediction Approach for Heterogeneous Traffic Agents 63

4.1 Introduction 63
4.2 Related work 65
4.3 Methodology 68
4.3.1 Problem Statement 68
4.3.2 Model Architecture 68
4.3.3 Prediction and Training Settings 69
4.3.4 Model Evaluation 70
4.4 Experimental Results 71
4.4.1 Result Performance 71
4.4.2 Transfer Learning Performance 73
4.5 Conclusion 75

V. Predictive Trajectory Planning for Vehicles at Intersections with Deep Reinforcement Learning 76

5.1 Introduction 76

5.2 Related Work . . . 78
5.2.1 Trajectory Prediction . . . 78
5.2.2 Driving Risk Prediction . . . 81
5.2.3 Vehicle Trajectory Planning . . . 81
5.2.4 Reinforcement Learning . . . 83
5.3 Methodologies . . . 84
5.3.1 Problem Statement . . . 84
5.3.2 Scenario Generation . . . 86
5.3.3 Predictive Models . . . 93
5.3.4 Deep Reinforcement Learning-based Trajectory Planning . 95
5.4 Experiments and Results . . . 104
5.4.1 Experiment Data . . . 104
5.4.2 Calibrating the BGM Models . . . 104
5.4.3 Reduction of the Exploration Space . . . 106
5.4.4 Performance . . . 108
5.4.5 Comparison to Human Driving . . . 112
5.5 Conclusion . . . 115

VI. Conclusion and Discussions . . . 117

6.1 Research Findings and Conclusions . . . 117
6.2 Future Directions and Challenges . . . 119

APPENDIX . . . 122
A.1 Constant Speed Reference Path . . . 123

BIBLIOGRAPHY . . . 128

LIST OF FIGURES

Figure

1.1 The predictive-prescriptive trajectory planning framework 5
2.1 Real-time forecasting of driving risk. The trajectory adopts the RDP score of the cluster to which it is assigned. 23
2.2 The subject intersection located at the Huron parkway and Plymouth road, in Ann Arbor, MI 25
2.3 (a) raw and (b) pre-processed data 26
2.4 Feature illustration of raw trajectory data 27
2.5 Anomaly scores of training samples based on the trained iForest model . . 29
2.6 DTW-based time series k-means clustering results 30
2.7 Features and their 95% confidence intervals for normal and anomalous trajectories in clusters. The gray and blue regions show the 95% confidence intervals of normal and anomalous trajectories, respectively. 31
2.8 Contamination rate sensitivity analysis 32
2.9 Real time assessment of incoming trajectories 34
3.1 Step attention architecture 47
3.2 An LSTM-RNN layer illustration 48
3.3 The sequential learning strategy for trajectory prediction 52
3.4 The average displacement error at each prediction time step across the test dataset of the proprietary dataset. The shaded region highlights the 95% confidence interval 57
3.5 The average displacement error at each prediction time step across the test dataset of the complementary dataset. The shaded region highlights the 95% confidence interval 58
3.6 Prediction results for the case study in an intersection. The observed trajectories, i.e., 4-seconds-long history trajectories used for training, are colored in blue, while the predicted and the ground truth trajectories are colored in green and red, respectively. 59
3.7 Learning curve of the proposed model on the complementary dataset . . . 60
4.1 Comparison of the displacement errors for all agent types 71
4.2 Visualization of predicted trajectories 71
4.3 Transfer learning results 74
4.4 Probability distribution of three type of agents 75

5.1 An overview of the proposed methodology. Shaded boxes are the unique contributions of this chapter. 85
5.2 Scenario encoding workflow 90
5.3 Scenario generation workflow 91
5.4 Illustration of extracting a graph from a scenario at a given time step. The green circle represents the subject vehicle and ovals linked to it represent the surrounding agents. 97
5.5 POMDP diagram of the formulated problem 99
5.6 The proposed DQN architecture 101
5.7 Illustration of the two intersections selected for experimentation. 105
5.8 Finding the best combination for the number of components in the three BGM models 107
5.9 Illustrations of possible regions for the subject vehicles to explore within 5 seconds on example driving scenarios 109
5.10 Reward trends of model training 110
5.11 Planning results on example cases. Planned trajectories for the subject vehicle are in red. Ground truth of vehicle, pedestrian and cyclist trajectories are plotted in blue, yellow and gray, respectively. Stars mark the positions of agents at the planning start time. 111
5.12 Model performance compared to human driving 114
A.1 Reward trends of model training 125
A.2 Model performance compared to human driving 126

LIST OF TABLES

Table

2.1 Summary of the literature on identifying risky and aggressive driving . . 14
2.2 Inter- and intra-cluster distances measuring the similarity between and within clusters, respectively 28
3.1 Benchmark Performances 54
3.2 ADE and FDE metrics reported in the form ADE/FDE for all TrajNet sub-datasets for step attention and the three benchmarks 55
4.1 Overall benchmark performance. The values in each cell show the ADE/FDE metrics in meters. 72
5.1 Result metric statistics 115
A.1 Result metric statistics using a constant speed reference path 124

LIST OF ALGORITHMS

Algorithm

1 EKF algorithm for trajectory prediction 70

2 Scenario encoding and trajectory pool construction 89
3 Socially acceptable scenario generation with Gibbs sampling 92
4 Scenario graph construction 97
5 The proposed deep Q learning algorithm 103

ABSTRACT

The connected and automated vehicle (CAV) technology in recent years has demonstrated its potential in improving efficiency in transportation systems. Prediction, as a key component of the technology, enables smart vehicles to anticipate future movements of traffic agents and potential future risks, so as to plan in advance by incorporating these predictions in their trajectory planning. In this dissertation I propose a prediction-based framework to identify risky scenarios at urban intersections, develop strategies to mitigate them, and conduct prediction-based vehicle trajectory planning in a connected environment. The framework consists of three main components: (1) real-time risky driving prediction; (2) traffic agent trajectory prediction; (3) prediction-based vehicle trajectory planning.

For risky driving prediction, I propose an unsupervised learning framework to predict risky driving at urban intersections in a connected vehicle environment. The proposed framework uses time series k-means to categorize multi-dimensional time series trajectories into several context-aware driving patterns. I train an anomaly detection model on the trajectory dataset to identify anomalous trajectories, and apply this model to clusters of driving patterns to provide *Risky Driving Prediction* (RDP) scores for each driving pattern. I provide a real-time online assessment approach to predict the risk score of driving trajectories that travel toward a signalized intersection. I use real-world connected vehicle trajectories collected by a road-side unit (RSU) in Ann Arbor, Michigan to implement the framework. I use several quantitative measures as well as illustrative tools to validate the model, and further discuss how the RDP framework can be used to develop network-level and individual vehicle-level insurance and safety focused applications.

For traffic agent trajectory prediction, I propose a deep learning model, which I call step attention. The model learns trajectory patterns directly from input trajectories. For pedestrian trajectory prediction, I evaluate the model using a benchmark dataset and a proprietary dataset collected at urban intersections. I compare the performance of step attention with three existing state-of-the-art algorithms. Experiments show that on the benchmark dataset the average and final displacement errors of step attention for a 4.8-seconds prediction horizon are 0.53 and 1.72 meters, respectively. Both average and final displacement errors are favorable compared to the benchmark methods. Furthermore, on the urban intersection dataset, the proposed model has an average displacement error of 0.74 meters and a final displacement error of about 1.40 meters for a 6-seconds prediction horizon. I conduct a complementary set of experiments to further investigate model performance in a real-world intersection. In these experiments, the model gains an ADE/FDE of 0.76/1.70 m. The proposed model also produces accurate prediction results on different scenarios composed of different walking patterns (e.g., straight and curvy) and different environments (e.g., sidewalk and street). The average displacement errors on all datasets are within the length of a single step of an adult. The experiments also indicate that the displacement error grows almost linearly with the prediction horizon. Finally, I extend the architecture to different agents, namely, vehicles and cyclist, and demonstrate that step attention can perform well on long look-ahead trajectory prediction tasks for heterogeneous agents.

With aforementioned models and results, I put forward a predictive trajectory planning framework to help autonomous vehicles plan future trajectories. I develop a partially observable Markov decision process (POMDP) to model this sequential decision making problem, and a deep reinforcement learning solution methodology to learn high-quality policies. The POMDP model utilizes driving scenarios, condensed into graphs, as inputs, to devise safe, comfortable, and energy-efficient trajectories for the subject vehicle to follow. I propose a simulation framework to generate *socially acceptable* driving scenarios using a real world autonomous vehicle dataset. The simulation framework utilizes Bayesian Gaussian mixture models to learn trajectory patterns of different agent types, and Gibbs sampling to ensure that the distribution of simulated scenarios matches that of the real-world dataset collected by an autonomous fleet. I evaluate the proposed work in two complex urban driving environments: a non-signalized T-junction and a non-signalized lane merge intersection. Both environments provide vastly more complex driving scenarios compared to a highway driving environment, which has been mostly the focus of previous studies. The framework demonstrates promising performance for 5 seconds planning horizons. I compare safety, comfort, and energy efficiency of the planned trajectories against human-driven trajectories in both experimental driving environments, and demonstrate that it outperforms human-driven trajectories in a statistically significant fashion in all aspects.

CHAPTER I

Introduction

1.1 Motivation

Connected and automated vehicle (CAVs) technology is an emerging topic in the transportation field, and it is regarded as one of the pillars shaping the next generation of transportation systems. In the past decades, the rapid development of communication technologies, such as dedicated short range communication (DSRC), 5G, etc., and artificial intelligence (AI), has brought what was once considered science fiction closer than ever to reality. Connectivity and autonomy are two major properties differentiating CAVs from traditional vehicles.

Connected vehicle (CV) technology enables vehicles to communicate with other road users or infrastructures through embedded communication modules with low latency (Lu et al. (2014); Siegel et al. (2017)). More specifically, according to the type of the receiver during the communication process, connectivity can be defined as: i) Vehicle-to-vehicle (V2V), where communication occurs between vehicles; and ii) Vehicle-to-infrastructure (V2I), where communication occurs between vehicles and road infrastructures. V2V and V2I can be combined and referred to as Vehicle-to-everything (V2X). With V2X, vehicle information is transmitted through Basic Safety Messages (BSMs), with standardized contents. A BSM typically contains identification of the sender vehicle, its location and real-time motion status, etc. (Kamrani et al. (2018)).

The development of automated vehicle (AV) technology can be dated back to the late 20th century (Fenton et al. (1976); Shiller et al. (1991)). A fully automated vehicle, i.e., autonomous vehicle, is anticipated to operate safely and efficiently in daily transportation systems without human intervention. The introduction of CAVs to existing transportation systems is envisioned to bring about other benefits, which includes but are not limited to lowering greenhouse gas emissions (Greenblatt and Saxena (2015)), reducing traffic congestion (Wang et al. (2017)), boosting economy (Fagnant and Kockelman (2015)), etc.

From profitability to privacy and equity, debates regarding CAVs have never stopped since the birth of the technology. Among them a core issue is the reliability of the CAV technology itself. The development of reliable CAV technology is challenging due to the complexity of transportation systems. When traveling on roads, a CAV is required to interact with its dynamically changing surrounding environment, e.g., weather, other road users, road infrastructures, etc., in real-time, and make appropriate driving decisions. The presence of road users including pedestrians and cyclists, which can be more vulnerable than machinery, introduces additional requirements to CAVs. Such challenges can be enhanced in urban areas, especially at urban intersections, which host more complex traffic environments.

With the advancements in the field, more and more CAVs have begun hitting public roads and testing the reliability of the technology, thereafter a large volume of defects in existing CAV-related systems have been exposed. Although systems are well-engineered, CAV-involved crashes and accidents still emerge occasionally (Petrović et al. (2020); Favarò et al. (2017)). A recent study shows that 74% of respondents do not trust AVs, nor do they believe that AVs can outperform a human driver. Participants also believed that AVs would be prone to malfunction (Othman (2021)). The *safety* concern of CAV-related systems is further put under the spotlight by the public for the future development of these systems (NHTSA (2017); Nunes et al. (2018)). "How to benefit from the CAV technology while guaranteeing safety of our daily transportation systems" has become an non-negligible question for researchers, policy makers, and CAV manufacturers.

Successful development and deployment of CAV systems requires combining the strengths of technology with those of human drivers. Human drivers typically anticipate what will happen next in their surrounding traffic environment before making driving decisions. For example, a driver may pause her right turn at an intersection if she anticipates a pedestrian is about to cross the street, and waits until she feels safe enough to finish the right turn. This property equips human drivers with the ability to avoid potential accidents, and lowers the risks they pose to their surrounding road users. Similarly, such predictive-prescriptive mechanism can be integrated to CAV systems to improve their safety performance. With connectivity and other sensor technologies, CAVs are able to obtain information about other road users in their surrounding environment. This information can be used to inform predictive models that CAVs can employ to anticipate the future status of their surrounding environment. Such predictions can then be taken into consideration to improve a CAV's performance as well as its interactions with the rest of the transportation system. The aforementioned challenges and the potential benefits of developing safety-focused CAV systems are the main drivers of the proposed framework in this dissertation, which focuses on one of the most complex driving environments–urban intersections. To estimate where an agent is heading, trajectory prediction models are proposed within the framework to predict long look-ahead future trajectories of traffic agents. To further anticipate the risk brought by surrounding vehicles, a driving risk prediction framework is proposed. Based on these predictive models, a predictive trajectory planning framework is proposed to enable CAVs to make informed driving decisions.

1.2 Challenges

Four main challenges currently hinder successful development and implementation of a predictive-prescriptive CAV-related framework:

- **Uncertainty**: The surrounding environment of CAVs is composed of a number of components, such as weather, road surface condition, road geometry, traffic volume, etc. Many of these factors are time-sensitive and geo-location dependent. Therefore, when traveling on roads, the surrounding environment of a CAV keeps changing. Such changes are difficult to enumerate exhaustively. Hence, part of the uncertainty originates from the inability of CAVs to exactly capture all the factors in the surrounding environment that may impact them. Another source of uncertainty is related to other road users. Human behaviors are complex by nature as an individual's intention and psychological state may change frequently, and heterogeneity of behaviour exists between different individuals. This behavior can further be affected by physical or social factors (Staats and Staats (1963); Gardner and Stern (1996)). Uncertainty affects the accuracy of the output of the framework, and thereby its effectiveness and safety implications. As such, uncertainty should be embedded into any trajectory planning framework.

- **Scalability**: To be able to operate in real-world systems, such a framework should be scalable. When it comes to transportation systems and CAVs, the number of potential users can grow dramatically when service areas are enlarged. As such, the framework should be able to support a potentially increasing number of system users and regions.

- **Generalizability**: Existing research in the CAV field relies heavily on learning-based approaches. By examining existing data samples, learning-based methods extract features from the data, and gain knowledge about the hidden patterns to facilitate system functionalities. If not formulated and developed properly, data-driven approaches can easily lead to poor out-of-sample performance. In other words, the generalizability of the models can be impaired, leading to systems that work well in specific regions, but perform poorly on others. A generalizable model ensures the validity and performance of the system when it is transferred from one location to another.

- **Real-time responsiveness**: Real-world environments change rapidly, and therefore decision making in such environments is time-sensitive. A high-quality but late decision by a CAV can easily result in a disaster. As such, for CAVs operating in the real world, the framework should generate reliable outputs within reasonable time to maintain its effectiveness.

1.3 Contribution and Dissertation Outline

This dissertation puts forward a general predictive-prescriptive safety framework that utilizes infrastructure-based and vehicle sensors to monitor intersections with the objective of predicting future trajectories of agents, identifying high-risk scenarios and mitigating them, and generating high-quality driving polices for vehicles. The framework includes three main modeling modules, namely a risky driving prediction module, a traffic agent trajectory prediction module, and a vehicle trajectory planning module that are embedded within a control framework. The framework and its modules are displayed in Figure. 1.1. Each of these modules is elaborated in the following.

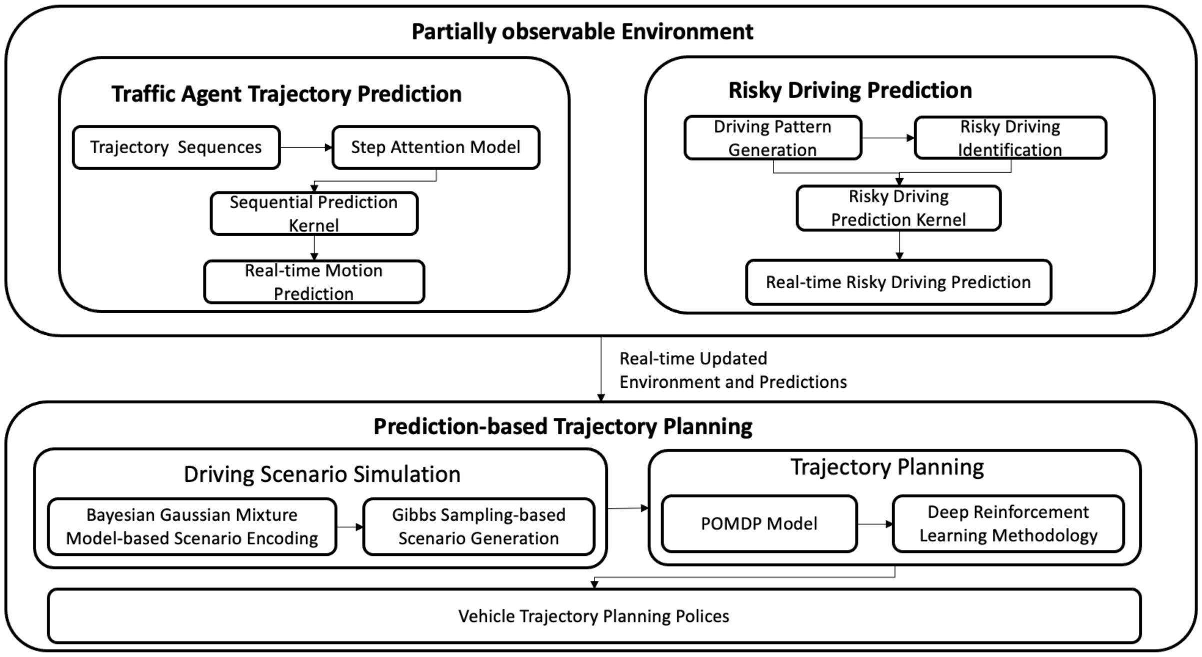

Partially observable Environment
Traffic Agent Trajectory Prediction
Trajectory Sequences
Step Attention Model
Sequential Prediction Kernel
Real-time Motion Prediction
Risky Driving Prediction
Driving Pattern Generation
Risky Driving Identification
Risky Driving Prediction Kernel
Real-time Risky Driving Prediction
Real-time Updated Environment and Predictions
Prediction-based Trajectory Planning
Driving Scenario Simulation
Bayesian Gaussian Mixture Model-based Scenario Encoding
Gibbs Sampling-based Scenario Generation
Trajectory Planning
POMDP Model
Deep Reinforcement Learning Methodology
Vehicle Trajectory Planning Polices

Figure 1.1: The predictive-prescriptive trajectory planning framework

1.3.1 Predicting Risky Driving in a Connected Vehicle Environment

In Chapter II I propose an unsupervised learning framework to predict risky driving at intersections in a connected vehicle environment. The proposed framework uses time series k-means to categorize multi-dimensional time series trajectories into several context-aware driving patterns. Dynamic time warping (DTW) is implemented within the time series k-means algorithm for measuring the similarity between trajectories. DTW is adopted to make the framework robust to temporal distortions and missing data. I train an isolation forest model on the trajectory dataset to identify anomalous trajectories, and apply this model to clusters to provide Risky Driving Prediction (RDP) scores for each driving pattern. I provide a real-time online assessment approach to predict the probability of aggressiveness for driving trajectories that move toward a subject intersection. I use real-world connected vehicle trajectories collected by a road-side unit (RSU) to implement this framework. Several quantitative measures and illustrative tools are used to validate this model. I discuss how safety-focused warning systems at the individual vehicle level as well as at the system level can be developed using this framework.

1.3.2 Step Attention: Sequential Pedestrian Trajectory Prediction

In Chapter III I propose a deep learning model, which I call step attention, for pedestrian trajectory prediction. The model has a special architecture which consists of recurrent neural networks, convolutional neural networks, and an augmented attention mechanism. Rather than developing architectures to model factors that may affect the walking behavior, the step attention model learns trajectory patterns directly from input sequences. I evaluate the performance of the step attention model using TrajNet–a publicly available benchmark dataset collected from a diverse set of real-world crowded scenarios. I compare the performance of step attention with three existing state-of-the-art algorithms, including social LSTM, social GAN, and occupancy LSTM on the TrajNet benchmark dataset. Experiments show that the average displacement error (ADE) of step attention for a 4.8-seconds-long prediction horizon is about 0.53 m. The final displacement error (FDE) is 1.72 m. Both average and final displacement errors are favorable compared to the benchmark methods. I conduct a second set of experiments using data collected from a four-way intersection through roadside camera sensor platforms to study the effectiveness of the proposed model in uncrowded environments. On this dataset, the proposed model has an ADE of 0.74 m and a FDE of about 1.40 m for a 6-seconds-long prediction horizon. A complementary set of experiments is conducted to further investigate model performance in a real-world intersection. In these experiments, the model gains an

ADE/FDE of 0.76/1.70 m. The proposed model also produces accurate prediction results on different scenarios composed of different walking patterns (e.g., straight and curvy) and different environments (e.g., sidewalk and street). The average displacement errors on all investigated datasets are within the length of a single step of an adult. The experiments also indicate that the displacement error grows almost linearly with the prediction horizon.

1.3.3 A Learning-based Trajectory Prediction Approach for Heterogeneous Traffic Agents

In Chapter IV I present a learning-based trajectory prediction method for different road users, including vehicles, pedestrians, and cyclists. The model uses history position information of traffic agents, and predicts future positions of subjects within a finite horizon. Instead of developing different model architectures for different agent types, a generic model architecture is used to learn trajectory patterns. This common architecture is then trained using agent-specific datasets, providing individualized models for different agent types. I evaluate the model on the Lyft dataset–a publicly available dataset collected by a set of autonomous vehicles–and compare its performance against extended Kalman filter (EKF) as a benchmark. Results indicate that the learning-based method outperforms the benchmark method and provides high accuracy predictions in 5-second prediction horizons across all agent types. I also show that the prediction accuracy on rarely-seen agents can be greatly improved using transfer learning.

1.3.4 Predictive Trajectory Planning for Autonomous Vehicles at Intersections Using Reinforcement Learning

In Chapter V I put forward a predictive trajectory planning framework to help autonomous vehicles plan future trajectories. I develop a partially observable Markov decision process (POMDP) to model this sequential decision making problem, and a deep reinforcement learning solution methodology to learn high-quality policies. The POMDP model utilizes driving scenarios, condensed into graphs, as inputs. More specifically, an input graph contains information on the history trajectory of the subject vehicle, predicted trajectories of other agents in the scene (e.g., other vehicles, pedestrians, and cyclists), as well as predicted risk levels posed by surrounding vehicles to devise safe, comfortable, and energy-efficient trajectories for the subject vehicle to follow. In order to obtain sufficient driving scenarios to use as training data, I propose a simulation framework to generate *socially acceptable* driving scenarios using a real world autonomous vehicle dataset. The simulation framework utilizes Bayesian Gaussian mixture models to learn trajectory patterns of different agent types, and Gibbs sampling to ensure that the distribution of simulated scenarios matches that of the real-world dataset collected by an autonomous fleet. I evaluate the proposed work in two complex urban driving environments: a non-signalized T-junction and a non-signalized lane merge intersection. Both environments provide vastly more complex driving scenarios compared to a highway driving environment, which has been mostly the focus of previous studies. The framework demonstrates promising performance for planning horizons as long as 5 seconds. I compare safety, comfort, and energy efficiency of the planned trajectories against human-driven trajectories in both experimental driving environments, and demonstrate that it outperforms human-driven trajectories in a statistically significant fashion in all aspects.

CHAPTER II

Predicting Risky Driving in a Connected Vehicle Environment

2.1 Introduction

The next generation of transportation systems is envisioned to be greatly influenced by a number of technological advancements, including the connected and automated vehicle (CAV) technology (Zhang and Masoud (2020)). The connected vehicle (CV) technology enables a vehicle to be in constant communication with other road users and infrastructures trough broadcasting and receiving basic safety messages (BSMs), which contain information on the vehicle's location, speed, etc. CV technology is anticipated to enhance mobility by enabling innovative mobility systems (Zhang et al. (2020); Masoud and Jayakrishnan (2016); Abdolmaleki et al. (2019)). Additionally, CVs can increase safety in the transportation system through reducing vehicle perception times, allowing vehicles to 'see' beyond line of sight, and providing vehicles with frequent status updates on their local neighbourhood. CV technology can also help better regulate the traffic flow (Liu et al. (2020)), which contributes to reduction of accidents by reducing congestion and moderating driver fatigue and frustration, and reducing the potential for poor driver decisions and reactions that contribute to accidents (van Wyk et al. (2019); Jeong et al. (2018)). In addition, CV technology can inform safety focused predictive models that allow for anticipating risky scenarios in different contexts and proactively taking action to mitigate them. Such predictive models can help reduce crash rates by monitoring risky drivers and issuing warning messages to all road users in a connected vehicle environment or advising traffic controllers to adjusting traffic light timing to avoid crashes. In this chapter, I develop a framework for risky driving prediction at intersections. This framework allows the infrastructure to cluster trajectories into different categories with different driving patterns as they approach an intersection, and provides predictive RDP scores in real-time, thereby granting the intersection time to take action (e.g., warn other road users or adjust traffic light timing) to mitigate potentially unsafe situations.

A challenge in adopting supervised learning methods in predictive frameworks is the need for manual labeling of trajectories. As such, the proposed framework has been designed based on an unsupervised learning framework to ensure model scalability and avoid introducing bias through subjective labeling. First, since driving behavior can be dependent on geolocation and the driving environment, labeling will be needed when transferring a supervised learning model from one intersection to another. This limits the scalability of the framework. The need for labeling can also become a bottleneck if one needs to update the model periodically based on newly collected data. In this case, manual labeling can be time-consuming, thereby limiting the ability of the system to adapt and update the model in real-time. Secondly, supervised learning methods tend to introduce labeling bias (Barbosa and Chen (2019)); that is, manual labeling introduces subjectivity in identifying risky driving behavior.

For a vehicle approaching a traffic intersection, the proposed framework maps the vehicle trajectory into a set of cluster centroids, where each cluster is associated with a driving pattern and a probabilistic measure of risk. As the vehicle approaches the intersection, more of its trajectory profile is revealed, allowing the framework to monitor the vehicle's driving behavior and update its RDP score in real-time. I use real-world connected vehicle data, collected by the Ann Arbor Connected Vehicle Test Environment (AACVTE) Safety Pilot Model Deployment project (Woodhouse (2014)), to identify and predict risky driving behavior in real-time. AACVTE is considered as the largest CV test facility around the world to this date, containing daily trajectories from over 2500 CVs and 75 RSUs (AACVTE (2020)). Connected vehicle data provides standardized safety messages that contain basic motion-related vehicle information with low overhead (Kamrani et al.

(2018)). Using CV data, I construct multi-dimensional time series input features, including measures on basic vehicle motion information, energy, and force to capture and differentiate driving styles.

The proposed approach constructs a driving risk scale at the vehicle trajectory level, which can be used by insurance companies and city traffic engineers to monitor and quantitatively measure driving risk and intersection safety, respectively (Zhang et al. (2021)). The proposed framework does not require pre-labeled training data, which makes it scalable and cost-effective to implement in practice. This is an important feature of the proposed method, since in transportation systems driving patterns could be significantly dependent of geolocation and the driving environment, rendering manual labeling of data impractical.

2.2 Related Work

Traffic intersections are scenes to about 50% all of the combined total of fatal and injury crashes within the United States (USDOT (2018)). Since 1990s, researchers have been attempting to identify risky and aggressive driving behaviors that play a role in road traffic accidents (Lajunen and Parker (2001); Houston et al. (2003); Mizell et al. (1997); Constantinou et al. (2011)). Many studies have focused on improving intersection safety with consideration of driving behavior (Wali et al. (2018); Aycard et al. (2011)). Risky driving can be marked by speeding, harsh acceleration or deceleration, ease of getting agitated by other drivers, frequent honking, cutting across one or more lanes in front of other vehicles, passing on the shoulders, etc. (Aljaafreh et al. (2012); Shinar and Compton (2004)). There are many factors that may result in risky driving, such as driver's personality and the driving environment (Ma et al. (2020); Alonso et al. (2019)). Regardless of what factors lead to risky driving, it overwhelmingly manifests itself in vehicle trajectories through unusual changes in acceleration, as it directly changes vehicle motion (Hong et al. (2014); Ma et al. (2020); Moukafih et al. (2019)). As such, abnormal accelerating and decelerating behaviors are considered to be main indicators of risky driving patterns.

There are a number of studies in the transportation literature that focus on risky and aggressive driving prediction. The strong majority of these studies rely on survey data to develop models for predicting risky driving patterns and driver aggressiveness. J. Deffenbacher et al. (Deffenbacher et al. (1994)) developed a driving anger scale through self-reported accident frequency surveys. Based on this scale, the authors categorized drivers into several classes, where each class was associated with an angry level ranging from *'not at all'* to *'very much'*. Similarly, E. Constantinou (Constantinou et al. (2011)) investigated the personality factors that contribute to driving risk. J. DePasquale et al. (DePasquale et al. (2001)) developed a propensity-for-angry-driving scale based on survey data to measure aggressiveness in drivers. Using survey data, B. Krahe et al. (Krahé and Fenske (2002)) provided an approach to predict aggressive driving behavior based on drivers' personality and attributes of their of vehicles, e.g., vehicle power.

Relying merely on survey data for predicting risky driving could result in training models that are biased and/or non-generalizable. Survey-based data can be subjective, as survey respondents could have different understandings of risky driving behavior (Dula and Geller (2003)). Furthermore, survey-based models typically focus on driver-level attributes, e.g., driving experience, age, gender, etc., failing to incorporate driving environmental and situational contexts that could affect driving patterns.

To avoid the drawbacks of survey-based approaches, in recent years some studies have started leveraging information generated by vehicles and sensors to train models that can detect risky driving in a timely manner. Among them there are a few studies that use pre-labeled datasets to train models or compare a trajectory with "templates"/thresholds, so as to detect risky and aggressive driving behavior. B. Bose et al. (Bose et al. (2018)) proposed a sensor-based classification model to classify trajectories into aggressive or non-aggressive, and provided binary classification results. J. Lee et al. (Lee and Jang (2019)) proposed a framework to diagnose aggressive driving through thresholds based on in-vehicle records such as steering and yaw rate information. A. Aljaafreh et al. (Johnson and Trivedi (2011)) used smartphone as a sensor platform to recognize driving styles. They used sensors available on smart phones, such as camera and microphone, to detect drivers' gestures and driving events, e.g., accelerating and decelerating. By comparing these events to aggressiveness "templates", they categorized driving styles into typical (non-aggressive) or aggressive. Similarly, H. Eren (Eren et al. (2012)) used smartphone data to train a Bayesian classification model to mark unsafe driving behavior. O. Karaduman et al. (Karaduman et al. (2013)) utilized information from a vehicle's CAN Bus to detect driver aggressiveness. Their method selected a number of essential features from the CAN bus and provided a binary detection of vehicle aggressiveness. P. Droz et al. (Droz and Zhu (2014)) proposed a method to identify nearby aggressive drivers based on their estimated distance, with the goal of providing a control strategy for the subject vehicle. Y. Ma et al. (Ma et al. (2020)) proposed a conceptual road model and performed on-line analysis on vehicle planes based on in-vehicle kinematic data to identify aggressive driving based on road alignments.

Compared to survey-based approaches, this class of studies can provide more timely results on identification of risky driving. However, they only provide binary detection results (i.e., safe or unsafe, risky or non-risky). These studies fail to recognize that not all risky driving behaviors are similar, thereby requiring different mitigation strategies. As such, they may not be sufficient for real-world safety focus applications. In practice, different responses may be needed for different risky driving behaviors. For example, harsh braking and speeding may both be marked as risky driving behaviors by a binary classification model. However, they may impose different risk levels and require different responses from a mitigation response system (Collins (2011)). Another drawback of these models is that they require labeled datasets, or manually-labeled risky driving "templates", making the training process cumbersome and subjective, since the labeling criteria is subject to interpretation. In addition, these methods are capable of detecting risks in driving patterns, but have limited ability to forecast future behavior.

As discussed, existing methods used to identify risky driving suffer from one or more of the following drawbacks: (*i*) they are based on descriptive surveys, and therefore may not be sensitive to situational changes in the driving environment; (*ii*) they apply only to the drivers under study, and cannot be necessarily generalized to the driving population; (*iii*) they provide binary results, identifying the driving behavior as normal or abnormal, thereby limiting the set of mitigation strategies that can be taken in response to predictions of risky and aggressive driving; and (*iv*) they require pre-labeled datasets, while the

Study	Automatic data collection	Situational awareness	Automatic labeling	Generalizable	Multiple classes
Deffenbacher et al. (1994)	✗	✗	✗	✗	✓
DePasquale et al. (2001)	✗	✗	✗	✗	✓
Krahé and Fenske (2002)	✗	✗	✗	✗	✓
Johnson and Trivedi (2011)	✓	✓	✗	✓	✗
Eren et al. (2012)	✓	✓	✗	✓	✗
Karaduman et al. (2013)	✓	✓	✗	✓	✗
Droz and Zhu (2014)	✓	✓	✗	✓	✗
This chapter	✓	✓	✓	✓	✓

Table 2.1: Summary of the literature on identifying risky and aggressive driving

labeling process can be cumbersome and subjective. An overview comparison of the existing studies in the literature on risky driving is provided in Table 2.1.

In the era of CV technology, the need for comprehensive and generalizable diagnostic and predictive models is conveniently met with high volumes of data collected through V2V and V2I communications (Zhang et al. (2019)). By leveraging the possibilities brought by connectivity, the proposed method satisfies the critical requirements of a scalable risky driving prediction solution, as detailed in Table 2.1. It utilizes real-world CV data, which is automatically collected by RSUs or CVs. Incorporating situational awareness in predicting risky driving is important since driving risk is not necessarily a personality trait and can vary depending on the driving situation. The proposed framework incorporates situational awareness as it predicts driving risk by identifying trajectories that are anomalous in a pool of trajectories all subjected to the same driving environmental.

Manual data labeling, which is a practice widely adopted in the literature, can enter subjectivity into the process. Furthermore, it can hurt the generalization performance of aggressive driving prediction models, as such models can be dependent on geolocation and the driving environment. Additionally, significant resources are required to perform the labeling task, which may limit the size of the datasets used for training, and possibly generate biased models.

Another benefit of the proposed method is its ability to provide non-binary predictions. Although there are several existing methods that can provide real-time identification of driving risks, they are limited to binary predictions. However, the mere knowledge on whether a driver is risky or not is not sufficient for informing actions by all road users and/or traffic engineers. The non-binary results produced by the proposed model provides a probabilistic view of risky driving, enabling issuing warnings to all road users and for traffic engineers to adopt more informed mitigation strategies.

The contributions of this chapter are as follows. Unlike the majority of the existing literature, which are based on descriptive surveys or are heavily reliant on manual labeling to mark risky and aggressive trajectories, in this study I automate this process by developing an unsupervised learning method, thereby overcoming the shortcomings of manual data labeling, including subjective labeling, potential biased models, and lack of scalability. Secondly, in this work I propose a vehicle motion and physics-based feature selection methodology to construct high-dimensional features based on basic driving information. Furthermore, the resulting clusters can be interpreted as driving patterns, providing context for taking mitigatory actions when a driver belonging to a cluster demonstrates risky behavior. Another contribution of the framework, to the best of my knowledge, is producing probabilistic scores to quantify the level of risk, thereby providing more information that can be leveraged by system users. Finally, I use real world data collected in a CV environment to implement the framework, and validate the results using both quantitative measures as well as illustrative tools.

The rest of the chapter is organized as follows: I start by presenting the framework. This section includes a DTW-based time series k-means algorithm to categorize multi-dimensional time series trajectories into several clusters that correspond to different driving patterns, and applying a trained iForest to clusters to assign a risk score to each driving pattern. Next, I conduct experiments using real-world CV data, and discuss how the framework can be used to enable individual vehicle-level insurance and safety focused applications. I end the chapter with the Conclusion section.

2.3 Methodology

In this section I provide a framework for risky driving prediction. The framework implements unsupervised learning on multi-dimensional connected vehicle trajectories to learn driving patterns, and applies an anomaly detection method to identify risky driving patterns. I start this section by introducing a feature selection methodology to quantitatively measure the driving risk by considering a vehicle's real-time motion. Next, I use a time series k-means algorithm with a DTW similarity measure to cluster multi-dimensional trajectories into groups of driving patterns. Driving patterns at intersections capture differences in driving styles. I then train an iForest model to detect anomalous trajectories and assign risk scores to clusters. An assessment approach is then proposed to predict real-time driving risk in a probabilistic manner. Finally, I describe how the proposed framework can be used to determine the risk score of a trajectory as a vehicle approaches the signalized intersection under study.

2.3.1 Feature Selection

BSMs contain motion-related information including vehicle geolocation, speed, acceleration, heading, etc. (Cronin (2012)). At each time step, I use this information to form a multi-dimensional feature profile that can capture drivers' level of risk. A number of features are constructed separately for each time step, and concatenated over multiple time steps, to form time series feature sequences. The details of feature selection and construction rules are elaborated in the following.

1. Relative location measure, d_t: As I am interested in identifying and forecasting risky driving at intersections, the distance of the subject vehicle to the intersection at time t, denoted as d_t, is incorporated into the feature space.

2. Vehicle motion measures: velocity, v_t, and acceleration, a_t: Research has shown that the driving risk level is typically related to speed and acceleration (Dula and Ballard (2003); Wang and Lukic (2011); Danaf et al. (2015)). For example, risky and aggressive drivers are more likely to drive over the speed limit. Vehicle velocity, denoted by v_t, represents the vehicle motion status at current time t, and vehicle acceleration, a_t, is a measure of change

in vehicle velocity ($a_t = (v_t - v_{t-1})/\Delta t$). Both v_t and a_t carry important information about the level of risk imposed/faced of a driver, and can be obtained through the vehicle's on-board sensors or its controller area network (CAN) bus.

3. Energy measure, Δv_t^2: Risky drivers typically accelerate or apply the brakes more harshly than their peers to achieve a rapid change in acceleration or speed. The functionality of the vehicle brake is to change the kinetic energy into other forms of energy (e.g., thermal energy), while the vehicle accelerator creates kinetic energy to facilitate motion. Vehicle occupants may experience a sharp change in the kinetic energy of the vehicle due to a variety of reasons, including insufficient driving experience, risky driving attitude, or driving environmental factors, e.g., pavement quality or the surrounding traffic. As such, the change in the energy measure corresponds to the intensity of the impact. It should be noted that not all these factors may be directly observable; however, regardless of the underlying reason, the changes in kinetic energy can capture the intensity of such impacts. Therefore, here I incorporate the energy measure in the feature space to capture changes in the kinetic energy of the vehicle. The kinetic energy of a moving vehicle traveling with mass m and velocity v_t at time t can be calculated as $E_t = \frac{1}{2}mv_t^2$. Therefore, at time step t, I can compute the change in kinetic energy as $\Delta E_t = \frac{1}{2}m(v_t^2 - v_{t-1}^2)$, i.e., $\Delta E_t \propto \Delta v_t^2$.

4. Force measure, Δa_t: According to Newton's second law ($F_t = ma_t$), it is the force that moves a vehicle. Therefore, the change in acceleration between two consecutive time steps can capture the change in the vehicle's tractive effort, i.e., $\Delta F_t \propto \Delta a_t$, prompting us to incorporate Δa_t as a feature in the model. Note that force and energy measures capture fundamentally different aspects of vehicle motion while both measures utilize a compressed memory of vehicle motion based on a history of vehicle trajectory, the force measure makes use of a longer temporal history compared to the energy measure, as the force measure at time t contains temporal velocity information from time $t-2$ to time t, and the energy measure contains that from time $t-1$ to time t. Furthermore, these two measures serve different purposes: While the energy measure monitors the general motion-related impacts on the vehicle, the force measure captures braking-related behavior.

The goal of this study is not only to identify risky driving patterns at intersections, but more importantly to predict a vehicle's level of risk based on its real-time trajectory while the vehicle approaches the intersection stop line. As such, I form the feature space by concatenating several seconds of vehicles' information before they reach the intersection. Stacking recent trajectory data provides the model with the knowledge of the history behavior of the driver, which can be helpful in extrapolating their future behavior. With a 10Hz frequency of data transmission, which is the original setting of the AACVTE environment, the trajectory sequence is formulated as in Eq. (2.1).

$$\begin{aligned} f =& [(v_1, a_1, \Delta v_1^2, \Delta a_1, d_1), (v_2, a_2, \Delta v_2^2, \Delta a_2, d_2), ..., \\ & (v_{99}, a_{99}, \Delta v_{99}^2, \Delta a_{99}, d_{99})]^T \end{aligned} \tag{2.1}$$

2.3.2 Time Series K-means

Unlike existing methods in the risky driving literature that use manual labeling to prepare training sets, in this chapter I adopt an unsupervised learning method to cluster vehicle trajectories into a number of driving patterns. The adopted clustering method is consistent with the feature vector constructed in the previous section, which is a time series sequence.

T. Liao (Liao (2005)) provides a survey on time series clustering methods, concluding that k-means-based methods provide good performance when clustering time series data. Hence, based on the survey results, in this study I use time-series k-means to cluster multi-dimensional vehicle trajectories. Time series k-means is an extension of the k-means algorithm, with proven high performance in learning from time series data. Given a set of observations $(x_1,x_2, x_3, ..., x_n)$ and k clusters, the objective of the k-means algorithm is to find a partitioning of data so as to minimize the sum of the intra-cluster variances. Based on the well-known Loyd's algorithm, k-means starts by randomly selecting k objects as cluster centers (centroids). Then, in an iterative process, the following two steps are repeated: (*i*) each data point x_j in the dataset is assigned to its nearest centroid based on a distance function $dist(c_i, x_j)$, where $c_i \in C, i = 1, ..., k$, and C is the set including all cluster centroid; (*ii*) each cluster centroid is updated based on the objects in the cluster, typically by taking the mean: $c_i = \sum_{j=1}^{\mu_i} x_j / \mu_i$, where μ_i is the size of cluster i. In its basic form, the k-means algorithm uses a Euclidean distance function, guaranteeing that it converges in a finite number of iterations. The convergence criterion is reached when the cluster centers do not change in consecutive iterations.

The number of clusters in k-means is an input parameter and cannot be endogenously determined by the algorithm. Therefore, in this chapter I use the elbow method to heuristically determine the number of clusters. The elbow method determines the number of clusters based on the elbow score, which is computed as the sum of the intra-cluster variances. It is easy to see that a higher number of clusters will result in a smaller elbow score. The elbow method selects the number of clusters at which the rate of change of the elbow score becomes smaller than a threshold (Ketchen and Shook (1996)).

Under this setting, by treating each time series feature sequence as a high-dimensional data point, time series clustering can be implemented using the k-means algorithm. Unlike traditional k-means in which each point consists of a vector of independent features, in time series k-means each training point is a vector of dependent features. As such, the traditional Euclidean distance is not the best measure for quantifying the similarity between points. I introduce the DTW similarity metric as the measure to compute the distance between two time series sequences.

2.3.2.1 Dynamic Time Warping (DTW)

When samples to be clustered are time series sequences, the Euclidean distance measure attempts to find the similarity between two points based on the correspondence between their time indices. Therefore, the Euclidean matching of two time series sequences can fail to identify the similarity of two trajectories with identical patterns, but a small shift in the time domain. DTW is a similarity measure that resolves this issue by asynchronously mapping two time series sequences. In the problem of interest, if two vehicles have the exact same multi-dimensional trajectory but one starts preparing for taking action a few tenth-of-a-seconds later than the other, the Euclidean distance between them could be as high as that computed for two completely different trajectories. DTW can capture temporal distortions in time series sequences by finding the optimal alignment path between sequence points (Berndt and Clifford (1994); Petitjean and Gançarski (2012);

Taylor et al. (2015)). Other advantages of DTW include being able to compute similarities between two trajectories of different lengths, and being resilient to missing data points. Many DTW computation methods are provided in the existing literature. In this chapter, I use a widely adopted symmetric DTW computation method (Sakoe and Chiba (1978)). Given two time series sequences $S = (s_1, s_2, s_3, \cdots, s_n)$ and $M = (m_1, m_2, m_3, \cdots, m_k)$, DTW computes the Euclidean distance between S and M, and finds an optimal alignment path such that the Euclidean distance between the aligned time series is minimal. Denote the optimal warping trajectory between S and M as $\pi^* = (w_1^*, w_2^*, ..., w_Q^*)$. An alignment path π satisfies the below constraints:

- for a position $w_q = (i_q, j_q)$ in π, $0 \leq i_q \leq n, 0 \leq j_q \leq k$
- w_q is related to w_{q-1} as follows: $i_{q-1} \leq i_q \leq i_{q-1} + 1, j_{q-1} \leq j_q \leq j_{q-1} + 1$

The optimal alignment, π^*, and its corresponding optimal DTW similarity measure, can be obtained through Eq. (2.2), subject to satisfying the two previously mentioned constraints.

$$d_{dtw} = min_{\pi} \sqrt{\sum_{(i,j) \in \pi} \|S_i - M_j\|^2} \tag{2.2}$$

The obtained optimal warping path π^* is a temporal alignment of S and M, and provides a symmetric similarity measure regardless of the size of S and M.

Since the DTW algorithm provides the optimal warping trajectory, and following Lloy's algorithm, it is easy to show that the sum of intra-cluster variances decreases with iterations of k-means, guaranteeing that the DTW-based k-means algorithm will converge in a finite number of steps.

DTW-based clustering serves as a cornerstone of the proposed framework. Using DTW-based clustering, the framework identifies different driving patterns from the raw trajectory data. The risk probability for each driving pattern is then computed based on an anomaly detection method, which will be introduced in following sections. Once the model is fully trained and the probabilistic risk score of each cluster is obtained, the DTW-based clustering results serve as a template that can be leveraged during the online assessment (prediction) process.

2.3.3 iForest

With a multi-dimensional feature vector and using a DTW-based time series k-means, I obtain clusters that represent different context-aware driving patterns. Risky driving trajectories can be modeled as anomalies because the majority behaves "normal" (Ellison et al. (1995)). Each class of driving pattern can include anomalous points, which I consider to be risky trajectories. As such, I use iForest to identify risky trajectories in the dataset. iForest is an unsupervised learning algorithm based on decision trees, and offers an efficient way for outlier detection in high-dimensional feature spaces (Liu et al. (2012)). Unlike popular outlier detection methods, e.g., elliptic envelope (Rousseeuw and Driessen (1999)) and one-class SVM (Perdisci et al. (2006)), which profile normal data points or can be very sensitive to outliers (Amer et al. (2013)), iForest explicitly identifies anomalies by taking full advantage of the anomalies' properties of few and different. An iForest uses several isolation trees to detect anomalies. An isolation tree is constructed as a proper binary tree, which has either zero or exactly two child nodes. When constructing, given a sample X from the data with n instances, X is divided recursively by randomly selecting an attribute and a split value until the tree reaches a height limit, or all data points in X have the same value, i.e., all instances are isolated (Liu et al. (2008)). The 'anomaly score' for an iForest, denoted as $\kappa(x, \phi)$ can be computed as in Eq. (2.3)

$$\kappa(x, \phi) = 2^{-\frac{\mathbb{E}(h(x))}{nc(\phi)}} \tag{2.3}$$

where x is a data point, ϕ is the sub-sampling size, $h(x)$ is the the depth of the tree, and $c(\phi)$ is the average of $h(x)$ given ϕ, n is the number of fitted estimator during the process. $\mathbb{E}(h(x))$ is the average depth of the isolation trees that form the forest. The lower the score, the more abnormal a sample is. Note that the anomaly score penalizes the higher number of partitions for the normal class.

The iForest has been successfully used in detecting anomalies in stream data (Ding and Fei (2013)) and to detect anomalous user behavior (Sun et al. (2016)). I train an iForest on the history trajectory data collected by an RSU at the intersection of interest. The iForest finds outlier trajectories, which are assumed to be risky driving trajectories, given the selected feature space. In the next section, I discuss how the trained iForest can be used to generate RDP scores for clusters of driving patterns.

2.3.4 Risk Probability

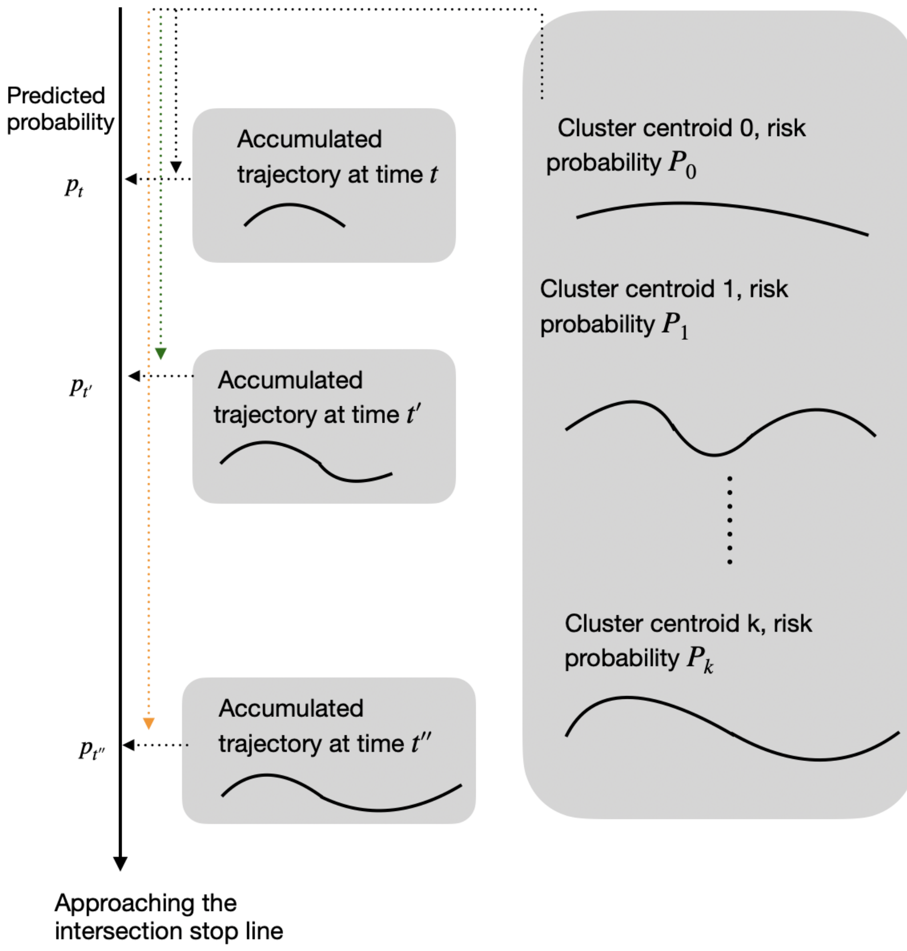
Survey-based studies typically devise risk and aggressiveness scales based on the frequency at which the survey participants report manifesting risky and aggressive driving behaviors. Similarly, I adopt a frequency-based method to compute risk scores of clusters that correspond to driving patterns.

Using a DTW-based time series k-means, I cluster trajectories into different classes based on a risk-informed features space. To quantify the risk level in each cluster, I use the trained iForest to identify the risky trajectories in each cluster. By computing the percentage of risky trajectories in each cluster, I construct a quantitative measure of risk for each driving pattern, as $P_i = T_i^{'}/T_i$, where P_i is the risk probability of cluster i, also known as the RDP score, $T_i^{'}$ is the number of trajectories identified as risky by the iForest in cluster i, and T_i is the total number of trajectories in cluster i.

2.3.5 Real-time Assessment

The goal of this chapter is to predict risky driving before vehicles arrive at an intersection stop line so as to inform nearby drivers and traffic engineers to take appropriate actions. As a driver approaches an intersection, more of its trajectory becomes revealed. In the real time assessment step, the framework computes the DTW similarity measure between the revealed portion of the trajectory and cluster centers to find the cluster that best represents the trajectory. The trajectory then adopts the RDP score of this cluster.

The real time assessment process is illustrated in Figure 2.1, where P_k is the RDP score of cluster k, and $p_t, p_{t'}, p_{t''} \in \{P_0, P_1, \cdots, P_k\}$ are the RDP scores of the approaching vehicle at times t, t' and t'', respectively. Once a vehicle enters the CV communication range of the RSU located the intersection, the framework is activated. At each updating step, the DTW similarity between the so-far revealed trajectory of the vehicle and all cluster representatives are computed. The cluster representative that provides the closest



Predicted probability
p_t
$p_{t'}$
$p_{t''}$
Accumulated trajectory at time t
Accumulated trajectory at time t'
Accumulated trajectory at time t''
Cluster centroid 0, risk probability P_0
Cluster centroid 1, risk probability P_1
Cluster centroid k, risk probability P_k
Approaching the intersection stop line

Figure 2.1: Real-time forecasting of driving risk. The trajectory adopts the RDP score of the cluster to which it is assigned.

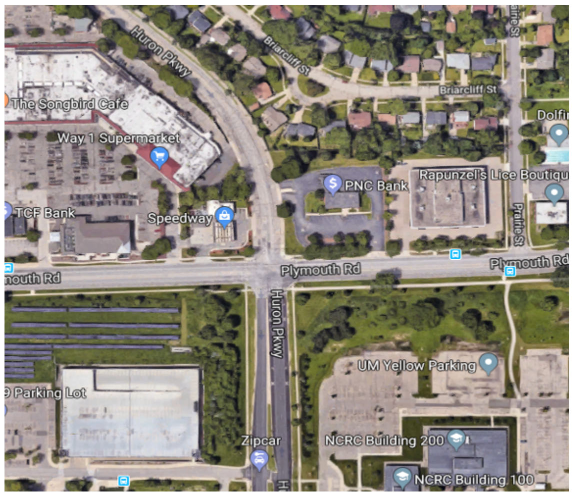
DTW similarity measure is selected as the cluster that best represents the trajectory, and the RDP score of that cluster is adopted as the RDP score of the vehicle. This flow of information is depicted in Figure 2.1 in the form of dashed lines. Assuming that each cluster representative is a multi-dimensional sequence with length n, then for an incoming vehicle the computational time complexity of real time assessment is $O(kn^2)$, in which the complexity of DTW similarity measure is $O(n^2)$. The computation can be further boosted through GPU-based parallel DTW computation (Xiao et al. (2013)).

2.4 Experiments

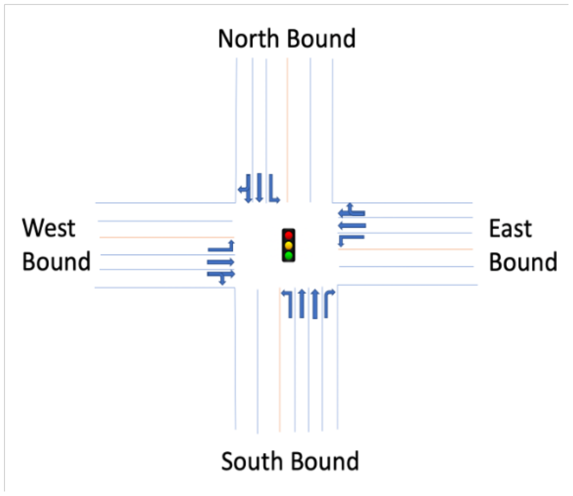
The CV data for the work is collected by AACVTE (Woodhouse (2014)). Connected vehicles in AACVTE are able to communicate with other surrounding connected vehicles and with RSUs. When a CV is driving around an intersection equipped with an RSU, it continuously sends real-time BSMs to the RSU. The RSU then forwards the BSMs to the central server. As the dataset can include a mix of time-stamped messages reported by many vehicles, I extract vehicle-level information based on unique vehicle IDs. A part of such vehicle-level information is a time-series sequence. Each BSM includes information on vehicle ID, its basic motion information including location, speed, and acceleration, its nearby RSU ID, etc. Based on this information, I construct the time-series feature sequences using Eq.(2.1).

I investigate the performance of the framework using the CV data collected at the Huron parkway and Plymouth road intersection, located in Ann Arbor, MI, United States. The subject intersection is shown in Figure 2.2.

The intersection under study is a fully signalized 4-way intersection with protected left turns. There is a right-turn only lane at the south bound of the intersection, while all other three approaches have three lanes: one for left-turn, one for through movement, and one for through movement and right-turn. The speed limits of the intersection north/south bound and east/north bound lanes are 35 mph and 45 mph, respectively. Raw data generated by CVs can contain trajectories that are not driving on the road. For example, if a CV is driving in a plaza near the RSU, its trajectory will be collected by the RSU. Therefore pre-processing is needed to exclude those trajectories that are not using the intersection.



(a) Satellite image of the subject intersection



(b) Conceptual diagram of the subject intersection

Figure 2.2: The subject intersection located at the Huron parkway and Plymouth road, in Ann Arbor, MI

2.4.1 Pre-processing Process

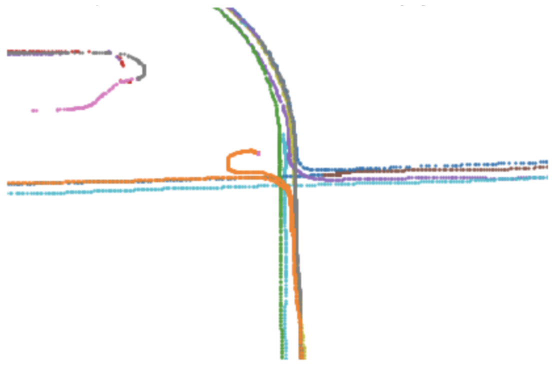
I start the experiments by pre-processing the raw data following the two steps laid out below:

Step 1: For each trajectory in the dataset, find its nearest point to the center of the intersection. If this distance is greater than the midpoint of the diagonal end points of the intersection, the trajectory is assumed not to be passing through the intersection, and is excluded from the dataset. The threshold is selected based on the structure of the transportation network at the intersection.

Step 2: For each trip, extract the trajectory that is farther than 10 seconds away from the intersection. These trajectories will be utilized to construct the features. Generally, the selected trajectories should be as long as possible so as to allow time for mitigatory actions in case of a risky environment. However, selecting a longer feature profile would result in trajectories that do not pass though the intersection, or whose behavior is not impacted by the presence of the intersection yet, reducing the effectiveness of the trained model.

The result of the pre-processing is demonstrated in Figure 2.3. As this figure shows, this process helps exclude the trajectories that do not pass through the intersection.

After pre-processing, time series profiles of the main features are illustrated in Figure 2.4. The processed dataset includes trajectories from CVs during a one-month period. For


(a) Raw CV trajectories

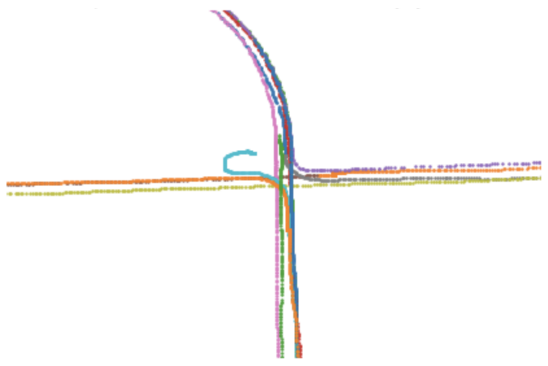
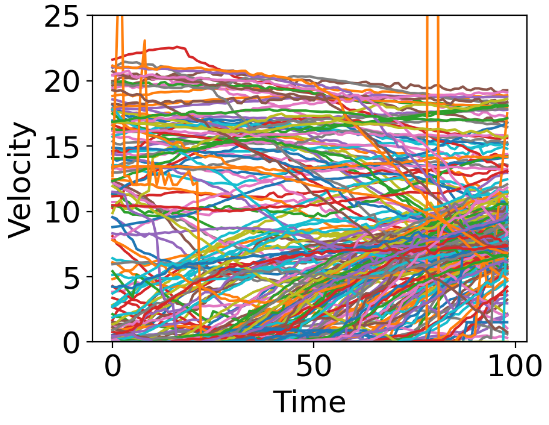

(b) Pre-processed trajectories

Figure 2.3: (a) raw and (b) pre-processed data

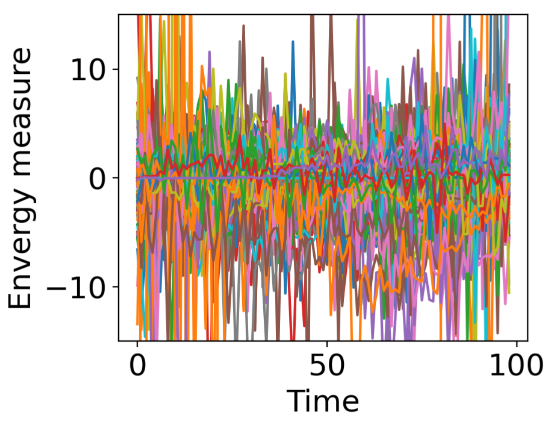
visualization purposes, Figure 2.4 only illustrates trajectories generated by 40 randomly sampled CVs. This figure serves as a benchmark against which the performance of the clustering algorithm can be judged.

2.4.2 Time Series K-means

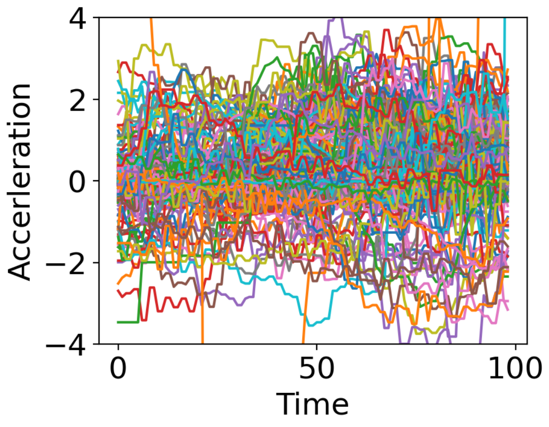
In this section, I demonstrate the driving patterns generated in the target intersection using the time series k-means algorithm. Since the time series k-means algorithm is a locally optimal algorithm, I run this algorithm multiple times, each time using a randomly selected set of points as cluster centers, and use the best obtained clustering, i.e., the one with the smallest total intra-cluster distances, as the final solution. The elbow method, used in conjunction with the time series k-means algorithm suggests 3 to 8 number of clusters for the intersection of interest. The clustering results for the case of 6 clusters are shown in Figure 2.6 with the visualization of motion-related features. Figure 2.6 shows that vehicle trajectories are well-clustered compared to the raw data in Figure 2.4. Based the clustering results, the proportion of trajectories in clusters are 16.9%, 7.8%, 17.4%, 17.9%, 27.7%, and 12.3% from clusters 0 to 5, respectively. This figure demonstrates that trajectories in each cluster share similar feature profiles, and different clusters showcase different patterns, providing a visual confirmation that different clusters correspond to different driving patterns. To quantify the clustering performance, the average inter- and intra-cluster distances are computed and demonstrated in Table 2.2. The diagonal terms



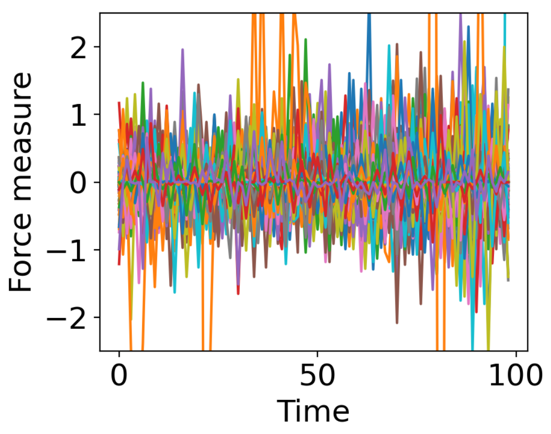
(a) Velocity feature



(b) Energy measure



(c) Acceleration feature



(d) Force measure

Figure 2.4: Feature illustration of raw trajectory data

in this table show the intra-cluster distances, computed by averaging the DTW similarity measures between each cluster member and its cluster representative. Each off-diagonal value is the inter-cluster distance between its corresponding row and column clusters, and is computed as the DTW similarity measure between the cluster representatives of the row and column clusters. As demonstrated by this table, for each cluster inter-cluster distances are overwhelmingly greater than the intra-cluster distance, indicating that trajectories with similar patterns are clustered together, while trajectories with different patterns are assigned to different clusters.

		To cluster					
		0	1	2	3	4	5
From cluster	0	**3.32**	9.72	6.97	4.83	8,65	4.30
	1	9.72	**3.12**	4.90	8.43	3.13	7.03
	2	6.97	4.90	**3.72**	3.46	1.74	5.64
	3	4.83	8.43	3.46	**3.82**	5.45	6.48
	4	8.65	3.13	1.74	5.45	**3.95**	5.69
	5	4.30	7.03	5.64	6.48	5.69	**4.23**

Table 2.2: Inter- and intra-cluster distances measuring the similarity between and within clusters, respectively

One observation from Table 2.2 is that the intra-cluster distances are not always lower than inter-cluster distances. This occurs because different clusters may contain cross behaviors. Note that cross behavior is an intrinsic part of clustering. There is typically a trade-off between increasing the number of clusters so as to reduce cross behavior, and accepting some degree of cross behavior in favor of more meaningful clusters. Furthermore, the impact of the cross-cluster behavior in the general framework is mitigated due to the fact that clustering is only used to construct driving patterns, whose boundaries are not exclusive by nature, and the risky trajectory detection step is independent of the clustering step.

2.4.3 iForest

To systematically mark risky trajectories in each cluster, I train an iForest model on the full dataset with different contamination rates. I selected contamination rate of 0.18 in the the experiment as it results in a wide range of RDP scores for clusters, with one cluster

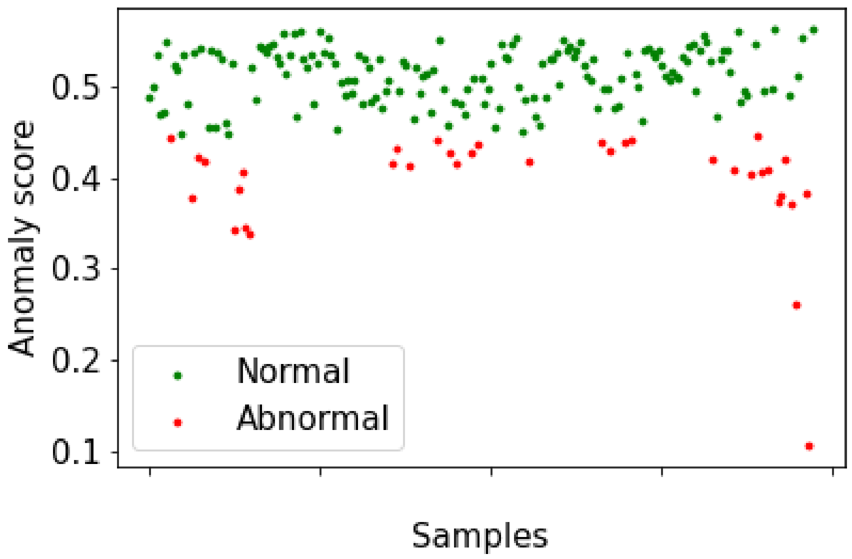
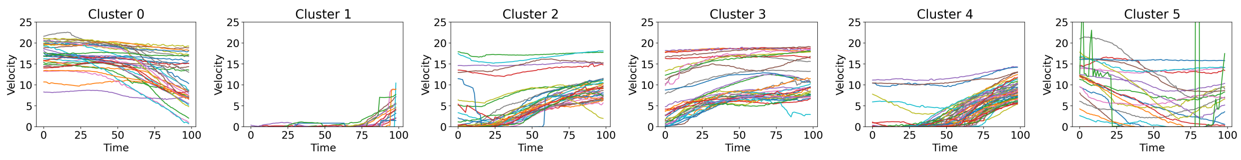


Figure 2.5: Anomaly scores of training samples based on the trained iForest model

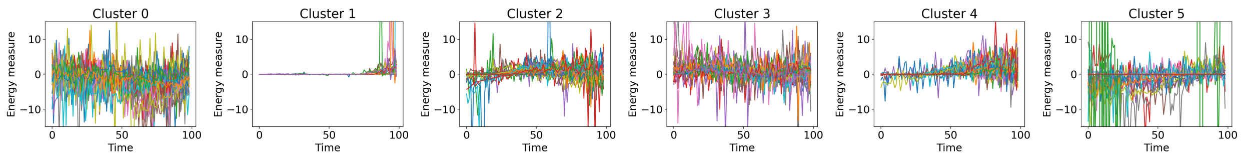
achieving a zero RDP score, thereby proving a bolder distinction between clusters as it relates to risky driving. This value is based on the general driving attitude in the region, and could vary depending on the application and the tolerance for abnormal behavior. Results are demonstrated in Figure 2.5. This figure demonstrates the anomaly scores of the data points as generated by the iForest, and quantified using Eq. (2.3). Anomalous trajectories are generally easier to isolate, leading to shallower trees to correctly identify them. The average score for anomalous trajectories is the study is 0.4, while that of normal trajectories is 0.51. The iForest model can successfully identify risky driving behavior, as trajectories with lower risk typically demonstrate fewer unexpected behaviors, such as dramatic speed and acceleration change, while trajectories that contain high jerk values differ substantially from the rest of the trajectories due to sudden changes in their profile.

2.4.4 Overall Framework

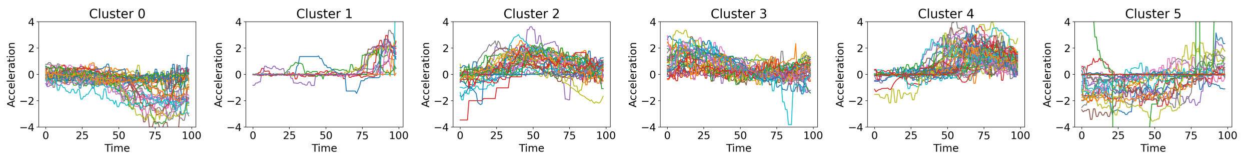
To further evaluate the overall performance of the proposed framework, Figure 2.7 demonstrates the mean values and 95% confidence intervals of all features for both normal and anomalous trajectories in each cluster. In Figure 2.7, solid lines represent mean values, and blue and gray regions mark the 95% confidence intervals of risky and normal



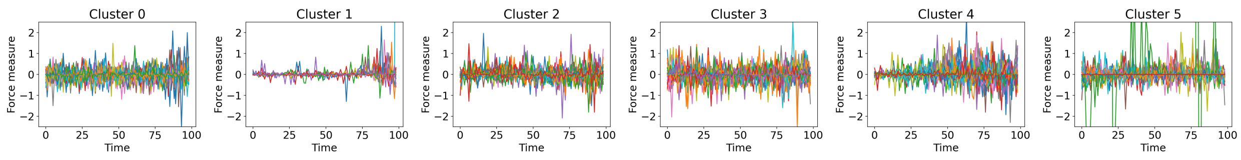
(a) Clustered feature illustration-velocity



(b) Clustered feature illustration-energy measure



(c) Clustered feature illustration-acceleration measure

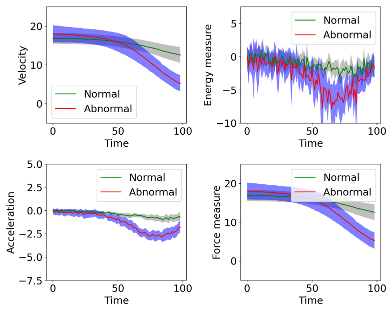


(d) Clustered feature illustration-force measure

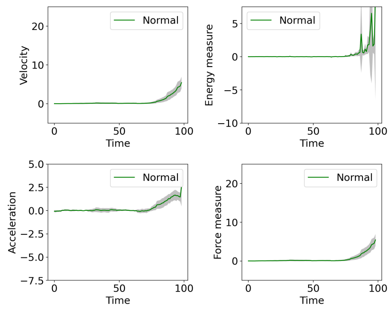
Figure 2.6: DTW-based time series k-means clustering results

trajectories, respectively.

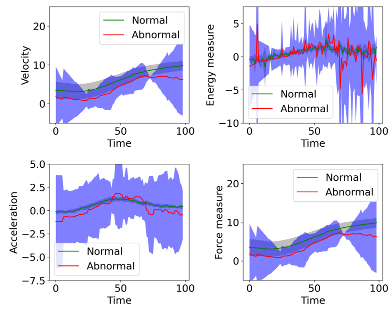
As shown in Figure 2.7, anomalous trajectories typically correspond with much wider confidence intervals, regardless of the feature. Anomalous trajectories also demonstrate a higher level of uncertainty in driving behavior as identified by frequent changes in feature mean values. These characteristics correspond to more unexpected behaviors and therefore higher risk levels. The RDP score of each cluster, defined as the percentage of anomalous trajectories in the cluster, can be interpreted as the probability of a trajectory assigned to the cluster demonstrating anomalous behavior. The RDP scores range from 0.0 to 0.412, which indicate the corresponding risk level of each cluster. As can be noticed in Figure 2.7, within a cluster, the confidence ranges of normal and anomalous trajectories may overlap. This overlap is due to the fact that normal and anomalous trajectories may have similar patterns for several time segments. However, note that the unit of modeling here



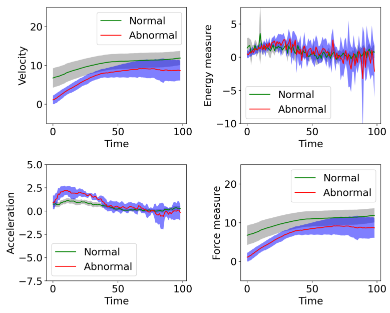
(a) Cluster 0



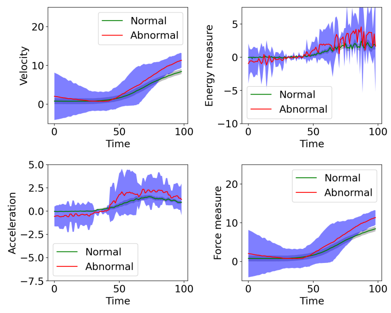
(b) Cluster 1



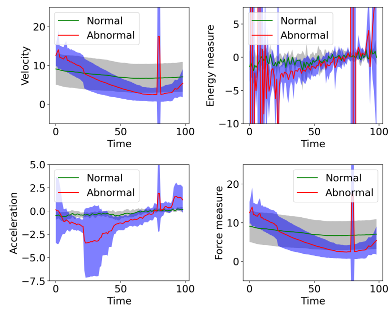
(c) Cluster 2, RDP score of 0.088.



(d) Cluster 3



(e) Cluster 4



(f) Cluster 5

Figure 2.7: Features and their 95% confidence intervals for normal and anomalous trajectories in clusters. The gray and blue regions show the 95% confidence intervals of normal and anomalous trajectories, respectively.

is a full trajectory. Therefore, the cluster robustness is determined by confidence ranges of full trajectories and is not affected by different clusters sharing small portions of their trajectories, as demonstrated in Table 2.2. In addition, the confidence ranges of anomalous trajectories are typically much wider than those of normal trajectories, demonstrating the difference between normal and risky trajectories. Figures 2.6 and 2.7 provide a validation of the premise of this study that, based on the proposed method, different clusters represent different driving patterns, and different driving patterns are associated with different levels of risk.

By investigating feature profiles of the clusters, as indicated in Figure 2.6 and Figure 2.7, rough interpretations can be provided for each driving pattern. Cluster 0 represents a “decelerating driving pattern” according to its acceleration profile. Note that the full feature space should be used for such interpretations, and the acceleration profile here is used only for demonstration purposes. Cluster 1 signifies “waiting and then start moving”, since the feature profiles show that the majority of trajectories are initially stopped, and then move at very low speeds, together with changes in energy, acceleration and force measures. It is likely that such a feature profile results from a change in the traffic signal light from red to green. By examining the feature profile of cluster 2, the driving pattern represented by this cluster can be explained as “safe acceleration”, because the feature profiles show that the trajectories are accelerating within a narrow acceleration range, and the changes detected in energy and force profiles are not drastic with a low RDP score. Similarly, I can explain cluster 3 as “driving with low acceleration”, cluster 4 as “accelerating after a short waiting period”, and cluster 5 as “driving under uncertainty”, as its feature profiles showcase several spikes and drastic changes. Note, however, that the above interpretations are given using only the limited set of features introduced here. Access to further information such as signal phase and timing (SPaT) or the status of the surrounding vehicles may enable more accurate, context-aware interpretation.

The iForest requires a pre-defined contamination rate, i.e., the percentage of anomalous trajectories in the dataset. Here, I conduct sensitivity analysis on the contamination rate and report the results in Figure 2.8. In this figure, a wide range, 0.10 to 0.45, of contamination

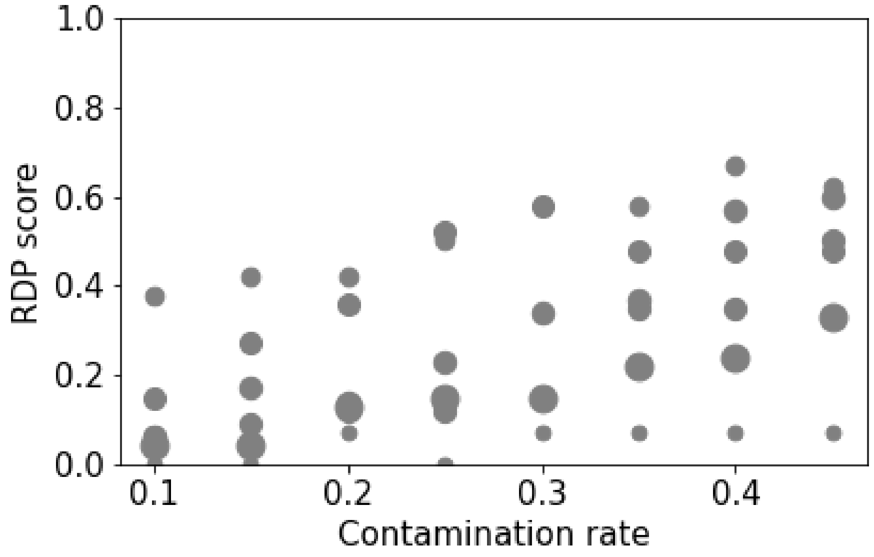


Figure 2.8: Contamination rate sensitivity analysis

rates are applied to the six clusters. For a given contamination rate, each dot represents a cluster, where the RDP score of the cluster is marked on the vertical axis, and the size of the dot represents the proportion of trajectories in it. As Figure 2.8 demonstrates, the contamination rate affects the range of the RDP scores. This hyper-parameter may be fine-tuned for different application scenarios. If a larger RDP score range is preferred, a larger contamination rate could be adopted.

Results from the RDP framework can be used by transportation management agencies to obtain structural or operational intersection safety profiles, which can be used for improving intersection safety. For instance, if for a subject intersection most trajectories are located in clusters that have low RDP scores, then it indicates that the subject intersection is in generally safe and the existing intersection design is likely to be appropriate. On the other hand, if an intersection located in the same region is hosting a large number of highly risky trajectories, this implies that there can be structural or operational issues with the intersection, e.g., blind spots, narrow lanes, or sub-optimal phasing and timing.

The results of real time assessment and prediction of 10 sample trajectories at the intersection under study are shown in Figure 2.9. These trajectories are deliberately selected to showcase different driving patterns. In this figure, I re-assess the clustering assignments of vehicles every second. In Figure 2.9, when vehicle 5 enters the RSU detection region of the intersection, it is assigned an RDP score of of 0.412. As it approaches the intersection, its risk is reduced substantially, to the point of reaching an RDP score of 0.171 at the intersection. As another example, the RDP score of vehicle 6 does not change throughout the interval this vehicle approaches the intersection, and remains at the high value of 0.273. It is worth noting that the RDP scores of all vehicles remain constant just before they reach the intersection. This is because as vehicles approach the intersection, more of their trajectory profiles are accumulated, enabling the model to produce more robust predictions.

At the vehicle level, RDP scores can be used to set the appropriate mitigation strategies when vehicles are still farther away from the intersection, such as issuing safety alerts to drivers who have high RDP scores, or for behavior-based insurance policies. Similarly, a safety-focused intersection-monitoring system can also benefit from the proposed framework by broadcasting warnings to other vehicles at the intersection, or taking other actions for passive safety, e.g., adjusting the traffic signal light timing to avoid crashes. The

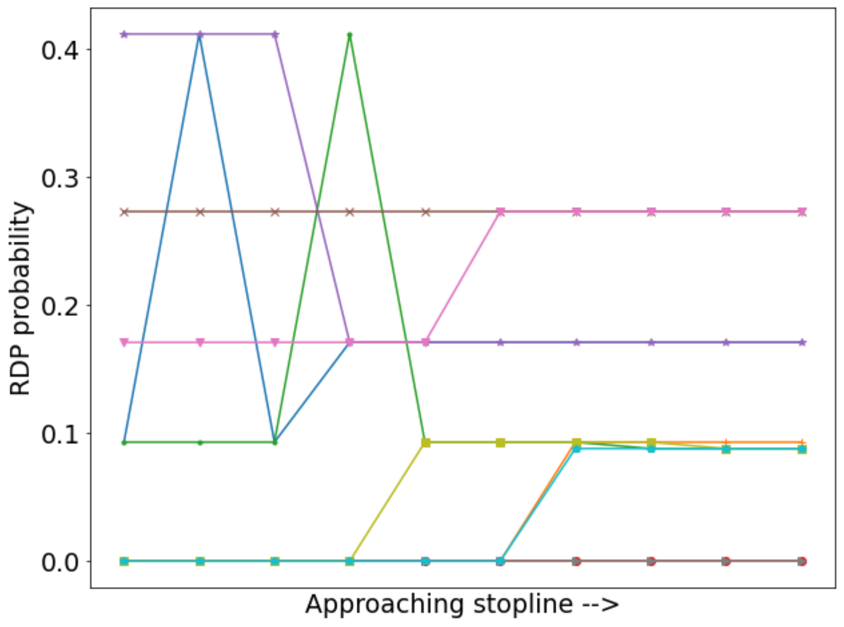


Figure 2.9: Real time assessment of incoming trajectories

total training time for a training set of size 10,000 is 608 seconds on a personal computer with 4 Intel-i7 CPU cores, using a parallel training procedure. During the prediction process, with the pre-trained model, it takes 2.5-4.0 ms to provide the assessment result. In real-world deployments, the RDP model can be trained offline, and the pre-trained models can be used for prediction purposes. Considering the general BSM frequency, which is 10Hz (receiving a new message every 100ms) in the experiments, the computation time is suitable for real-time applications. The CV environment is readily applied to single intersections, corridors, or regions to ascertain risk and safety and facilitate insurers to implement dynamic insurance policies as well as enable traffic engineers to reduce crash rates by traffic signal cycle lengths.

2.5 Conclusion

In this chapter I propose a scalable framework to detect and predict risky driving in a CV environment. The proposed framework is based on unsupervised learning, and uses a multi-dimensional time series feature space, constructed based on connected vehicle trajectories, to assign risk scores to vehicles approaching a signalized intersection. Unlike existing studies, the proposed framework does not require a labeled dataset, making it scalable and easy to implement, especially since risky driving patterns could vary by location and driving environment. Additionally, avoiding manual labeling and/or using survey-based datasets eliminates the possibility of introducing subjectivity into the model. I implement the framework using vehicle trajectories collected through the AACVTE, for an intersection located in the city of Ann Arbor, Michigan. Results validate the existence of different context-aware driving patterns among drivers using this intersection. I demonstrate how this methodology can be used to dynamically update the risk score of a vehicle approaching the intersection. Furthermore, I discuss how this framework can be used to construct intersection safety profiles and safety warning systems.

The RDP model can be used by a number of stakeholders to enhance safety at intersections. It can be used by city traffic engineers to implement real-time control strategies in response to risky driving scenarios, as informed by driving patterns. In addition to real-time control measures, cities can identify intersections with lower safety levels, which correspond to consistently high risk scores across drivers, and attempt to improve the general safety of the intersection by, for example, improving the intersection geometry, optimizing SPaT, etc. Additionally, RSUs can send warning messages to risky drivers with recommendations on how to improve their driving behavior, or issue risk warnings to all road users to avoid potential accidents. Insurance companies can use this driver-level risk-based analysis to provide premiums based on a region rather than assessing premiums individually, thus offering competitive cost insurance rates. Finally, this information can be used by law enforcement agencies to curb risky driving behavior of individual drivers.

The proposed framework can be easily transferred to different types of intersections. When extending the framework to different intersections, intersection-specific models may be trained. Users may adopt different model parameter settings to customize the model for different intersections. Examples include using different cluster numbers to capture the range of driving patterns/contexts at different intersections and different contamination rates to model the general level of risky behavior at the intersection of interest. Additionally, trajectories collected from an intersection may capture intersection-specific characteristics, including the influence of the geometry of the intersection, existing SPaT control measures, etc. As such, training intersection-specific models will allow users to capture such idiosyncrasies.

CHAPTER III

Step Attention: Sequential Pedestrian Trajectory Prediction

3.1 Introduction

Recent studies on AV technology have demonstrated its potential in improving efficiency in transportation systems (Fagnant and Kockelman (2015); Hult et al. (2016); Bagloee et al. (2016)). At the same time, safety concerns surrounding autonomous vehicles are gaining more attention, partly due to several publicly known incidents involving AVs (Favarò et al. (2017); Bissell (2018)). Different from traditional vehicles, AVs are equipped with more sensors to capture the surrounding environment, such as dynamics of other surrounding traffic agents. AVs are envisioned to share the roadway network with legacy vehicles as well as other road users. Consequently, one important aspect of autonomous driving is for the autonomous entity to predict the future trajectories of its surrounding agents, and plan its motion accordingly (Rudenko et al. (2020)). In addition to alleviating safety concerns, high-quality trajectory prediction of other agents can enhance model predictive control (MPC) performance of AVs (Kim et al. (2001); Liu et al. (2017); Almasri et al. (2016)).

This chapter focuses of trajectory prediction for the most vulnerable road users: the pedestrians. Due to the advancements in sensor technologies, real-time pedestrian location information can be obtained easily. Pedestrians' movements can be perceived by cameras, localization units, or other device in real time (Tian et al. (2019); Yu et al. (2019); Etinger et al. (2013); Kim et al. (2015)). Given a perceived trajectory of a pedestrian, a timely research direction is to develop methods that produce high-quality predictions of future pedestrian movements.

Pedestrian trajectory prediction is becoming a critically important topic of research as we move toward the next generation of transportation systems in which pedestrians interact with AVs, creating safety concerns (Jaipuria et al. (2018); Møgelmose et al. (2015); Ma et al. (2019); Chandra et al. (2019)). AVs require precise prediction of their dynamic surrounding environment, which includes pedestrians, for long look-ahead planning. Such predictions will allow AVs to plan paths that have safety guarantees, and safely engage in other related driving tasks (Hussein et al. (2016); Saleh et al. (2017)). This could ensure the safety of the AV occupants, pedestrians, and other surrounding road users. However, the complex nature of the walking behavior makes long look-ahead walking trajectory prediction a challenging problem. In general, pedestrians move slower than vehicles, but their motion change can be rapid due to the complex nature of human behavior. For an adult, the length of a step ranges from 0.71 m (walking) to 1.49 m (running), taking approximately 0.33 s to 0.53 s to take a step (Hoeger et al. (2008)). Additionally, a pedestrian's walking behavior can be subjective, depending on individual characteristics (e.g., age, gender), their goal of walking, and their dynamically changing surrounding environment (e.g., public environment, road surface, etc.) (Zacharias (2001)).

In this chapter, I propose a deep learning model, which I call *step attention*, to achieve accurate pedestrian trajectory prediction for long look-ahead horizons. In the proposed model, I introduce long-short term memory (LSTM)-based and gated recurrent unit (GRU)-based recurrent neural networks (RNNs) to capture sequence-related hidden features in pedestrian trajectories. A time-distributed kernel is proposed with multiple deep convolutional neural network (CNN) layers and an augmented attention mechanism in order to learn patterns of walking behavior.

Unlike the existing deep learning approaches, step attention does not explicitly model environmental factors such as social interactions, pedestrian density in the under-study region, road surface conditions, weather, etc. Not only can it be unrealistic to collect real-time information on all these factors, but also incorporating all such features creates large-scale input spaces on which learning generalizable models may prove to be impractical (Rasouli and Tsotsos (2019)). On the contrary, step attention captures these factors by means of learning patterns on revealed pedestrian trajectories. As such, step attention can be implemented in any generic environment, and does not require a certain density of other agents or network conditions to perform well.

I showcase the performance of step attention by conducting experiments on the well-known and diverse benchmark dataset TrajNet (Sadeghian et al. (2018)), as well as real-world pedestrian trajectories collected through infrastructure-based roadside sensors and portable devices. I show that owing to its generic model architecture, step attention provides accurate predictions and outperforms state-of-the-art methods based on two commonly used evaluation metrics, i.e., the average displacement error (ADE) and the final displacement error (FDE) along the prediction horizon. The proposed model also shows good performance when generalizing to driving environments, specifically intersections, for long look-ahead predictions, with an approximately linear prediction error propagation trend within the prediction horizon.

The main contributions of this chapter are as follows: i) A deep learning-based model is proposed for long horizon pedestrian trajectory prediction. Instead of utilizing as much information as possible, e.g., surrounding traffic status, road surface condition, weather condition, etc., the model uses only history trajectory of subject pedestrians, reducing the information requirements compared to other learning based methods. Less information requirement makes the model more adaptive to different systems, as history trajectory is one of the the most basic information and can be easily accessed in different environments; ii) A novel architecture is designed and developed within the model, including an LSTM-RNN kernel, a time-distributed kernel, and a GRU-RNN kernel, to learn different patterns. In the time-distributed kernel, augmented attention is introduced as an improvement strategy to mitigate the drawbacks of CNNs and to learn global information from its previous layers; iii) The proposed model uses a sequential prediction strategy, which allows users to select the prediction horizon freely based on their need and/or tolerance to prediction error; iv) The proposed model is rigorously tested on three different datasets that are collected using different sensor systems: (1) several benchmark datasets collected using drones and top-down view cameras in various crowded public spaces, (2) a proprietary dataset collected using roadside cameras at an intersection, and (3) A complementary dataset collected using portable devices at a secondary intersection location. v) The proposed step attention model outperforms existing state-of-the-art models and obtains high-quality prediction, with ADE/FDE of 0.53/1.72 m for a 4.8-seconds-long prediction horizon on the benchmark datasets. It also performs well in predicting pedestrian trajectories at intersections, obtaining an ADE/FDE of 0.74/1.40 m for a 6-seconds-long prediction horizon on a proprietary intersection dataset, and ADE/FDE of 0.76/1.70 m for a 6-seconds-long prediction horizon on a complementary intersection dataset. The performance of the model is visually demonstrated under different scenarios such as stopping, walking in curvy pattern, walking straight, turning, etc. on the complementary dataset.

3.2 Related Work

In this section I review the existing work on pedestrian trajectory prediction, with special emphasis on a type of deep learning architecture called the attention mechanism.

3.2.1 Pedestrian Trajectory Prediction Methods

In the past several years, a large number of methods and algorithms for pedestrian trajectory prediction has emerged due to its significance in achieving a safe autonomous driving environment, among other applications (Rudenko et al. (2020); Xu et al. (2018); Zhang et al. (2019); Alahi et al. (2016); Xue et al. (2019); Sun et al. (2018)). Existing work on this topic can be mainly summarized into three categories: (i) physics-based models; (ii) planning-based models; and (iii) pattern-based models (Rudenko et al. (2020)). The rest of this section is dedicated to the literature on these three categories.

3.2.1.1 Physics-based Models

Physics-based models observe a pedestrian's motion properties, such as location, speed, etc., and use laws of physics to predict its future movements. S. Kim et al. used a Kalman filter and machine learning-based approach to predict pedestrian trajectories through velocity-space reasoning (Kim et al. (2015)). This method constructed a motion model to compute the desired velocity of pedestrians, and was shown to offer good performance.

F. Zanlungo et al. used a social force-based model to predict pedestrian locations so as to avoid collisions (Zanlungo et al. (2011)). They modeled walking behaviors following the social force paradigm with physical constraints, and stated that the model achieves high performance in describing pedestrians' trajectory sets. However, due to the nature of the social force model, its performance could be negatively affected when pedestrian density is low. A. Martinelli et al. proposed a method for pedestrian dead reckoning based on step length estimation (Martinelli et al. (2017)). Based on the classified walking behavior, the step length of an individual was estimated, and then the position was inferred. Similary, W. Kang et al. proposed a smartphone-based method for pedestrian position inference (Kang and Han (2014)). The step length estimation-based inference was demonstrated to have good performance in indoor environments, but the inference error accumulated over time as walking distance increased (Hou and Bergmann (2020)). Based on Wi-Fi signal fingerprints and smartphone signals, K. Gao et al. proposed a probabilistic method for indoor position estimation with gyroscope and accelerometer (Gao et al. (2021)). It demonstrated good performance on overcoming signal changes and improved average accuracy.

Most physics-based models rely on manually specified model parameters or rules, which makes their application limited to scenarios such as predicting trajectories in a closed space, where the uncertainly of movement is more limited compared to outdoor scenarios. Unlike these models, the proposed step attention model does not manually specify parameter values; rather, it learns trajectory patterns using history trajectory profiles.

3.2.1.2 Planning-based Models

Planning-based approaches to predict pedestrian trajectories are typically goal-directed. B. Ziebart et al. proposed a planning-based model for pedestrian trajectory prediction with a destination distribution, and formulated a Markov decision process to conduct the planning and prediction procedures (Ziebart et al. (2009)). This model outperformed the generalization ability of the variable-length Markov model, which was a hidden Markov model-based approach capable of predicting 3 s long trajectories (Galata et al. (2001)).

N. Deo et al. proposed a variational Gaussian mixture model (VGMM) to learn position distribution. VGMM clustered trajectories into sub-categories based on estimated sources and destinations, and then trained a model for each sub-category to predict pedestrian trajectories (Deo and Trivedi (2017)). The planning based approach was a probabilistic framework and outperformed monolithic VGMM, i.e., a single model for all sub-categories, of equivalent complexity. E. Rehder et al. adopted deep neural networks to predict pedestrian trajectories through a planning-based approach (Rehder et al. (2018)). This method relied on an inferred mixture density function for possible destinations to conduct goal-directed planning, after which the planner used as a predictor. This work concluded that planning-based approaches perform well for short horizon predictions, but could diverge when conducting long term-horizon predictions. P. Dendorfer et al proposed a deep learning model called goal-GAN for trajectory prediction. It used a two-phase strategy that first estimated the goals, and then generated predicted trajectories by routing from the current position toward the estimated goal (Dendorfer et al. (2020)). Y. Yao et al proposed a method for pedestrian trajectory prediction conditioned on goal estimation (Yao et al. (2021)). It used a bi-directional multi-modal setting to improve the performance of the model. H. Tran et al. proposed a goal-driven method for human trajectory prediction (Tran et al. (2021)). It decomposed the model into two sub-processes: i) goal process; and ii) movement process. By using a goal channel to govern the prediction and a movement channel to control the movement details, the authors claimed the method obtained good performance in long-term trajectory prediction.

Planning-based models always speculate the future goals of a pedestrian. Due to the uncertainty of a pedestrian's purpose, this setting can lower their performance when conducting longer horizon predictions, as demonstrated in some of their findings. In contrast, the proposed model has an open setting, which does not guess or speculate any future goals or destinations, thereby increasing prediction accuracy for longer horizon predictions and improving the generalization ability of the model.

3.2.1.3 Pattern-based Models

Pattern-based models have become mainstream in recent years due to advancements in deep learning. The majority of the existing studies focus on designing modules to learn the interactions and social features between pedestrians, as it is a direct contributor to individuals' movements. A. Alahi et al. proposed a deep learning model called social LSTM to predict human trajectories in crowded spaces (Alahi et al. (2016)). Social LSTM assumed the interactions between pedestrians could be captured with pooling layers in model architecture. It used a social pooling strategy to capture patterns of social interactions, and showed high performance when predicting human trajectories in long look-aheads. Similarly, H. Xue et al. adopted a different scale strategy to capture the neighborhood influence to a subject pedestrian (Xue et al. (2018)). A. Gupta et al. used a novel approach, i.e., social GAN, which was based on generative adversarial networks (GAN) to learn the interaction pattern between pedestrians and predict their trajectories (Gupta et al. (2018)). Social GAN predicted several possible future trajectories, and picked one as its final prediction. The social context was also captured by a shared pooling layer based on its assumption, similar to social LSTM. They demonstrated that the best prediction among all predicted candidates achieves a high accuracy when predicting a 4.8-seconds-long look-ahead time. P. Zhang et al. proposed a state refinement module SR-LSTM for LSTM networks to predict trajectories, and could moderately explain implicit social behaviors between pedestrians (Zhang et al. (2019)). T. Zhao et al. proposed a multi-agent tensor fusion model (MATF) to decode the social and interactive relationships between pedestrians (Zhao et al. (2019)). It aligned spatial encoding of scenes to encoding of each agent in the scene, and trained a GAN model to learn patterns and generate predictions. N. Nikhil et al. adopted a CNN based model that was computationally efficient, allowed fast parallel processing, and achieved competitive performance (Nikhil and Tran Morris (2018)). Y. Huang et al. extended the idea by taking temporal correlation between interactions into consideration and producing more socially plausible trajectories Huang et al. (2019). Y. Xu et al. used a deep neural network-based encoding approach to predict the displacement between pedestrians and therefore trajectories (Xu et al. (2018)).

This method assigned different weights to different pedestrians to model the impact of different social interactions. X. Song et al. proposed a deep convolutional LSTM network to predict human trajectory (Song et al. (2020)). It used tensors to represent environmental features, and designed a convolutional LSTM to predict trajectory sequences. R. Quan et al. proposed a LSTM-based model to predict pedestrian trajectory (Quan et al. (2021)). It designed a nontraditional LSTM mechanism to capture the intention of pedestrians, and estimated trajectories accordingly.

The majority of the existing approaches require information from all pedestrians in the scene, and use this information to inform specific modules in the model to capture pedestrian interactions and complete the prediction process. This can limit the generalization ability of the predictor if the surrounding environment is not a crowded scene, or not all pedestrians' information is accessible due to reasons such as data privacy and lack of corresponding data collection facilities, e.g., surveillance cameras. By constructing a model specifically for capturing patterns such as social interaction, these models have to find a trade-off between improving prediction accuracy in specific scenes and maintaining generalization ability, as there can be a large number of factors affecting an individual's walking behavior. In contrast, the model uses minimal information and adopts a decentralized approach; that is, it only uses the trajectory profile of the pedestrian for whom prediction is taking place. All other factors affecting the pedestrian's movement are treated as unknown factors affecting the walking trajectory, or uncertainty of the environment, and are assumed to be learned by the model. In contrast to the existing literature, this setting does not limit the applicability of the proposed model to specific scenes, such as crowded spaces, but provides high quality predictions in a more general environment, which is essential for real-world pedestrian safety applications.

3.2.2 Attention Mechanism

The past several years have been witness to the development of attention mechanisms due to their ability to capture interactions and relationships between features, even when they are not close to each other in sequences (Vaswani et al. (2017); Wang et al. (2017)), which is a restriction of CNNs. Attention mechanisms have been widely adopted to enable learning patterns from sequences in many fields such as video question answering, video captioning, speech recognition, neural machine translation and so on (Ye et al. (2017); Yan et al. (2019); Chorowski et al. (2015); He et al. (2018)). A main drawback of many deep learning-based methods on learning patterns from sequences is that they are not capable of remembering and learning information in long sequences, which harms their performance (Sutskever et al. (2014); Bahdanau et al. (2014)). The attention mechanism solves this problem. Sequence prediction tasks always require sequence inputs. To conduct sequence prediction, traditional deep learning models usually use encoder-decoder pairs to encode input sequences into patterns, and then directly decode the resulting learned patterns to output sequences (Park et al. (2018); Sak et al. (2017)). The general idea of the attention mechanism is the following: when predicting a sequence with a sequence input, instead of directly connecting the encoding networks with the decoding networks, the attention mechanism creates shortcuts between the context vector built in the encoder networks. That shortcut design compresses the global information by connecting the input sequence, e.g., the entire source input, to the target sequence, e.g., an RNN sequence (Bahdanau et al. (2014); Luong et al. (2015)). This helps models improve the performance on capturing global sequence patterns.

Common attention mechanisms use input and target sequences separately to capture relationships between them. Self-attention, which is a kind of recently proposed attention mechanism, does not use separate input and target sequences. Instead of computing representations of a target sequence, given an input vector, it computes the representation of itself (Cheng et al. (2016); Parikh et al. (2016)). Unlike traditional attention mechanisms which are parameterized in a deterministic way, self-attention layers are always trainable, which further provides robustness to models. The attention kernel constructed in the proposed model is a type of self-attention.

3.3 Methodology

In this section, I introduce different components of the proposed step attention model.

3.3.1 Problem Statement

In this chapter, I formulate the pedestrian trajectory prediction problem in a generic environment. Let $X_t = (x_t, y_t)$ be the location coordinates (Cartesian coordinates) of a pedestrian at time t. The goal is to predict the position of the pedestrian within the next n time steps, i.e., $(X_{t+1}, X_{t+2}, ..., X_{t+n})$, given a history trajectory location sequence $(X_{t-m+1}, X_{t-m+2}, ..., X_t)$ of the most recent m time steps, where m is the maximum trace-back history and n is the prediction time horizon, both measured as the number of time steps.

3.3.2 Step Attention

The main idea behind the proposed model is to learn patterns in trajectory sequences, so as to provide accurate predictions for future movements. In this section, I describe the detailed architecture of the proposed step attention model, and show how the model is designed to capture sequence patterns. The model architecture is shown in Figure 3.1. This architecture is carefully designed based on the advantages and drawbacks of different types of neural networks, as well as the computation cost. RNNs are known for their ability in learning time-related patterns. Since the raw input of the model is pedestrian history trajectories, I introduce an RNN kernel to learn time-related patterns. Within the RNN kernel, based on LSTM's demonstrated advantages in learning sequential sensor data (Li et al. (2019); Wang et al. (2018)), two LSTMs are used in order to explicitly learn time-related patterns within raw trajectory sequences. After the LSTM-RNN kernel, although time-related patterns are learned, high-dimensional patterns within each time step may be missed due to the drawbacks of RNN. To overcome this problem, the second kernel, i.e., the time-distributed kernel, is designed to learn patterns for each time step. Within this kernel, deep CNNs are adopted to learn patterns within a time step. However, due to the nature of the convolution operation, CNNs learn only local information of their inputs and represent learned patterns in 2-D convolutions based on convolution kernels. These 2-D convolutions are isolated between each other and their original spatial information in inputs are missing. The augmented attention is adopted to mitigate this drawback by introducing relative spatial logits to represent relative position information of inputs, and use such information along with learnable weights to learn global patterns of inputs. The time-distributed kernel outputs a new sequence, which represents previously learned patterns in time order. Hence, another RNN is developed to further learn the sequence-related features in the series. Unlike the first RNN kernel, the second RNN kernel uses a GRU due to its computation efficiency. Note that the proposed model predicts the position of the next step based on the history trajectory information, and uses this prediction to predict future time step positions sequentially, thereby producing a sequential trajectory prediction.

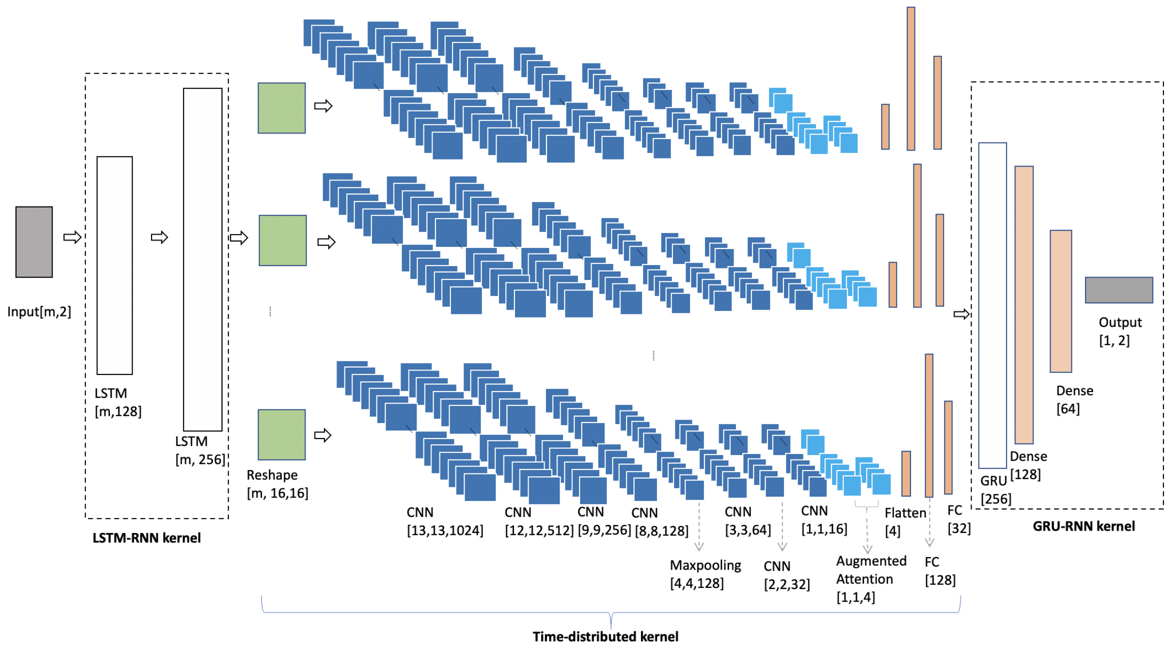


Figure 3.1: Step attention architecture

3.3.2.1 LSTM-RNN Kernel

In the past decade, RNNs have demonstrated great performance in capturing series-related features from sequences (Greff et al. (2016); Dey and Salemt (2017)). LSTM is a mechanism that is able to store and forget input information through its forget, input, and output gates. An LSTM-based RNN layer is a looped structure in which the output of a time step is connected to the next time step, as demonstrated in Figure 3.2, where m is the number of

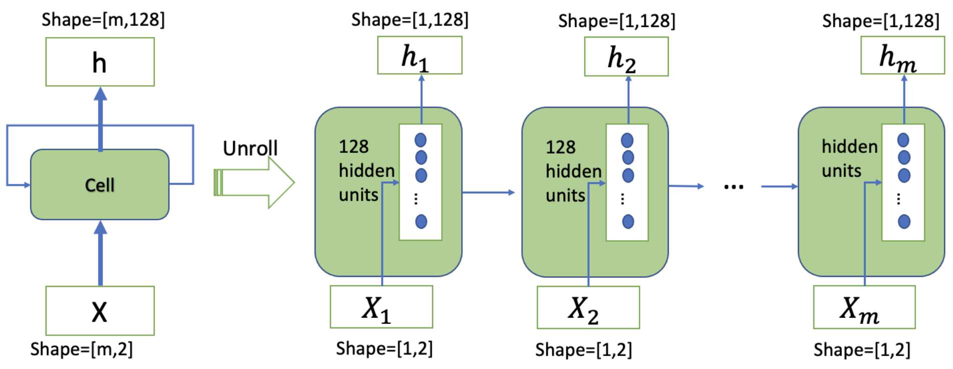


Figure 3.2: An LSTM-RNN layer illustration

time steps. Note that to make the LSTM process of $m \times 2$ shape into an output shape of $m \times 128$, the strategy I use is to set 128 units in the hidden layer of the LSTM cell. Each LSTM cell takes in a 2×2 shape vector and outputs the shape of 1×128 according to the hidden layer. An LSTM layer is formed by duplicating m LSTM cells, and therefore the output shape of $m \times 128$ is obtained after the LSTM layer.

As demonstrated in Figure 3.1, an LSTM-RNN kernel is implemented at the beginning of the architecture. The LSTM-RNN takes inputs of shape $[m, 2]$, where m is the history trajectory length and 2 is the dimension of the location coordinates. First, the trajectory is encoded and learned by an LSTM layer of shape $[m, 128]$, and then it is passed to a second LSTM layer of shape $[m, 256]$. The two LSTM layers extract time-related patterns directly from the input trajectory. The LSTM-RNN kernel processes the raw pedestrian trajectory into a new high-dimensional time-ordered sequence.

3.3.2.2 Time-distributed Kernel

I use a time-distributed kernel to capture patterns from the LSTM-RNN output sequences. Given a time-ordered sequence, the time-distributed kernel assigns a pipeline to every input in the sequence. Pipelines from different time steps do not share weights with each other. As a result, the time distributed kernel generates an ordered sequence as its output. In other words, each pipeline learns high dimensional patterns from a single time step for which it is trained.

I develop the time-distributed kernel with multiple CNN layers and an augmented attention mechanism, followed by several fully connected layers. In the beginning, the raw input trajectory sequence encoded by the LSTM-RNN layer is reshaped to $[m, 16, 16]$, i.e., into a sequence of m 2-D tensors with dimension $[16, 16]$. The purpose of the reshaping step is to prepare the encoded sequence for the following CNN kernel, as the CNN within each pipeline is constructed to take a 2 dimensional tensor as input. As shown in Figure 3.1, each pipeline begins with four CNN layers with shapes $[13, 13, 1024]$, $[12, 12, 512]$, $[9, 9, 256]$ and $[8, 8, 128]$. A max-pooling layer then performs pooling to conduct down sampling. Next, three more CNN layers with shapes $[3, 3, 64]$, $[2, 2, 32]$ and $[1, 1, 16]$ are designed to further learn high dimensional patterns. The augmented attention, which is explained in detail in the next section, is then implemented to squeeze and learn the patterns processed by the previous layers, and generate an output of shape $[1, 1, 4]$. Three fully connected layers with shapes $[4]$, $[128]$ and $[32]$ are then attached to learn from the patterns generated by the augmented attention mechanism. The final output of the time-distributed kernel is of the shape $[m, 32]$, as each trajectory contains m time steps.

Through the time-distributed kernel, the proposed architecture learns patterns in a high dimensional space within every time step. That is, rather than learning a global sequence pattern from the input trajectory, which is the task of the RNN kernels in the model, the time-distributed kernel focuses on learning patterns in each time step. As convolution sequences are generated by the CNN layers in the time-distributed kernel for each time step, the augmented attention mechanism learns the *global patterns in the convolution sequences generated in a time step*, which mitigates the drawback of CNNs, i.e., learning only local information.

3.3.2.3 Augmented Attention

Outputs of CNNs in the proposed architecture are sequences of 2 dimensional (2-D) convolutions. For each time step in the time-distributed kernel, after several CNN layers, I obtain long convolution sequences that represent local information of their leading layers. I introduce augmented attention to allow the model to capture global information of the convolutions from the previous layer. Traditionally, the family of attention mechanisms, including the self-attention mechanism, is used for one dimensional (1-D) inputs. I. Bello et. al proposed an attention mechanism that is augmented to convolutional networks. The augmented attention mechanism sees accuracy improvement over the ResNet50 (He et al. (2016)) baseline and outperforms other attention mechanisms (Bello et al. (2019)). The standard 1-D self-attention uses query Q, key K and value V logits to compute multi-head attention (MHA) results as proposed in the Transformer architecture (Vaswani et al. (2017)). Given a flattened 1-D input matrix X of the original matrix, the resulting value O_h of a single head h is computed as in Eq. (3.1).

$$\begin{aligned} O_h &= \text{Softmax}(\frac{(XW_q)(XW_k)^T}{\sqrt{{d_k}^h}})(XW_v) \\ &= \text{Softmax}(\frac{QK^T}{\sqrt{{d_k}^h}})V \end{aligned} \tag{3.1}$$

where W_q, W_k and W_v are learned weights of query, key and value, respectively. d_k is the depth of keys per attention head.

In Eq. (3.1), spatial information related to input width and height is lost as the original input is flattened into a 1-D vector. This may harm the performance of the model in learning global patterns. As a single convolution output is always 2-D, in the proposed framework I use an augmented attention mechanism to take 2-D convolutions as inputs, and learns global and spatial information within convolutions. In augmented attention, relative spatial information in a convolution is preserved by introducing additional logits to compute O_h, as demonstrated in Eq. (3.2):

$$O_h = \text{Softmax}(\frac{QK^T + S_H^{rel} + S_W^{rel}}{\sqrt{{d_k}^h}})V \tag{3.2}$$

In Eq. (3.2), S_H^{rel} and S_W^{rel} are logits representing relative position information from the input, computed as:

$$S_H^{rel}[i,j] = q_i^T r_{j_y - i_y}^H \tag{3.3}$$

$$S_W^{rel}[i,j] = q_i^T r_{j_x - i_x}^W, \tag{3.4}$$

where $r_{j_y - i_y}^H$ and $r_{j_y - i_y}^W$ are learned representations for relative height, $j_y - i_y$, and width, $j_x - i_x$, respectively. Here, q_i is the query vector at position $i = (i_x, i_y)$. The output O_h of Eq. (3.2) is a single head. While the input is 2-D, I obtain a multi-head attention (MHA) 2-D output. The augmented attention mechanism is then constructed by concatenating the attention with the previous convolution layer.

3.3.2.4 GRU-RNN kernel

The output of the time-distributed kernel is a set of trajectory patterns that are represented by a sequence of high-dimensional tensors. Here, I introduce another RNN kernel based on a GRU-RNN network to further learn time-related patterns in the new sequence, and decode it into future positions. Similar to LSTM, GRU uses gate design. However, the gates are limited only to reset and update gates. This makes GRU-based models more computationally efficient (Nurvitadhi et al. (2016)). The GRU-RNN kernel learns the time-related patterns of the new sequence. Unlike the LSTM-RNN kernel, the GRU-RNN layer does not return a time series sequence; instead, the kernel removes the temporal dimensions (hidden state outputs from time step 1 to $t-1$) of the GRU states, and outputs the state of the most recent GRU unit, which produces a one dimensional output. The GRU-RNN takes result of previous layers and then squeeze them into the shape [256], followed by three fully connected layers, with shapes [168], [64], and [2]. The model generates results with shape $[1, 2]$ as the prediction for the next time step.

3.3.2.5 Sequential Prediction

The proposed architecture predicts future positions of a pedestrian in a sequential manner. The model outputs n predictions through n executions, where each execution predicts the future position in a single time step. Given an history trajectory buffer $(X_{t-m+1}, X_{t-m+2}, ..., X_t)$ in the first execution, the model predicts the position of time step X_{t+1}. Next, the model removes the oldest point (i.e., X_{t-m+1}) from the history trajectory, and pushes the predicted position into the buffer, producing the new input sequence

$(X_{t-m+2}, ..., X_t, X_{t+1})$. This is followed by another $n-1$ executions of the model, where in each execution the oldest point is removed from buffer and the most recent prediction is added to it. The final result is a predicted trajectory $(X_{t+1}, X_{t+2}, ..., X_{t+n})$ with length n. The sequential prediction strategy is shown in Figure 3.3. Note that the model needs to be trained only once. Once the model is trained, by updating the trajectory buffer, a trajectory sequence with any desired length n can be predicted by invoking the trained model sequentially n times.

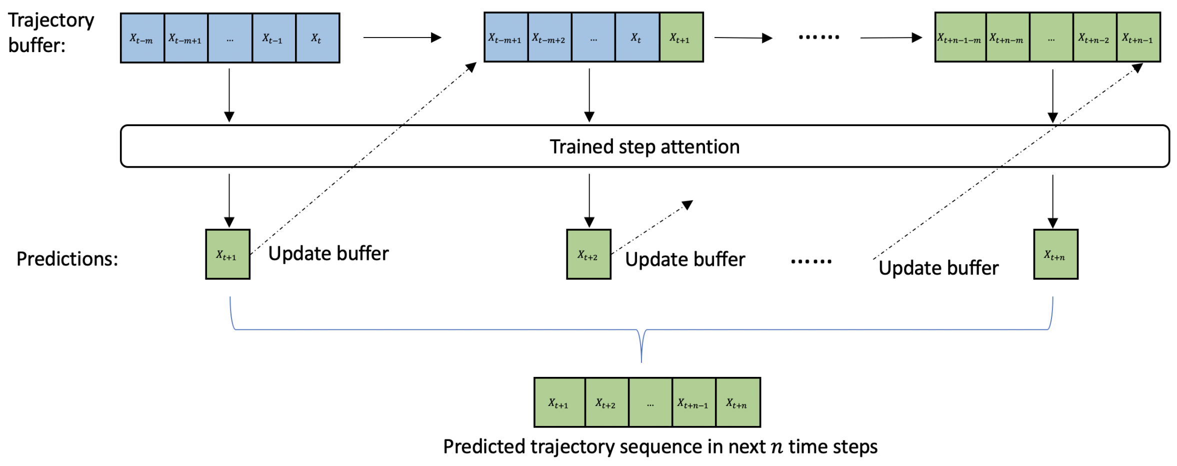


Figure 3.3: The sequential learning strategy for trajectory prediction

Unlike models that use a fixed-length sequence prediction strategy, this sequential prediction strategy allows us to freely control the length of the prediction horizon, which can prove helpful for real-world applications.

3.4 Experiments and Results

In this section, I study the performance of the proposed model by evaluating it on the publicly available TrajNet dataset (Sadeghian et al. (2018)) against three benchmark methods. Trajectories in the TrajNet dataset are collected in locations such as hotel entrances, campuses, public walking areas, etc. Since TrajNet trajectories are mainly collected in crowded spaces, this dataset has become a well-known benchmark for trajectory prediction when modeling human interactions. I also use a second proprietary dataset whose trajectories are extracted from roadside sensors in a typical inter-city intersection in the US. I use this dataset to validate the generalization performance of the proposed step attention model in real-world settings that are not necessarily crowded. Finally, I conduct a complementary set of experiment in Ann Arbor, MI, USA, to demonstrate the model performance and illustrate prediction results on satellite maps.

3.4.1 Data Collection and Preparation

The TrajNet dataset was originally collected through mounted cameras and drones on university campuses, hotel entrance, etc. Videos were then processed and converted into images. After that, positions of pedestrians were extracted. Next, pedestrian locations were converted to trajectories represented by local coordinates. In the chapter I use the same dataset as used by the benchmark methods (Alahi et al. (2016); Gupta et al. (2018)).

The proprietary dataset was originally collected in video format through roadside cameras. Videos were then converted into image frames. Within each frame, pedestrians were detected, and the corresponding pixel locations in the image were obtained. With the longitude and latitude information of a known base point (e.g., center of the intersection), previous extracted pedestrian locations were converted to geo-locations represented by longitude and latitude. Finally, the location information were converted to Cartesian coordinates, allowing trajectories to be represented using relative locations. As raw data were in video format, the output of the processing was also be in time order. The final location information was then used by the model for training and testing.

For the complementary experiments, data was collected using smartphones (i.e., IPhone devices equipped with SensorLog (Mosenia et al. (2017))) in an intersection in Ann Arbor, MI. Raw pedestrian motion data were recorded in real-time including longitude, latitude, speed, etc. Location information, i.e., longitude and latitude, were then extracted offline at the trajectory level. The collected trajectories were later converted to local coordinates with the center of the intersection as the origin.

3.4.2 Experiments on the TrajNet Dataset

I compare the proposed step attention model with three benchmark models, including social LSTM (Alahi et al. (2016)), occupancy LSTM (Alahi et al. (2016)) and social GAN (Gupta et al. (2018)) on the TrajNet dataset. These benchmark models are considered as state-of-the-art on pedestrian trajectory prediction, and are the most commonly used benchmark methods. The TrajNet dataset is a collection of multiple sub-datasets, including trajectories collected from top-down view cameras and drones. It also contains trajectories from earlier pedestrian trajectory datasets including the ETH (Pellegrini et al. (2010)) and UCY (Lerner et al. (2007)) datasets, and many drone-collected trajectories, mainly from university campuses. The dataset was initially constructed and used by the proposers of these benchmark methods to test and demonstrate their model performance for trajectory prediction in crowded spaces (Social-LSTM (2016); Occupancy-LSTM (2016); Social-GAN (2018)).

In the experiments with the TrajNet dataset, I use 8 history frames (a total of 3.2 s) for training–a setting that was also adopted by the three benchmark methods–and predict the next 12 frames (a total of 4.8 s). I use two metrics to compare the performance of the model against those of the benchmarks: (i) the average displacement error (ADE), and (ii) the final displacement error (FDE). ADE is computed by taking the average error of the 12 predicted time steps for a given trajectory, averaged across the entire dataset. FDE is computed by averaging the displacement of the final time step (12th time step) across all trajectories in the dataset. Table 3.1 compares the ADE and FDE measures of the proposed model against the three benchmarks, showcasing the superior performance of the step attention model.

Table 3.1: Benchmark Performances

Displacement Errors Benchmark Models	ADE (m)	FDE (m)
Social LSTM	0.68	2.10
Occupancy LSTM	1.10	3.12
Social GAN	0.56	2.11
Step Attention	**0.53**	**1.72**

Table 3.2: ADE and FDE metrics reported in the form ADE/FDE for all TrajNet sub-datasets for step attention and the three benchmarks

Benchmark Models Dataset	Social LSTM	Social GAN	Occupancy LSTM	**Step Attention**
BIWI ETH	0.98/3.02	1.02/4.21	2.11/6.07	**0.59**/**2.00**
Crowds Zara	0.69/2.18	0.41/1.58	1.12/3.19	**0.36**/**1.30**
Crowds Univ.	0.77/2.46	0.53/2.26	1.53/4.58	**0.51**/**1.88**
Drone 1	0.45/1.35	0.35/1.29	0.61/1.73	**0.29**/**1.03**
Drone 2	0.84/2.68	0.62/2.38	1.26/3.60	**0.56**/**1.92**
Drone 3	2.66/8.80	1.73/8.18	4.16/10.14	**1.28**/**5.71**
Drone 4	0.80/2.66	0.70/2.92	1.24/3.48	**0.53**/**1.85**
Drone 5	0.48/1.51	0.31/1.18	0.68/1.97	**0.25**/**0.88**
Drone 6	0.57/1.76	0.51/1.86	0.91/2.60	**0.38**/**1.32**
Drone 7	1.28/4.17	1.35/4.85	2.54/6.81	**0.97**/**3.39**
Drone 8	0.68/1.61	0.60/2.14	0.92/1.97	**0.23**/**0.70**
Drone 9	0.81/2.49	0.53/2.21	1.41/4.00	**0.46**/**1.64**
Drone 10	0.68/2.06	3.27/3.98	1.17/3.36	**0.35**/**1.21**
Drone 11	0.59/2.01	0.44/1.73	1.06/3.20	**0.30**/**1.07**
Drone 12	0.68/2.10	0.46/1.81	1.15/3.25	**0.42**/**1.43**
Drone 13	0.74/2.31	**0.52**/2.04	0.93/2.61	0.55/**1.80**
Drone 14	0.72/**2.22**	**0.63**/2.50	1.25/3.62	1.36/3.81
Drone 15	0.72/2.24	0.57/2.23	1.11/3.15	**0.44**/**1.48**
Drone 16	0.29/0.95	**0.17**/0.94	0.56/1.56	0.20/**0.71**
Drone 17	0.69/2.17	0.56/2.33	1.12/3.13	**0.42**/**1.41**

As Table 3.1 demonstrates, the proposed step attention model outperforms all three benchmark methods in terms of both ADE and FDE. The overall performance reported in this table is computed by averaging the ADE and FDE metrics across all sub-datasets.

The ADE of the step attention model is 0.53 m. Not only is this value smaller than the ADE values of the benchmark models, but it is also smaller than the typical range of a single step length of a pedestrian, which is between 0.71 and 1.49 m (Hoeger et al. (2008)). Moreover, the FDE of the step attention model for the 4.8 s long-horizon is 1.72 m. This value is much smaller than those of the three benchmarks, with values 2.10, 3.12 and 2.11 m for the Social LSTM, Occupancy LSTM, and Social GAN, respectively.

The detailed performance results of all sub-datasets are shown in Table 3.2. These results are shown in the form of (ADE/FDE) for each sub-dataset. The model that provides the best performance in terms of ADE and FDE metrics is highlighted in bold for each sub-dataset.

This table indicates that the step attention model outperforms the three benchmarks across all sub-datasets almost universally. Additionally, analysis on sub-datasets indicates that in most cases the ADE error of step attention is within a single step length of an adult.

The benchmark sub-datasets in the TrajNet dataset are mainly composed of trajectories collected from crowded spaces, providing an ideal benchmark for methods that seek to learn human interactions in crowded spaces. In contrast, the step attention model is not specifically structured to capture human interactions. This distinction highlights two benefits of the step attention model: (1) even though it is not designed specifically to capture interactions in crowded spaces, it can nonetheless outperform state-of-the art methods and provide high performance in crowded space scenarios, and (2) its generic architecture allows it to be implemented in scenarios where interactions between individuals is not the driving factor to govern their motion.

3.4.3 Experiments at Signalized Intersections

Tables 3.1 and 3.2 demonstrate the promising performance of the step attention model over the state-of-the-art methods on the benchmark dataset. However, the TrajNet dataset can deviate from commonplace scenarios that can be realized in the real world as it concentrates on crowded spaces. There are predictive models that perform well on TrajNet but have poor performance when generalizing to real-world instances. For example, MATF outperforms Social LSTM on TrajNet, but its average displacement error can be more than 3 m for a 4-seconds-long prediction when extended to a private real-world dataset called Massachusetts Driving Dataset (Zhao et al. (2019)).

One of the main applications of the proposed model is to enable autonomous vehicles to accurately predict trajectories of pedestrians, so as to ensure their safety when planning vehicle motion or allow them to intervene with active safety functions. In this section, I demonstrate the performance of the step attention model on two datasets collected at signalized intersections.

The first dataset is a proprietary dataset collected through infrastructure sensors at a 4-way intersection, and contains more than 10,000 pedestrian trajectories. These trajectories are segmented into large sub-sequences for training and testing. Unlike the TrajNet

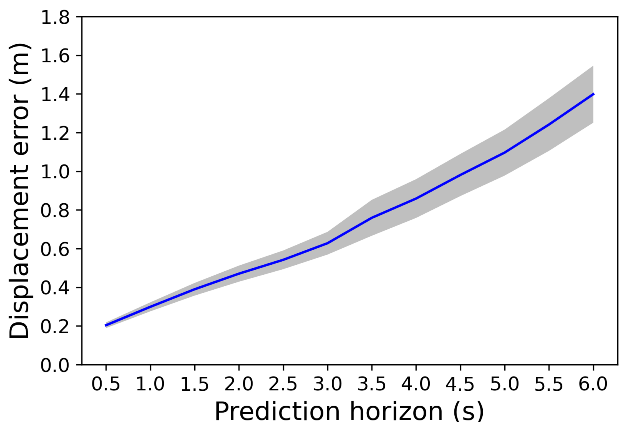


Figure 3.4: The average displacement error at each prediction time step across the test dataset of the proprietary dataset. The shaded region highlights the 95% confidence interval

dataset, this dataset is not limited to crowded scenarios, thereby enabling us to assess the generalization performance of the step attention model. All pedestrian trajectories used for the study have been captured using cameras and then converted into pedestrian trajectories (sequences of data coordinates) using detection and tracking algorithms.

As the prediction is done in a sequential manner, the error of earlier predictions can propagate to later predictions. To demonstrate the performance, similar to the demonstration strategy adopted by (Zhao et al. (2019)), I report the ADE metric averaged at each prediction time step to show detailed model performance over the entire test set, as demonstrated in Figure 3.4.

While the prediction horizon in the literature is typically limited to 4 to 5 s, here I predict the pedestrian trajectories for the next 6 seconds, extending the prediction horizon beyond the existing literature. Experimental results indicate that step attention can obtain ADE/FDE of 0.52/0.98 m for a 4-seconds-long prediction, and 0.74/1.40 m for a 6-seconds-long prediction. Figure 3.4 shows how the model prediction error propagates with the length of the prediction horizon. The shaded region highlights the 95% confidence interval. As this figure demonstrates, the prediction error and its 95% confidence interval increase with the length of the prediction horizon. However, the increase in the displacement error is approximately linear for a rather long horizon, showcasing the performance of the step attention model. When deploying the model in a real-world setting, the prediction horizon

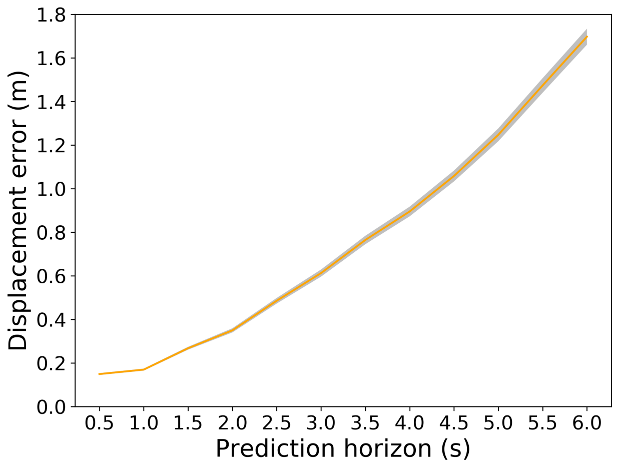
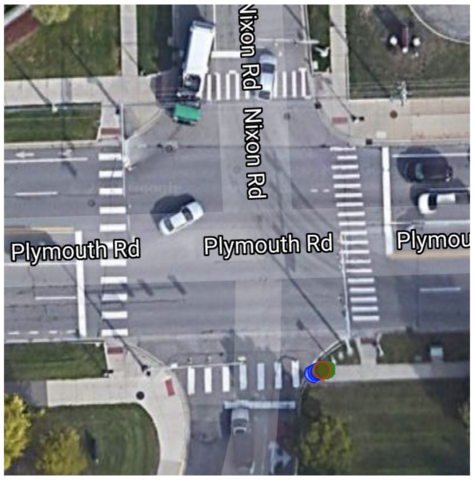


Figure 3.5: The average displacement error at each prediction time step across the test dataset of the complementary dataset. The shaded region highlights the 95% confidence interval

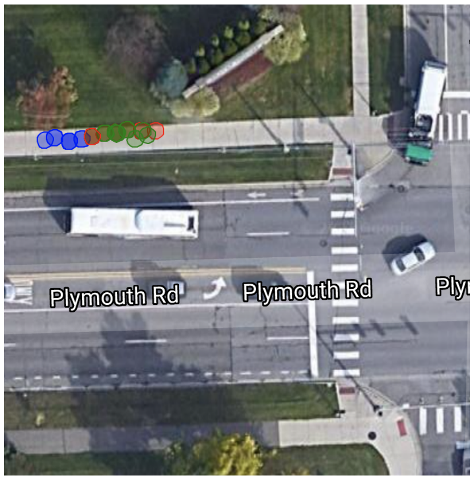
can be set based on the desired tolerance for the displacement error.

Due to the proprietary nature of the previous dataset, I are not able to visualize and analyze the prediction performance under different scenarios. To visually illustrate the performance of step attention, I conduct a complementary set of experiments by collecting pedestrian trajectories using IPhone devices equipped with SensorLog (Mosenia et al. (2017)) as the sensor platform at a major intersection in Ann Arbor, Michigan, USA. The dataset is constructed similarly to the proprietary dataset by using a moving time window on raw trajectories. This dataset has 226,947 samples for training, 28,368 samples for validation, and 28,369 samples for testing.

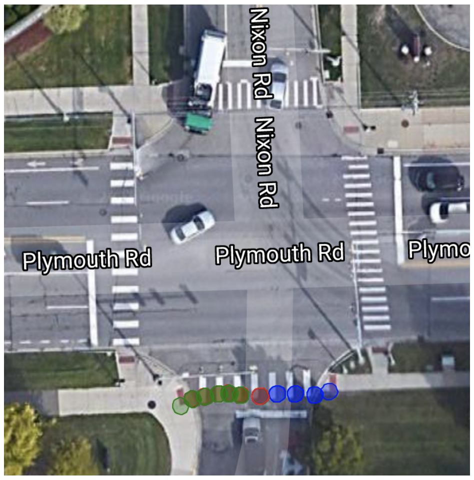
Figure 3.5 shows the overall performance of step attention on the test dataset. For a 6-seconds-long prediction, the ADE/FDE of the proposed step attention model is 0.76/1.70 m. Similar to the proprietary dataset, both the displacement error and the confidence interval increase when the prediction horizon increases, showing the further I predict into the future, the lower the accuracy. However, the overall low ADE and FDE values demonstrate that step attention is able to produce accurate predictions within a 6-seconds-long prediction horizon. The training and validation error trends are shown in Figure 3.7. It demonstrates that the training error converges to a small error within 100 epochs, while no over-fitting occurs as the validation loss decreases gradually.



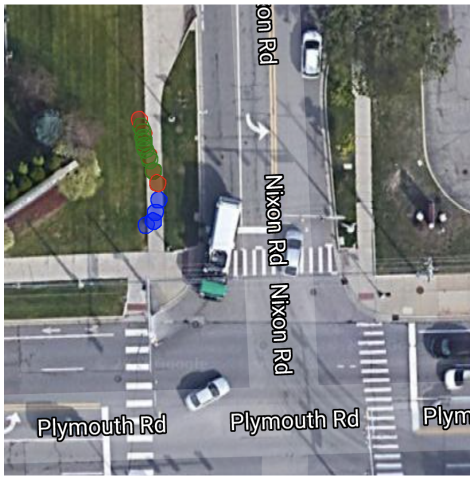
(a) Scenario 1: stopped at intersection, ADE=0.03m



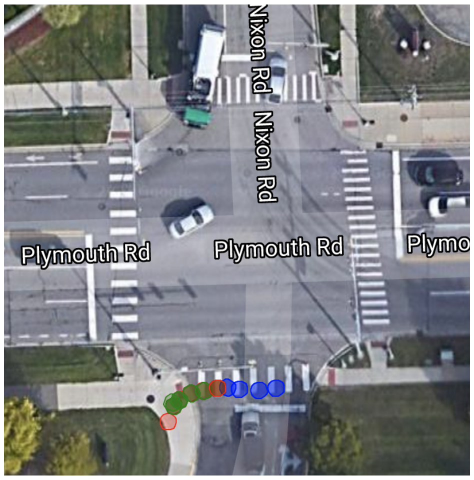
(b) Scenario 2: walking straight on sidewalk, ADE=0.22m



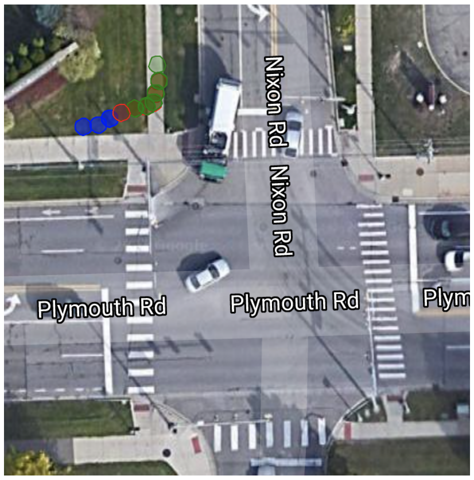
(c) Scenario 3: walking on crosswalk, ADE=0.50m



(d) Scenario 4: curvy walking on sidewalk, ADE=0.29m



(e) Scenario 5: turning on crosswalk, ADE=0.49m



(f) Scenario 6: turning on sidewalk, ADE=0.30m

Figure 3.6: Prediction results for the case study in an intersection. The observed trajectories, i.e., 4-seconds-long history trajectories used for training, are colored in blue, while the predicted and the ground truth trajectories are colored in green and red, respectively.

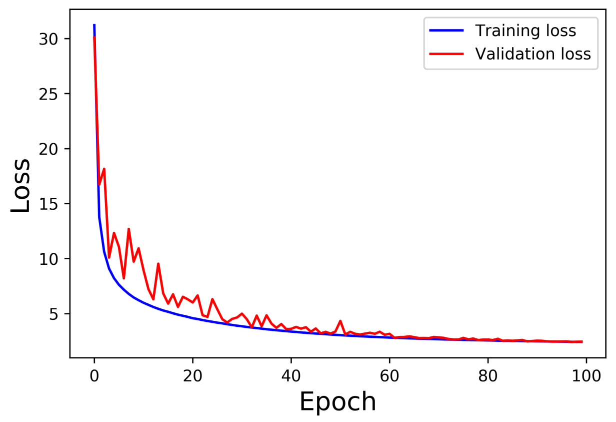


Figure 3.7: Learning curve of the proposed model on the complementary dataset

Figure 3.6 illustrates the performance of the step attention model under different walking scenarios. For each scenario, one representative case is selected, and the pedestrian trajectory is plotted on top of a satellite map to better visualize the road geometry and the surrounding environment. Scenario 1 in Figure 3.6(a) shows the predicted trajectory when the pedestrian is stopped at the intersection. Step attention accurately predicted the future trajectory which consists of the pedestrian not moving. Scenario 2 in Figure 3.6(b) demonstrates model performance on a randomly selected case where the pedestrian walks straight on the sidewalk toward the intersection. Here, the corresponding ADE is 0.22 m for a 6-seconds-long prediction horizon. Scenario 3 in Figure 3.6(c) depicts and instance of walking on the crosswalk. The corresponding ADE is 0.5 m. Scenario 4 in Figure 3.6(d) shows model performance on the case where the pedestrian walks in a curvy pattern on the sidewalk. The model accurately predicts the future 6-seconds-long trajectory with an ADE of 0.29 m. Both Figure 3.6(e) and Figure 3.6(f) illustrate the performance of the model on predicting turning behaviors. In scenario 5, the pedestrian walks on the crosswalk but turns left to the sidewalk. The step attention model accurately predicts this behavior, with corresponding ADE of 0.49 m. In scenario 6, the pedestrian is walking on the sidewalk towards the intersection, and makes a turn to continue walking on the sidewalk rather than cross the intersection. As demonstrated in Figure 3.6(f), step attention successfully predicts the future trajectory with an ADE of 0.30 m.

3.4.4 Error Sources

The model error results from multiple sources: i) sensor quality: All data used in this chapter are collected through sensors such as cameras or potable devices. The quality of the sensors largely affects the quality of the collected data, and consequently the performance of the model; ii) prediction horizon: The prediction error is also affected by prediction horizon–the longer the prediction horizon, the larger the error. The proposed work adopts a sequential prediction strategy, indicating that prediction error can accumulate and propagate across time. However, one advantage of sequential prediction is that users can freely control the prediction horizon based on their needs and/or tolerance for error; iii) unexplored features in the input space: Similar to other supervised learning methods, the proposed model learns patterns from the training data. If a not-previously-seen pattern appears, the prediction outcome can be less accurate due to the lack of exposure of the model to that pattern at the training stage.

3.4.5 Computational Efficiency

With an NVIDIA TITAN V GPU, the computation time of one epoch on training and validation sets is 750 s with the batch size of 32. Corresponding computation time is 0.11 s for predicting a 6-seconds-long trajectory. With the same hardware setting, it takes 0.17 s, 0.05 s, and 0.09 s for Social LSTM, Occupancy LSTM and Social GAN to produce the prediction of the same trajectory length, respectively. As specified previously, it takes 0.33 s to 0.53 s for an adult to take a step. Once the model is trained offline, the prediction can be completed within a step time. Therefore, the proposed model can provide real-time prediction results.

3.5 Conclusion

In this chapter, I propose a deep learning model, called step attention, to predict pedestrian trajectories. The proposed model architecture is designed to learn patterns from time-series input sequences. The architecture is built using a combination of RNN and CNN layers and a novel augmented attention mechanism. Unlike the existing literature on pedestrian trajectory prediction that aims to learn factors affecting an individual's walking behavior, e.g., social interactions, the proposed step attention model has a more general design that aims to learn trajectory patterns directly from input sequences, without making any assumptions on the environmental settings. This makes step attention more robust, and applicable to a wide range of real-world scenarios.

I showcase the performance of the step attention model by comparing it to three state-of-the-art benchmark methods, including social LSTM, social GAN, and occupancy LSTM, on the well-known TrajNet dataset. For a 4.8-seconds-long prediction, the typical prediction horizon length used in the literature, step attention outperforms the benchmark methods in terms of average and final displacement errors. The average displacement error of step attention (0.53 m) on the TrajNet dataset is less than a single step length of an adult.

I conduct two additional sets of experiments using two pedestrian trajectory datasets collected at signalized intersections. The first dataset contains pedestrian trajectories collected using infrastructure-based cameras. Experimental results indicate that step attention can obtain ADE/FDE of 0.74/1.40 m for a 6-second-long prediction horizon. The second dataset is collected using portable devices. Step attention obtains an ADE/FDE of 0.76/1.70 m on this dataset for a 6-second-long prediction horizon. Furthermore, using this dataset I provide visual validation that the proposed step attention model can accurately predict future trajectories for different walking behaviors. In both experiments, I demonstrate that the displacement error increases at an approximately linear rate with the length of the prediction horizon for horizons as long as 6 seconds.

CHAPTER IV

A Learning-based Trajectory Prediction Approach for Heterogeneous Traffic Agents

4.1 Introduction

The past decade has witnessed the rapid development of autonomous vehicle (AV) technology. AVs are able to drive on their own without any human intervention (Fagnant and Kockelman (2015)). AVs are envisioned to lower traffic accident rates (Hulse et al. (2018); van Wyk et al. (2019)), reduce greenhouse gas emissions (Faisal et al. (2019)) and reduce traffic congestion (Metz (2018)), and enhance mobility (Zhang et al. (2020); Masoud and Jayakrishnan (2016); van Wyk et al. (2020); Abdolmaleki et al. (2019); Liu et al. (2020)). To achieve these goals, AVs are expected to accurately perceive, predict, and interact with their surrounding transportation system (Kato et al. (2015); Miglani and Kumar (2019); Yoon and Kum (2016)). This chapter focuses on the prediction component, where an AV uses the spatial temporal information of its surrounding agents to predict their future trajectories. Using these predictions, AVs can prepare themselves for safely interacting with other road users, e.g., avoiding collisions (Lefèvre et al. (2014)), improving decision-making performance (Shi et al. (2020)), etc. Accurate trajectory prediction enables AVs to anticipate future movements of their surrounding road users, and plan in advance by incorporating these predictions in their motion planning, e.g., by incorporating model predictive control (Kim et al. (2001); Liu et al. (2017)). This can help improve driving performance and mitigate risks both to the AV occupants and its surrounding road users.

When incorporated in road-side units, such predictive algorithms can be utilized by state and federal transportation agencies for managing traffic through adjusting traffic control units (e.g., traffic signals and ramp meters) and/or issuing alerts to agents at risk.

Trajectory prediction for traffic agents can be a challenging task for three main reasons: (i) The complex nature and heterogeneity of traffic agents: Each traffic agent typically aims to maximize their own utility, where utility functions may be different across agent types with different dynamics (Ma et al. (2019)), or even across time and space for a single agent type (Hoeger et al. (2008)); (ii) The complexity of traffic systems: Traffic systems consist of different agent types that interact with each other and environmental factors; (iii) Increasing accumulated uncertainty of prediction within the prediction horizon: When predicting future movements, due to the uncertainty of the environment, the prediction error can accumulate along the prediction horizon, rendering predictions that are further into the future less reliable.

In this chapter I use a learning-based approach for trajectory prediction based on the previous work discussed in Chapter III that focuses on pedestrian motion prediction. Unlike the existing learning-based models that develop agent-specific model architectures, in this work I use a single model architecture to predict trajectories for different agent types, including pedestrians, vehicles, and cyclists. I train the model on a number of agent-specific datasets, and evaluate its performance on real-world scenarios. Results show that the learning-based method outperforms EKF and provides high prediction performance when predicting 5-second trajectories for pedestrians, vehicles, and cyclists. The major contribution of this work is that it utilizes a learning-based approach for general purpose trajectory prediction for traffic agents of different types. The approach does not use agent-specific architectures or designs, and provides highly accurate results for long-horizon trajectory prediction. Furthermore, I show that with the proposed approach, trajectory prediction accuracy for cyclists, which are less commonly observed in transportation systems compared to pedestrians and vehicles, can be boosted with transfer learning. The rest of the chapter is organized as follows. I first review the existing work. Next, I elaborate the model design and its training and prediction strategies. Finally, I evaluate and show the performance of the model for vehicles, pedestrians, and cyclists using a public dataset collected by a set of autonomous vehicles, and show that transfer learning can be used to significantly improve the prediction accuracy of cyclist trajectories.

4.2 Related work

In this section I review the existing work on trajectory prediction. Current work can be roughly categorized into two classes: (i) physics-based models, and (ii) learning-based models.

Physics-based approaches model subjects as dynamic entities that follow the laws of physics. These models can be more or less complicated depending on the extent to which they keep track of the properties of entities to perform prediction. Since this type of models typically only rely on low level properties of motion, e.g., dynamic and kinematic properties, physics-based motion models are limited to short-term motion prediction. Typically, they are unable to anticipate any changes in vehicle motion caused by the execution of a particular maneuver, e.g., slow down and then accelerate to make a turn at an intersection, or sudden changes caused by external factors (Xie et al. (2017)). Among physics-based models, Kalman filter (KF)-based approaches are the most wildly used methods due to their generalization ability (Zernetsch et al. (2016); Schulz et al. (2018); Prevost et al. (2007)). C. G. Prevost et al. proposed an extended Kalman filter (EKF) model to estimate the trajectory of moving objects (Prevost et al. (2007)). In their work they utilized dynamics of objects observed from Unmanned Aerial Vehicle (UAV) to build a motion model, and provided predictions based on EKF. They demonstrated that the method can successfully estimate future trajectories on a simulated dataset; however, result can be very sensitive to pre-defined parameters in the model. J. Schdulz et al. proposed an unscented Kalman filter method for driver intention estimation and trajectory prediction (Schulz et al. (2018)). They showed that their method improved prediction accuracy and provided a lower run time compared to a sequential Monte Carlo method.

Besides KF-based models, there also exists a body of literature that uses probabilistic estimation for trajectory prediction using a physics-based approach. L. Hewing et al. proposed a trajectory prediction method based on Gaussian process dynamics (Hewing et al. (2020)), in which trajectory dynamics were modeled as a Gaussian process, and predictions were generated by estimating a Gaussian distribution. They claimed that this method can improve prediction accuracy at the cost of increased computational demand for long prediction horizons. However, as indicated by authors, in some cases this method can lead to significant prediction errors due to the local approximation around the mean.

Another body of literate uses learning-based methods, based on machine learning and artificial intelligence, for trajectory prediction. Learning-based methods have garnered considerable attention recently in light of their good performance in long horizon predictions and superior generalization ability (Ma et al. (2019); Altché and de La Fortelle (2017); Alahi et al. (2016); Deo and Trivedi (2018b)). Existing learning-based methods are mainly agent-specific, i.e., they use different model architectures for different agent types, and even for different environmental settings. A Alahi et al. proposed an approach called social LSTM, where they modeled human interactions using a long short term memory (LSTM)-based neural network to predict pedestrian trajectories in crowded spaces (Alahi et al. (2016)). This method achieved good performance in predicting pedestrians' movements in a 4.8-second prediction horizon. However, the method is designed specifically for pedestrians in crowded spaces, limiting its application to other agent types, scenarios, or even pedestrians in non-crowded spaces. N. Nikhil et al. proposed a convolutional neural network (CNN)-based approach for human trajectory prediction (Nikhil and Tran Morris (2018)). Their model used CNN to learn temporal representations of pedestrian behavior, and enabled parallelism. It provided competitive results compared to other models, including social LSTM. K. Saleh et al. proposed a recurrent neural network (RNN)-based approach for cyclist trajectory prediction (Saleh et al. (2018)). This method utilized bidirectional LSTM to learn from cyclist trajectories, and outperformed EKF by 50% in terms of displacement error within a 3-second prediction horizon. However, the model used longitude and latitude as its input format, rendering it geo-location dependent. Similarly, Z. Huang et al. proposed an LSTM-based model to predict cyclist trajectories. This model learned the interactions between a cyclist and its surroundings, and obtained a 0.86m average error within a prediction horizon of 2.4 seconds (Huang et al. (2021)). B. Kim et al. proposed a probabilistic method for vehicle trajectory prediction based on RNN (Kim et al. (2017)). This method utilized occupancy grid maps to discretize the output space in order to generate accurate predictions. The authors indicated that the method worked reasonably well in a 2-second prediction horizon.

In addition to aforementioned agent-specific models, there are a few methods that provide general models for different types of agents. Y. Ma proposed an LSTM-based model to learn agent behaviors and their interactions (Ma et al. (2019)). This method gained around 20% improvement in accuracy compared to agent-specific methods, e.g., social LSTM (Alahi et al. (2016)), social attention (Vemula et al. (2018)), etc. The approach used raw image sequences as inputs, and required the availability of information from all agents at all time steps. By modeling the interactions between agents, this method could learn and predict the behaviors of agents within the scenario from which the images were generated. H. Cheng et al. proposed an LSTM-RNN model for mixed traffic trajectory prediction. They indicated that the model can well predict traffic agents' trajectories in a shared space within a 3-second prediction horizon (Cheng and Sester (2018)).

As described above, the existing work suffers from one or more of the following drawbacks: (1) Agent-specific model architectures, e.g., model architectures are designed specific for pedestrians; (2) Discrete output spaces, as discretizing the output space can harm the prediction accuracy. Because in real-world scenarios, trajectories are continuous; (3) Short-prediction horizons; (4) Sensitivity to model parameters; and (5) High information requirement, e.g, information of all agents present in a scene.

In contrast to the existing literature, in this chapter, I propose a generic model to predict trajectory of traffic agents, regardless of their type of the environment. Unlike methods that require information from all agents in a scene, the proposed model only uses history trajectory information of the subject agents, and learns patterns within those trajectories. The model does not use agent-specific design in its architecture, and provides trajectory predictions in a continuous space. The model uses a single architecture and learning setting, and is trained with different datasets collected from different agent types. The model is capable of providing high accuracy prediction within a 5-second prediction horizon.

4.3 Methodology

In this section I delve into the proposed methodology, including the problem definition, the model architecture, and the learning and prediction settings of the proposed approach.

4.3.1 Problem Statement

In this chapter I consider the problem of predicting an agent's trajectory in a heterogeneous transportation system with mixed agent types by utilizing their history trajectory. The proposed approach obverses the most recent trajectory history of a subject agent, and learns its motion patterns. Formally, the trajectory prediction problem in a generic environment can be formulated as follows: Assume $P_t = (P_{xt}, P_{yt})$ is the position of an agent at time t under a Cartesian coordinate system, where $P_{xt} \times P_{yt} \in \mathbb{R}^2$. An agent's most recent history trajectory of length m can be represented as $(P_{t-m+1}, P_{t-m+2}, ..., P_t)$. The objective of the problem is to predict the position of the agent for the next n time steps, i.e., $(P_{t+1}, P_{t+2}, ..., P_{t+n})$.

4.3.2 Model Architecture

Developing different architectures for different agent types can be time-consuming and harder to maintain in real-world applications. More specifically, with agent-specific architectures, the modeler would need to tune hyper-parameters of all model structures (e.g., training optimizer, learning rate, etc) separately, making it more difficult to update and maintain the different structures and requiring more resources (i.e., data, time, and computational resources). In the proposed approach I develop a generic deep learning model with a single set of hyper-parameters to learn patterns in trajectories directly, and provide accurate predictions on agents' future positions. Instead of using different model architectures, the proposed approach provides agent-specific predictive models by conducting the training process on agent-specific datasets.

The common architecture used in this study includes a combination of RNN and CNN kernels. The input to the architecture includes two-dimensional local coordinates of the history trajectory of an agent. First, an LSTM-based RNN kernel is used to process the raw trajectory inputs and learn time-related patterns. The patterns in each time step are then reshaped into a square tensor, and are fed to a time-distributed kernel constructed by CNNs and augmented attention, which is used to capture global information from the CNN outputs in order to learn patterns within a time step. A number RNN layers are then used to decode the learned patterns in the time-distributed kernel. Finally, the architecture outputs the position of the agent within the next time step.

4.3.3 Prediction and Training Settings

During the prediction process, the trained model predicts the future position of an agent one time step ahead at a time. Hence, a full trajectory is generated sequentially. When predicting a trajectory, the model maintains an time-ordered observation buffer with size m. Starting with an observation buffer of m time steps of history trajectory, i.e., $(P_{t-m}, P_{t-m+1}, ..., P_t)$, the model predicts the agent position in the next time step P_{t+1}. Next, the observation buffer is updated by adding the most recent predicted point P_{t+1} to the end of the time-ordered buffer, and removing the first element in the buffer, rendering the new buffer $(P_{t-m+2}, ..., P_t, P_{t+1})$. With the updated buffer, the position of the agent for the next time step is then predicted. By repeatedly updating the prediction buffer n times, a predicted trajectory with length n is obtained. Using this form of sequential prediction, the prediction horizon can be controlled freely, which is desirable for real-world applications, as it allows users to change the prediction horizon without having to change the model architecture and re-train it.

The common model architecture is trained on agent-specific datasets, so as to learn agent-specific models. Prior to training, the model architecture is specified with a set of hyper parameters, including training optimizer, learning rate, etc. Next, this common architecture is trained on pedestrian, cyclist, and vehicle trajectory datasets separately, resulting in agent-specific models.

4.3.4 Model Evaluation

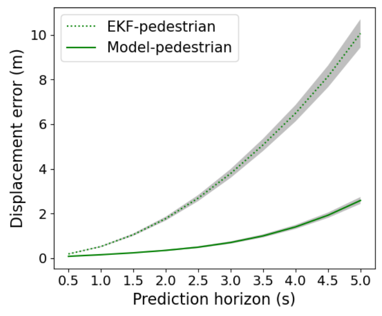
I benchmark the model against EKF, which is a wildly adopted state estimation method, and a popular approach in trajectory prediction. I use two evaluation metrics: Average displacement error (ADE) and final displacement error (FDE). ADE is computed as the average displacement error of all predicted time steps of a trajectory, averaged across all predicted trajectories. FDE is the displacement error of the last time step of a predicted trajectory, averaged across all predicted trajectories.

I use a velocity-based motion model for the EKF benchmark method, detailed in Algorithm 1.

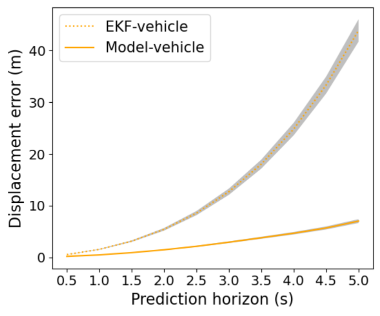
Algorithm 1 EKF algorithm for trajectory prediction

```
Result: Predicted trajectory sequence S with length $n$
1  Require: $\mu_{t-1}$, $\Sigma_{t-1}$, $u_t$, $e_t$, $H$, $R$, $Q$
2  Initialize result sequence S = $\emptyset$, round=1
3  Repeat step 1 to step 8 below $m$+$n$ times:
4  $\overline{\mu}_t = \mu_{t-1} + \begin{pmatrix} -\frac{v_t}{\omega_t} sin\mu_{t-1,\theta} + \frac{v_t}{\omega_t} sin(\mu_{t-1,\theta} + \omega_t\Delta_t) \\ \frac{v_t}{\omega_t} cos\mu_{t-1,\theta} - \frac{v_t}{\omega_t} cos(\mu_{t-1,\theta} + \omega_t\Delta_t) \\ \omega_t\Delta_t \end{pmatrix}$
5      $\overline{e}_t = h(\overline{\mu}_t)$
6  $G_t = \begin{pmatrix} 1 & 0 & \frac{v_t}{\omega_t} cos\mu_{t-1,\theta} - \frac{v_t}{\omega_t} cos(\mu_{t-1,\theta} + \omega_t\Delta_t) \\ 0 & 1 & \frac{v_t}{\omega_t} sin\mu_{t-1,\theta} - \frac{v_t}{\omega_t} sin(\mu_{t-1,\theta} + \omega_t\Delta_t) \\ 0 & 0 & 1 \end{pmatrix}$ $\overline{\Sigma}_t = G_t\Sigma_{t-1}G_t^T + R_t$
7  $K_t = \overline{\Sigma}_t H_t^T (H_t\overline{\Sigma}_t H_t^T + Q_t)^{-1}$
8  $\mu_t = \overline{\mu}_t + K_t(e_t - \overline{e}_t)$
9  $\Sigma_t = (I - K_t H_t)\overline{\Sigma}_t$
10 If round $> m$ , add $\mu_t$ to S, round = round +1
11 $\mu_{t-1} = \mu_t$, $\Sigma_{t-1} = \Sigma_t$
```

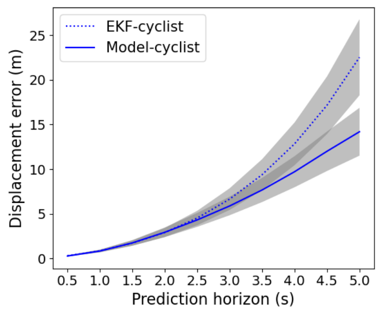
In Algorithm 1, θ is the angular orientation, $u_t = (v_t, \omega_t)^T$, where v_t is the velocity and and ω_t is rotation velocity at time step t. Both v_t and ω_t can be calculated based on the history trajectory. R and Q are pre-defined noise matrices on the observation and motion models, respectively. e is the observation model defined as $e = h(\mathcal{X}) + Q$.



(a) Pedestrian trajectory prediction

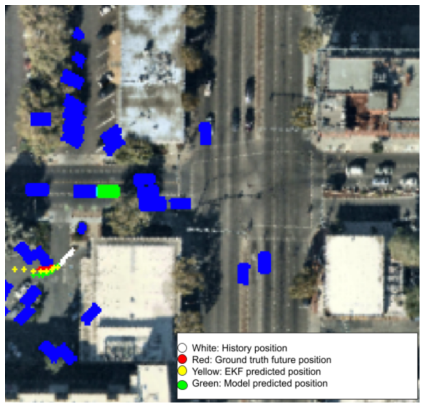


(b) Vehicle trajectory prediction

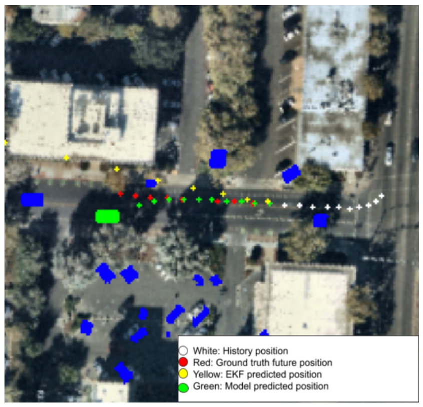


(c) Cyclist trajectory prediction

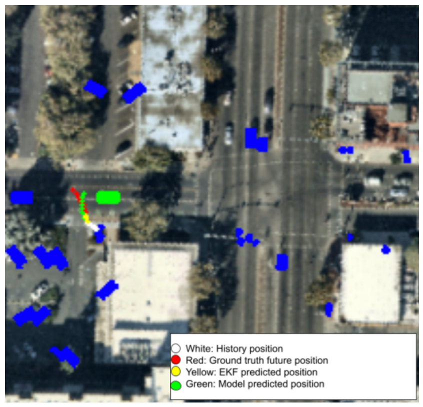
Figure 4.1: Comparison of the displacement errors for all agent types


White: History position
Red: Ground truth future position
Yellow: EKF predicted position
Green: Model predicted position

(a) A pedestrian trajectory prediction example


White: History position
Red: Ground truth future position
Yellow: EKF predicted position
Green: Model predicted position

(b) A vehicle trajectory prediction example


White: History position
Red: Ground truth future position
Yellow: EKF predicted position
Green: Model predicted position

(c) A cyclist trajectory prediction example

Figure 4.2: Visualization of predicted trajectories

4.4 Experimental Results

4.4.1 Result Performance

The proposed approach is evaluated on the Lyft L5 dataset (Houston et al. (2020)), which provides trajectories of different traffic agents collected by AVs through their on-board sensors. In this dataset, based on AV sensor detection results, traffic agents are labeled as pedestrians, vehicles, or cyclists. All information in the dataset is time-ordered and stored in three different lists: scenes, frames, and agents. Each frame contains information regarding the position of the ego vehicle and the indices of the agents in the frame. Each agent information contains its position, orientation, agent specific ID, and agent type.

Based on the dataset, I extracted and constructed agent-specific datasets of pedestrians,

Table 4.1: Overall benchmark performance. The values in each cell show the ADE/FDE metrics in meters.

Method	Pedestrian	Vehicle	Cyclist
EKF	3.98/10.06	15.21/43.87	7.89/22.52
This work	**0.89/ 2.58**	**2.95/7.03**	**5.96/14.18**

cyclists and vehicles with frequency of 2 Hz (0.5 seconds per position point), which is a general frequency setting in the existing literature (Cheng and Sester (2018); Rasouli et al. (2019)). 5-second history trajectories were used for training, with the goal of predicting the trajectory in next 5 seconds. The dataset was shuffled and separated into a training set and a test set. In the training phase, each trajectory containing 11 points or more (5.5 seconds) was separated through a sliding window to generate more unique trajectories, while trajectories containing less than 11 points were discarded. In the prediction and evaluation phases, 20 points (10 seconds) were extracted for each trajectory, where the model used the initial 5 seconds as inputs, and predicted the next 5 seconds sequentially.After pre-processing, the vehicle, pedestrian, and cyclist training sets includes 450,000, 191,094, and 31,813 samples, respectively. The ADE and FDE evaluation metrics obtained based on the predictions by the proposed method and the benchmark EKF are presented in Table 4.1. Each entry in the Table 4.1 shows the metrics ADE/FDE in meters.

The results in Table 4.1 indicates that the proposed approach has good performance and outperforms the benchmark EKF method. Based on results of the proposed approach, the ADE of the pedestrian trajectory prediction is around a single step length of an adult (0.71m to 1.49m), the ADE of the vehicle trajectory prediction is less than the average vehicle length (around 4.20m), and the ADE of the cyclist trajectory prediction is about 3 times of the average bicycle length (around 1.70m). The lower performance of the cyclist model compared to the vehicle and pedestrian models can be attributed to its smaller training sample size. The performance of the proposed model against the EKF benchmark during the prediction horizon is demonstrated in Figure 4.1, where the EKF predictions are shown using dashed lines and predictions of the proposed approach are plotted in solid lines. The gray regions indicate the 95% confidence intervals of predictions over the prediction horizon. This figure demonstrates that, not surprisingly, in both models, the displacement error increases with time. However, the rate of increase in the displacement error of the proposed model is significantly lower that that of the EKF model, resulting in the gap between the displacement errors of the two models to grow super-linearly with time. I random selected a pedestrian, a vehicle, and a cyclist agent, and illustrated their corresponding prediction results in Figure 4.2. These figures clearly show the background contexts in which the agents are active. The autonomous vehicle whose sensors have provided the data is highlighted with a green box.

Figures 4.1 and 4.2 demonstrate that EKF tends to have lower prediction errors when the prediction horizon is short, e.g., within the first 1 to 2 seconds. However, the EKF prediction error can grow exponentially as prediction horizon increases. On the contrary, with the proposed approach, the rate of increase in error is much slower.

4.4.2 Transfer Learning Performance

The performance of deep learning tasks can be related to the size of the available training datasets. As aforementioned, the size of the cyclist trajectory dataset is small as it is a less commonly-used vehicle in the real world, compared to pedestrians and vehicles. A main benefit of the proposed work is that, as a result of using a common model architecture for all agent types, transfer learning can be adopted to improve the prediction performance of cyclists. S. Pan et al. mentioned that transfer learning can be conducted if the domains of the two tasks are different, but share some similarities (Pan and Yang (2009)). Here, I investigate the transfer of the learned knowledge from vehicles and pedestrians to cyclists. The transfer learning task fits the domain of inductive transfer learning, where the target task (learning cyclist trajectory patterns) is different from the source task (learning pedestrian/vehicle trajectory patterns), no matter when the source and target domains are the same or not. The transfer learning results are shown in Figure 4.3. Conducting transfer learning from the pedestrian (vehicle) model results in the ADE/FDE for the 5-seconds cyclist trajectory prediction to reduce from 5.96/14.18m to 1.43/3.74m (2.86/7.13m).

An observation from these results is that, although the vehicle trajectory dataset is larger than the pedestrian set, it does not produce the best transfer learning performance for cyclist trajectory prediction. The reason is that performance of transfer learning is also affected by the similarity between the source and target tasks (Pan and Yang (2009)). Better

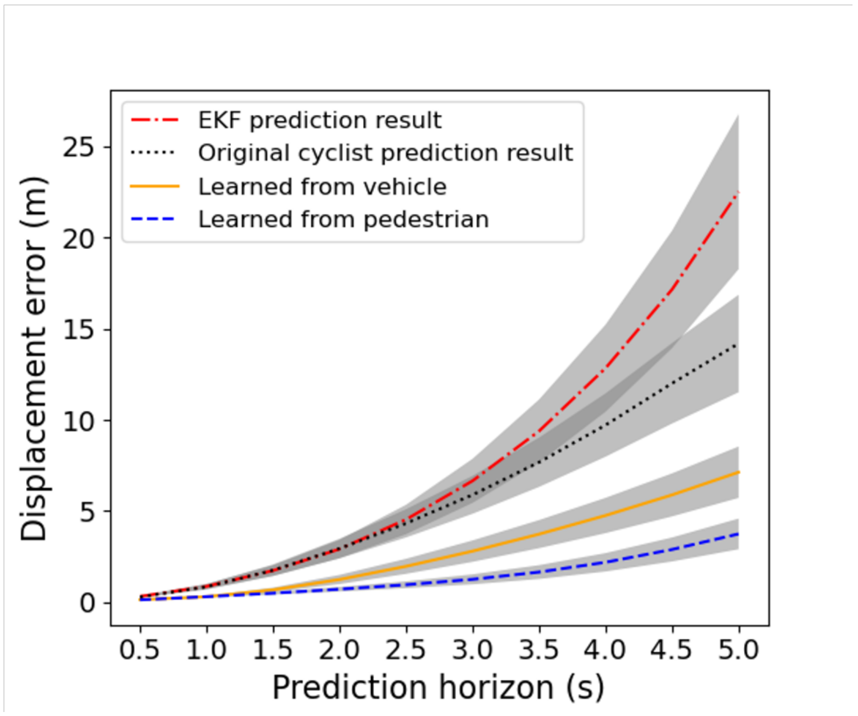


Figure 4.3: Transfer learning results

transfer learning results can be obtained with higher similarity between source and target tasks. To measure such similarity and to investigate its corresponding impact, the Kullback-Leibler divergence (KL divergence) (Mathiassen et al. (2002)) is introduced and computed between the distribution of pedestrian–cyclist task and that of the vehicle–cyclist task. A smaller KL-divergence value indicates a higher similarity between two distributions. I use the probability distribution of the distance between two adjacent positions for each agent type to represent them. This distribution captures the moving velocity of agents, which is one of the most basic patterns of behaviour attributed to traffic agents. Resulting distributions (fitted with Gaussian distribution) are shown in Figure 4.4. Based on these distributions, the KL-divergence of pedestrian-cyclist task is 51.15, while that of the vehicle-cyclist task is 127.76. These scores explain the performance difference of transfer learning results, i.e., the pedestrian-cyclist transfer learning task is more desirable as the source and target tasks show higher similarity.

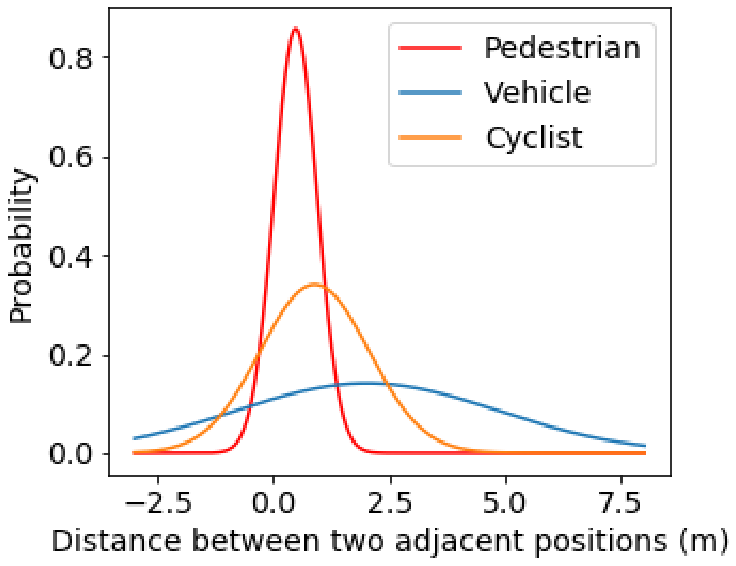


Figure 4.4: Probability distribution of three type of agents

4.5 Conclusion

In this work I utilize a learning-based approach to predict trajectories of pedestrians, vehicles, and cyclists in rather long horizons using a unified architecture. The proposed approach does not construct different model architectures to develop agent-specific models. Instead, it generates agent-specific models by training a common architecture over agent-specific datasets, rendering it more desirable for practical applications. I train the common architecture on three agent-specific datasets, namely, pedestrians, vehicles and cyclists, and compare it against an EKF benchmark model. Experimental results indicate that the proposed approach outperforms the EKF benchmark in longer-horizon 5-sec periods in terms of both average and final displacement errors. The generic architecture also allows for realizing the benefits of transfer learning. I demonstrate that by transferring knowledge from the pedestrian and vehicle datasets, which are larger in size, the prediction accuracy on cyclists can be significantly improved.

CHAPTER V

Predictive Trajectory Planning for Vehicles at Intersections with Deep Reinforcement Learning

5.1 Introduction

The last decade has been a witness to rapid development of autonomous driving technology. By leveraging breakthroughs in sensor technology and artificial intelligence, AVs are expected to perceive their surrounding environment, such as other road users and obstacles, and make driving decisions (Claussmann et al. (2019); Jo et al. (2015); Ma et al. (2015)). AVs have the potential to reduce human driving errors (van Wyk et al. (2020)), lower motor vehicle crash rates (Zhao et al. (2021); Geary and Danks (2019)), facilitate community-based mobility services (Hasan and Van Hentenryck (2021); Masoud and Jayakrishnan (2017); Zhang et al. (2020)), and increase fuel efficiency (Fagnant and Kockelman (2015); Xiangguo Liu (2021); Liu et al. (2022); Wu et al. (2019)), among other benefits. AVs are also expected to reduce traffic congestion and travel times (Zhang et al. (2020)).

In order to operate successfully in real world scenarios, AVs rely on three key technologies–perception, prediction, and planning (Raju et al. (2019); Sadat et al. (2020)). The purpose of perception is to help AVs understand their surrounding environment. The prediction step helps AVs forecast future movements and trajectories of their surrounding agents, including vehicles, pedestrians, cyclists, etc. Accurate prediction enables AVs to take more informed actions for the sake of safer driving. Finally, the planning module serves as the decision maker for an AV's future motion. A planner generates driving strategies based on the information collected and the knowledge gathered by the AV in the perception and prediction steps.

Perceiving the environment and forecasting the future movements of traffic agents allows AVs to create robust and safe planning strategies. However, due to the complexities involved in navigating a dynamic traffic environment and other changing factors, such as weather and road infrastructure conditions, the surrounding environment is only partially observable to AVs (van Wyk et al. (2020); Toghi et al. (2021); Hubmann et al. (2018); Zhang and Masoud (2021)). AVs are anticipated to be able to drive without human intervention, prompting safety to be considered as the top priority by users, manufactures, and policy makers (Moody et al. (2020); Koopman and Wagner (2017)). In addition to avoiding crashes with their surrounding vehicles, it is critical for AVs to interact safely with other road users, such as pedestrians and cyclists (Rasouli and Tsotsos (2019); Pammer et al. (2021)). One important reason is that these road users may be more vulnerable compared to airbag-protected drivers if involved in traffic crashes (Rahman et al. (2021); Das (2021); Penmetsa et al. (2019)). To investigate their reliability before scaling up the market, currently AVs are required to be tested in real world conditions, i.e., by accumulating as may driving miles as possible, so as to enable AV developers to observe, analyze, and improve their performance. Nonetheless, no matter how many miles are collected, uncertainty will still remain (Kalra and Paddock (2016)). Such opportunities and challenges make trajectory planning of AVs a challenging, yet essential problem, as planning policies directly determine how AVs behave.

In this chapter, I model the trajectory planning problem as a partially observable Markov decision process (POMDP), and develop a reinforcement learning (RL)-based approach to find a driving policy. This driving policy dictates acceleration and heading profiles that the subject AV should follow for several seconds into the future. The devised policy accounts for the predicted state of the surrounding traffic agents (e.g., vehicles, pedestrians, and cyclists). This predicted state includes trajectory prediction and driving risk prediction of surrounding road users. A deep Q learning algorithm is developed in the framework to learn from abstract graphs, which are constructed to represent surrounding traffic dynamics. In order to obtain enough driving scenarios for training purposes, a

Bayesian Gaussian mixture model and Gibbs sampling-based approach is proposed to simulate *socially acceptable* driving scenarios, which contain the subject AV, surrounding vehicles, pedestrians and cyclists. Different from the existing work in the literature, which mainly focused on planning for simple driving scenarios, e.g., highway driving or driving on straight paths, this work focuses on one of the most complex driving scenarios– non-signalized urban intersections. The proposed work is also more general in terms of planning task settings. While the existing literature mostly focuses on a specific task, such as avoiding obstacles when driving on a straight path, the proposed approach is not limited to a certain type of future movements of the subject AV, i.e., going straight, turning left or turning right. Rather, it can accomplish the planning task under a set of future goals for the subject AV. The framework learns an offline policy that prescribes the acceleration and heading profile of the AV in the next few several seconds. When implemented in real time, the proposed approach utilizes the learned policy for online planning and guides the subject AV to follow a path that is safe, comfortable and more energy efficient.

The rest of the chapter is arranged as follows: First I present the related work on trajectory prediction, driving risk prediction, AV trajectory planning, and RL. Next, I introduce the methodology and elaborate on the individual components of the framework. After that, I evaluate the proposed work using different driving experiments and compare the performance of the proposed method to that of human driving. At last I conclude the chapter.

5.2 Related Work

The proposed approach consists of three main components: trajectory prediction, driving risk prediction, and trajectory planning. In this section, I discuss related work regarding these topics as well as RL, which is the methodology I use for trajectory planning.

5.2.1 Trajectory Prediction

In the past decade, with the development of AV technology, traffic agent trajectory prediction has become an increasingly important topic because of its potential in reducing traffic accidents such as collisions (Rudenko et al. (2020); Ma et al. (2019)). With accurate trajectory prediction of traffic agents, AVs can have a clear picture of the state of the environment in the near future, which will inform their decision making. Due to the heterogeneity between different types of road users, trajectory prediction settings and models are usually different in order to capture different motion patterns in different road user groups (Zhang et al. (2021)). In this work, I embed the planning framework with trajectory prediction of three major types of road users–vehicles, pedestrians, and cyclists.

In recent years, several studies have focused on vehicle trajectory prediction. Deo and Trivedi (2018a) proposed a long short-term memory (LSTM) method with social pooling to predict future vehicle trajectories. They used a pooling layer to capture the interactions between vehicles, and showed promising performance on public datasets. Kim et al. (2017) proposed a recurrent neural network (RNN)-based model to conduct vehicle trajectory prediction. Their proposed method was mainly based on LSTM-RNN, and generated probabilities of future vehicle locations in a grid map. They showed that with a large amount training data the proposed method could learn dynamics of vehicles. Houenou et al. (2013) predicted future vehicle trajectories by combining advantages of the constant yaw rate and acceleration (CYRA) model and maneuver recognition. However, the method has several rigid assumptions, such as a constant road curvature. Ding and Shen (2019) proposed an online prediction framework for vehicle trajectory prediction in urban scenarios. It first anticipated a vehicle's driving policy, e.g., forward, turning left/right, etc., and then conducted optimization-based context reasoning to predict future trajectories. They demonstrated that the method had good performance in different road traffic configurations, such as curvy roads and intersections.

Different from trajectory prediction for vehicles, the trajectory prediction problem for pedestrians can be more complex, mainly because of the complicated nature of the human walking behavior. Xue et al. (2018) proposed a deep learning-based approach, called SS-LSTM, to capture patterns in pedestrian walking scenarios, which were used for trajectory prediction. Their approach showed good performance on public datasets and could predict acceptable trajectories in different scenarios. Based on encoding of crowd interaction, Xu et al. (2018) developed a method to accurately predict pedestrian trajectories.

Instead of treating all surrounding pedestrians as equal, they assigned different importance weights to different surrounding pedestrians. Alahi et al. (2016) used an RNN-based strategy, called social LSTM, to predict pedestrian trajectories. They introduced a "social pooling" mechanism in the model to learn the interactions between different pedestrians, and were able to provide accurate predictions in long prediction horizons. Gupta et al. (2018) proposed a general adversarial network (GAN)-based method, called social GAN, to predict pedestrian trajectories. Instead of predicting one target trajectory, it predicts a number of socially acceptable future trajectories of the subject pedestrian.

With the rising interests and concerns about cyclists' safety, trajectory prediction for cyclists has been gaining increasingly more attention. Zernetsch et al. (2016) studied cyclist trajectory prediction with both a physical model and a multi-layer perceptron model. Both methods showed good performance within a short prediction horizon. Huang et al. (2021) proposed an LSTM-based model by introducing a focal attention mechanism. They indicated the method could accurately predict 2-3 seconds cyclist trajectories. Pool et al. (2017) used road topology as prior information and proposed a cyclist trajectory prediction method. They demonstrated that their model could provide higher accuracy compared to benchmarks when predicting one second future cyclist trajectories.

I conducted trajectory prediction research in the Chapter III (Zhang et al. (2022)), in which I developed a trajectory prediction algorithm called "step attention". In contrast to the existing work requiring different information from the surrounding traffic environment, step attention used only history trajectory information. When predicting a pedestrian's trajectory, step attention outperformed state-of-the-art algorithms including the aforementioned social LSTM and social GAN, and demonstrated promising performance when predicting long look-ahead pedestrian trajectories in real-world urban intersections. I also showed that after being trained on a vehicle trajectory dataset, step attention was able to accurately predict 5 seconds vehicle trajectories (Zhang et al. (2021)). Cyclists are usually less seen in transportation systems compared to vehicles and pedestrians, which can result in fewer training data. However, I showed that using transfer learning, step attention can also accurately predict cyclist trajectories within a 5 seconds prediction horizon.

5.2.2 Driving Risk Prediction

Studies show that more than 1.3 million people die worldwide every year because of traffic accidents (Ali et al. (2021)). When vehicles are driving in transportation systems, they can pose risks to themselves and their surrounding traffic agents. Shi et al. (2019) proposed a XGBoost-based approach to assess driving behavior and predict risk levels. It utilized fuzzy c-means to label risks and conducted crash risk prediction. Wang et al. (2017) proposed a machine learning-based approach to predict driving risk. It used supervised learning methods on a pre-labeled dataset to obtain insights on human driving behavior. However, it was not able to predict future driving behaviors as it mainly conducted classification based on different driving-related information. Xu et al. (2022) demonstrated a risky driving prediction method with panel data. They used a deep learning framework to capture features in raw driving data, and predicted if a driving behavior is aggressive or not. Wang et al. (2010) proposed a method for driving risk/danger prediction based on a variety of features, e.g., vehicle dynamics, driver psychological data, etc. Using an RL-based algorithm, they determined driving danger levels. Different from existing work, in the previous work in Chapter II (Zhang et al. (2022, 2021)) I proposed an unsupervised learning based approach to predict driving risks. It uses clustering with dynamic time warping (DTW) as the similarity measure to learn driving patterns from raw vehicle trajectory data, and an anomaly detection method to obtain driving risk probabilities for the future several seconds in real-time. Unlike the majority of work in the literature, it does not require manual labeling, which avoids labeling bias introduced by human subjectiveness, and ensures scalability when deploying in real world systems. Instead of providing binary prediction results, the model quantifies driving risks and provides probabilistic outputs, which can be more informative in real-world applications.

5.2.3 Vehicle Trajectory Planning

Vehicle planning can be roughly categorized into three types: route planning, maneuver planning, and trajectory planning (Zhang et al. (2013)). Route planning provides the general destination information to the subject vehicle, e.g., which route to take when traveling from one city to another. Maneuver planning tells the subject vehicle what maneuver to take during driving, such as changing lanes, turning left, etc. Finally, trajectory planning generates strategies directly related to vehicle dynamics, such as acceleration and heading. In this work, I focus on trajectory planning. Trajectory planning is also one of the most challenging tasks for AVs, as it directly impacts a vehicle's driving behavior, driving comfort, energy efficiency, etc.

Zhang et al. (2013) proposed a dynamic trajectory planning method by decomposing maneuvers into simple sub-maneuvers. With a two-fold optimization-based approach, it could find optimal steering angles and showed good trajectory planning performance in terms of driving comfort. Ntousakis et al. (2016) proposed an optimal control-based method to plan vehicle trajectories in a simple highway merging scenario. It utilized a model predictive control (MPC) scheme to consider possible traffic disturbances. Simulation results demonstrated validity and applicability of the method. Zhou et al. (2018) proposed a recursive planning framework to plan vehicle trajectories in simple on-ramp merge scenarios. Similarly, they modeled the problem using an optimal control approach and demonstrated its effectiveness. However, as indicated by the authors, the planning horizon of the method is time-varying, which can be affected by the state of traffic. Burger and Lauer (2018) proposed a mixed integer quadratic programming-based approach to plan multiple vehicle trajectories. It minimized the cost brought by road geometry, collisions, etc., and showed feasibility and benefits with simulated data compared to a priority-based method.

With the advancements of machine learning and artificial intelligence in the past decades, deep learning-based vehicle trajectory planning has been gaining more attention, as it shows robust performance in dynamic traffic environments (Aradi (2020); Ye et al. (2021)). Cai et al. (2019) introduced a trajectory planning approach based on imitation learning. It separated planning into three tasks: going straight, turning left, and turning right, and created a different model for each task. Authors indicated the method was able to plan collision-free future trajectories. Song et al. (2018) proposed a deep learning-based method to determine motion parameters. Directly based on image inputs, it can learn from humans'' driving behaviors, and plans vehicle motions that adapt to different roads.

Zhu et al. (2018) used a RL-based approach to plan vehicle trajectories in car-following scenarios. It used historical trajectory data to train the model in order to find an optimal inter-vehicle spacing strategy. Josef and Degani (2020) demonstrated an online RL-based method to conduct safe local planning for vehicles in an unknown terrain. By formulating the surrounding environment with a Markov decision process (MDP) and using a dense reward signal, it learned an optimal planning policy with the help of a constructed deep neural network. According to their experimental results, the planned trajectory can capture the vehicle"s dynamics and its interaction with terrains, and avoid obstacles.

5.2.4 Reinforcement Learning

RL can be viewed as a solution methodology for sequential decision making problems (e.g., POMDPs), in which the state and action spaces are too large to warrant the use of exact methods. RL relies on a delayed reward signal to find a control/action policy. When learning, an agent takes actions to interact with its environment based on its current policy, and obtains a reward. The policy is then updated based on the reward signal. The process repeats until the policy converges. RL has been demonstrated as a robust tool in decision making, and has shown promising performance in many fields such as robotic control (Recht (2019); Arulkumaran et al. (2017)), signal processing (Zhang and Masoud (2020)), traffic signal control (Mannion et al. (2016)), congestion pricing (Pandey et al. (2020)), etc. This field has been further promoted in recent years with a number of deep learning-based algorithms.

In this chapter, I propose a predictive trajectory planning framework based on deep Q learning (Mnih et al. (2015)). Different from some of the existing applications of RL, in this chapter I focus on more complex traffic environments, i.e, urban intersections. I use graphs to represent dynamic traffic scenarios, and formulate the problem as a POMDP. The model is trained offline to find a trajectory planning policy for the next several seconds given an observation at a given time step. The proposed method is not limited to finding a trajectory planning policy for a specific task, e.g., through movement, which is usually the case in the existing literature. I evaluate the learned trajectory planning strategy in terms of safety, driving comfort (trajectory jerkiness), and energy efficiency.

The main contributions of this chapter are as follows:

- A predictive-prescriptive framework is proposed to conduct long look-ahead vehicle trajectory planning, while integrating future trajectories of surrounding agents and driving risks of surrounding vehicles;
- A graph-based representation is proposed to capture dynamic traffic environment;
- A Bayesian Gaussian mixture model-based scenario encoding and a Gibbs sampling-based scenario simulation approach are proposed to generate training samples under the guidance of real-world driving scenarios;
- An RL-based framework, including a POMDP formulation, reward design, a deep Q network and the training strategy, is proposed to learn planning policies. The resulting planning policy is demonstrated to enhance safety, comfort, and energy efficiency compared to human driving.

5.3 Methodologies

In this section I elaborate on different components of the proposed framework, including scenario simulation, predictive models, the POMDP that models the system, and its accompanying RL solution methodology. An overview of the methodology section is provided in Figure 5.1.

5.3.1 Problem Statement

The purpose of this work is to plan the future trajectory of the subject vehicle for several seconds in complex and dynamic traffic environments, and in the presence of different road users–vehicles, pedestrians, and cyclists. The proposed trajectory planning framework does not require prior knowledge on the vehicle's intention (e.g., through movement, left turn, right turn, etc.). More precisely, the framework is trained on the history trajectory of the subject vehicle as well as predicted trajectories of surrounding agents, and outputs an acceleration and heading profile for the subject vehicle to follow within the next several seconds.

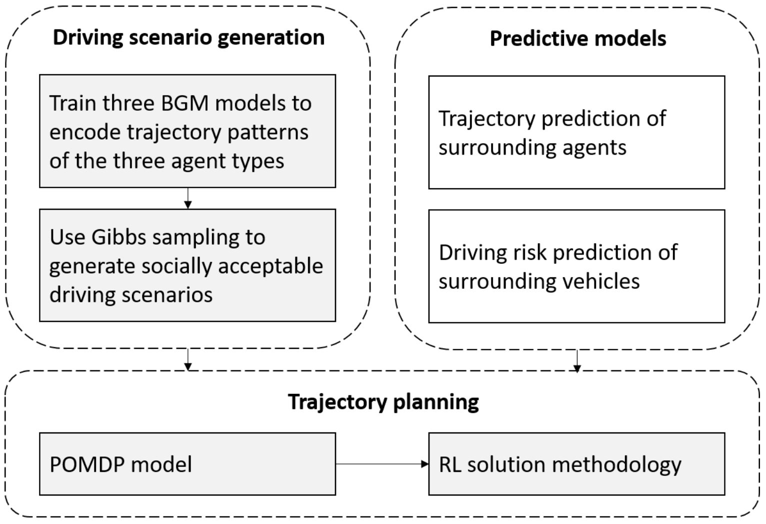

Driving scenario generation
Train three BGM models to encode trajectory patterns of the three agent types
Use Gibbs sampling to generate socially acceptable driving scenarios
Predictive models
Trajectory prediction of surrounding agents
Driving risk prediction of surrounding vehicles
Trajectory planning
POMDP model
RL solution methodology

Figure 5.1: An overview of the proposed methodology. Shaded boxes are the unique contributions of this chapter.

Consider a vehicle traveling on a road network at time t. Let us denote the history trajectory of the subject vehicle by $T_s^t = (X_{t-m+1}^s, X_{t-m+2}^s, ..., X_t^s)$, where $X_t^s = (x_t, y_t)$ denotes the geo-coordinates of the subject vehicle at time step t, m is the trace-back history in the number of time steps, and $x_t \times y_t \in \mathbb{R}^2, \forall t$. At time step t the subject vehicle observes its surrounding traffic agents. Let $T_v^t = \cup_{i=1}^k T_{vi}^t$ be the set of observed history trajectories of the vehicles in the surrounding environment of the subject vehicle, where k is the number of surrounding vehicles, and $T_{vi}^t = (X_{t-m+1}^{vi}, X_{t-m+2}^{vi}, ..., X_t^{vi})$ denotes the history trajectory of the i^{th} surrounding vehicle with a trace-back history of length m. Similarly, sets of observed history trajectories of pedestrians and cyclists can be denoted as $T_p^t = \cup_{i=1}^h T_{pi}^t$ and $T_c^t = \cup_{i=1}^q T_{ci}^t$, respectively, where h denotes the number of pedestrians and q denotes the number of cyclists present within scene at time t. The priori of the problem at time step t is expressed as $\text{Priori}_t = (T_s^t, T_v^t, T_p^t, T_c^t, M)$, where M denotes the map. Based on Priori_t and given a planning horizon n, the purpose of this work is to provide the subject vehicle with an acceleration profile and a heading profile (future driving strategies), i.e., $\text{Output} = (P_1, P_2, ..., P_j, ..., P_n), P_j = (a_j, \theta_j)$, for the subject vehicle to follow in next n time steps, so as to achieve better driving safety, driving comfort, and fuel efficiency. Here, a_j and θ_j are planned acceleration and heading values at a future time step j, respectively.

5.3.2 Scenario Generation

Trajectory planning approaches need to be developed and validated in realistic driving scenarios, either in a real-world environment or under realistic simulated driving conditions. I define a driving scenario (or scenario for short) as a 10-seconds long realization of the transportation network. A scenario includes (1) a map of the transportation facility of interest, including lane boundaries, and (2) all agents who are present in the transportation facility of interest within the specified 10 seconds, either for the entity of the 10 seconds or a part of it. In this study, I will generate all scenarios in experimented intersections.

A noticeable feature of deep learning models is their need for a large volume of training examples to guarantee generalizability of the final trained model. There exist a few publicly available driving datasets that provide samples of driving scenarios, but the number of scenarios available in such datasets are severely limited. In order to train the deep learning model with a sufficient number of training samples, in this work I propose a driving scenario generation method that utilizes a Bayesian Gaussian mixture model (BGM) and Gibbs sampling to generate realistic driving scenarios.

Simulated driving scenarios have to satisfy two conditions: (1) at the individual agent trajectory level, all agent trajectories in the scenario should be reasonable in terms of their behavior:

- For vehicles, acceleration/deceleration should be within a reasonable range to have trajectories that are kinematically feasible;
- The movement of vehicles should be confined within lane boundaries;
- The step length of pedestrians should be reasonable, ranging from 0.71 m (walking) to 1.49 m (running). The time length of a step should be between 0.33 s to 0.53 s (Zhang et al. (2022));
- Cyclists should move slower than vehicles, but faster than pedestrians.

(2) at the scenario level, simulated scenarios should resemble real-world traffic scenarios and be *socially acceptable*, i.e., they should follow rules that are naturally followed in real traffic environments, e.g., rational agents who try to avoid collisions, etc.

To satisfy these two conditions and simulate reasonable driving scenarios, I present a hybrid simulation method. It simulates driving scenarios based on a real-world driving scenario dataset collected by a fleet of autonomous vehicles. In order to have the trajectories in the simulated driving scenarios follow the real behavior patterns of real-world agents, first I collect trajectories of different agent types from a wide range of driving scenes from an autonomous fleet driving dataset $D = \{D_v, D_p, D_c, M\}$, where D_v, D_p, and D_c denote datasets that contain vehicle, pedestrian, and cyclist trajectories, respectively, and M denotes the map, which contains information on the transportation infrastructure, including the geometric features of the network at the scene where data is collected. Next, I reconstruct driving scenarios using the collected real-world trajectories under the second condition. The details pertaining how to construct and simulate driving scenarios are provided in the following sections.

5.3.2.1 Bayesian Gaussian Mixture Model

The simulation of driving scenarios can be considered as a dataset augmentation process. A simulated driving scenario consists of four components: 1) vehicle trajectories $d_v \subset D_v$; 2) pedestrian trajectories $d_p \subset D_p$; 3) cyclist trajectories $d_c \subset D_c$; and 4) a map. In the scenario generation process in this study, the map information, i.e., lane boundaries, is static and consistent across all driving scenarios. Therefore, a driving scenario is uniquely determined by d_v, d_p and d_c. Let us represent a scenario as $\textbf{Scene} = [X, Y, Z]$, where X, Y, Z are encodings of d_v, d_p and d_c, respectively. The purpose of this encoding is to reshape, condense and extract patterns of trajectories with different agent types into more concise vectors that together construct the input space of the scenario generation model. Note that encodings are done at the scenario level, and all trajectories of the same agent type presented in a scenario correspond to a single encoding.

After extracting agent trajectories from raw scenarios, I use them to train three BGM models to capture the scenario pattern distributions. The scenario generation process learns one BGM model per agent type. As such, I denote by K_v, K_p, and K_c the number of components in the BGM models for vehicles, pedestrians, and cyclists, respectively. I use the phrase *scenario pack* to refer to all trajectories from an agent type in a given scenario. As the nature of the training process for all three BGM models is the same, in the following I elaborate on the training of a general BGM model with K components.

Unlike commonly used hard clustering methods such as K-means, BGM is a soft/probabilistic clustering approach (Patel and Kushwaha (2020)), which assumes samples are generated from a mixture of different Gaussian distributions with unknown parameters.

Assume that trajectories in a scenario pack are generated according to a random process that: 1) randomly selects one of K components from the mixture distribution $\pi = (\pi_1, ..., \pi_K)$, and 2) generates a pack of trajectories from the probability distribution of the selected component. The mixture model can be presented as:

$$P(X|\theta = \{\phi, \pi\}) = \sum_{k=1}^{K} \pi_k P(X|\phi_k), \tag{5.1}$$

where $\phi = (\phi_1, ..., \phi_K)$ includes the set of parameters associated with the K components in the mixture model. For a given component $k \in K$, $\phi_k = (\mu_k, \Sigma_k)$ with μ_k and Σ_k respectively denote the mean and the covariance matrix of the k^{th} Gaussian component, $P(X|\phi_k)$. During training, the unknown parameters $\theta = \{\phi, \pi\}$ are estimated using Bayesian estimation, as indicated in Eq. (5.2), based on the prior distribution in Eq. (5.3).

$$P(\theta|X) = \frac{P(\theta)P(X|\theta)}{\int P(\theta)P(X|\theta)\, d\theta} \tag{5.2}$$

$$\begin{aligned} \pi = (\pi_1, ..., \pi_K) &\sim Dir(\alpha) \\ \mu_1, ..., \mu_K &\sim \mathcal{N}(0, 1) \\ \Sigma_1, ..., \Sigma_K &\sim IW(\Phi, m) \end{aligned} \tag{5.3}$$

where $Dir(\alpha)$ is a Dirichlet distribution, α is a vector of positive real numbers, $\mathcal{N}(0, 1)$ is a standard Normal distribution, and IW is an Inverse-Wishart distribution, with Φ as the scale matrix and m as the degree of freedom.

The scenario generation algorithm is shown in Algorithm 2.

Algorithm 2 Scenario encoding and trajectory pool construction

```
Data: Real world driving scenario dataset $D$
Result: Scenario encodings; trajectory pools
1  Initialization: BGM training dataset $veh = \emptyset$, $ped = \emptyset$, $cyc = \emptyset$
2  foreach Scenario in D do
3      pack trajectories in a scenario
4      1. Extract agent trajectories $d_v$, $d_p$ and $d_c$ from the scenario: for each scenario, enumerate
        all vehicle, pedestrian, and cyclist trajectories in the order of the trajectory start times.
        Discard trajectories with the total trajectory length of less than 10 seconds;
5      2. Randomly choose up to 10 trajectories for each agent type;
6      3. Crop $d_v$, $d_p$ and $d_c$ into shape [10, $m$, 2] (if less than 10, pad 0s);
7      4. reshape $d_v$, $d_p$ and $d_c$ into [10, $m \times 2$];
8      5. Add reshaped results to sets $veh$, $ped$, $cyc$.
9  foreach dataset $df \in \{veh, ped, cyc\}$ do
10     1. Assume that vehicle, pedestrian and cyclist data in $df$ are generated by mixture
        models with $K_v, K_p$ and $K_v$ components;
11     2. Train a Bayesian Gaussian mixture model using dataset $df$;
12     3. Obtain component IDs (encodings) of all samples in $df$;
13     4. Collect all raw trajectories in each component to form a trajectory pool;
14     Output 1:Trajectory pools for component.
15     Encoding: for each scenario, use the three trained BGM models to find the component
        IDs of $d_v$, $d_p$ and $d_c$ and form a scenario encoding in the format [vehicle-pack
        component $i$, pedestrian-pack component $j$, cyclist-pack component $k$].
16     Output 2: Scenario encodings.
```

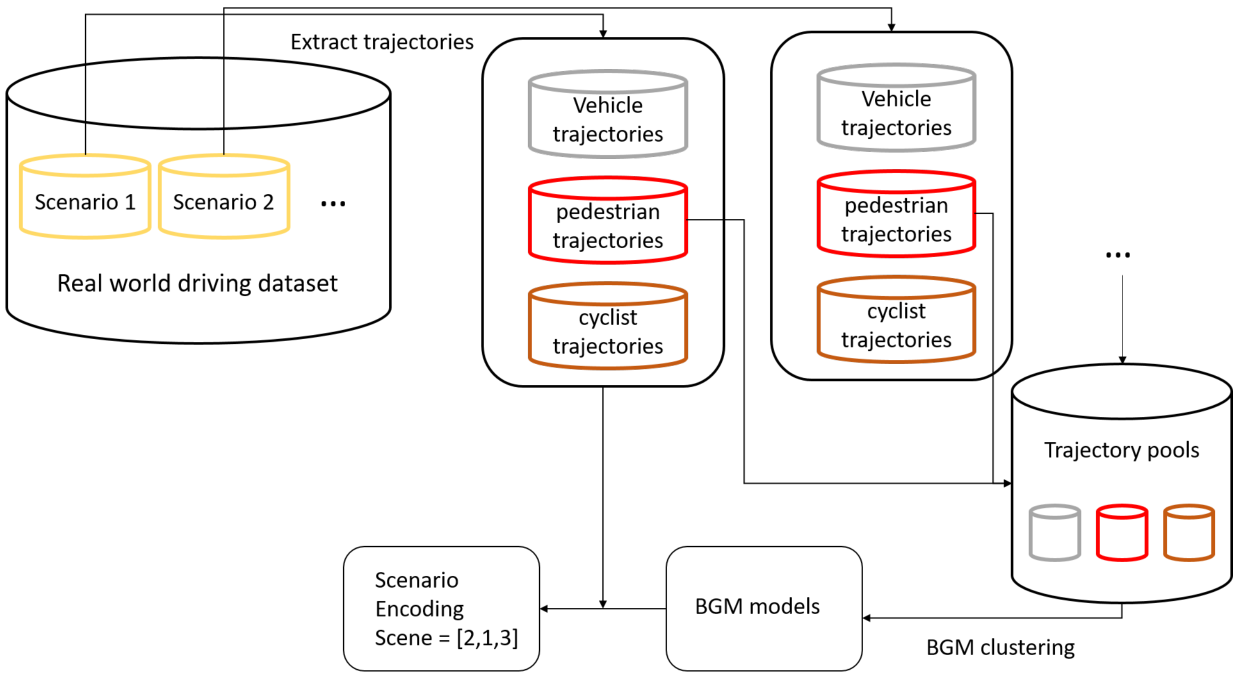


Figure 5.2: Scenario encoding workflow

The scenario encoding flow is shown in Figure 5.2, where the encoding process of an example scenario is illustrated.

5.3.2.2 Socially Acceptable Scenario Generation using Gibbs Sampling

In this chapter I develop a Gibbs sampling-based approach to generate a realistic dataset of driving scenarios. Gibbs sampling guarantees that the resulting dataset follows a similar distribution as the original real-world dataset in terms of patterns of movement by each agent type, as encoded in the previous step. In a real-world environment, trajectories generated by traffic agents are socially acceptable (Gupta et al. (2018)). Random sampling of trajectories to construct new scenarios may result in scenarios that are socially unacceptable. For example, randomly selecting vehicle trajectories to construct a scenario may result in crashes. In order to generate socially acceptable scenarios, once a candidate scenario is simulated, a cleaning step is conducted to remove agents with socially unacceptable behaviors.

In this work, I consider the patterns of vehicle trajectories to serve as the backbone of a simulated driving scenario; that is, Y and Z are affected by X. Due to the generally small

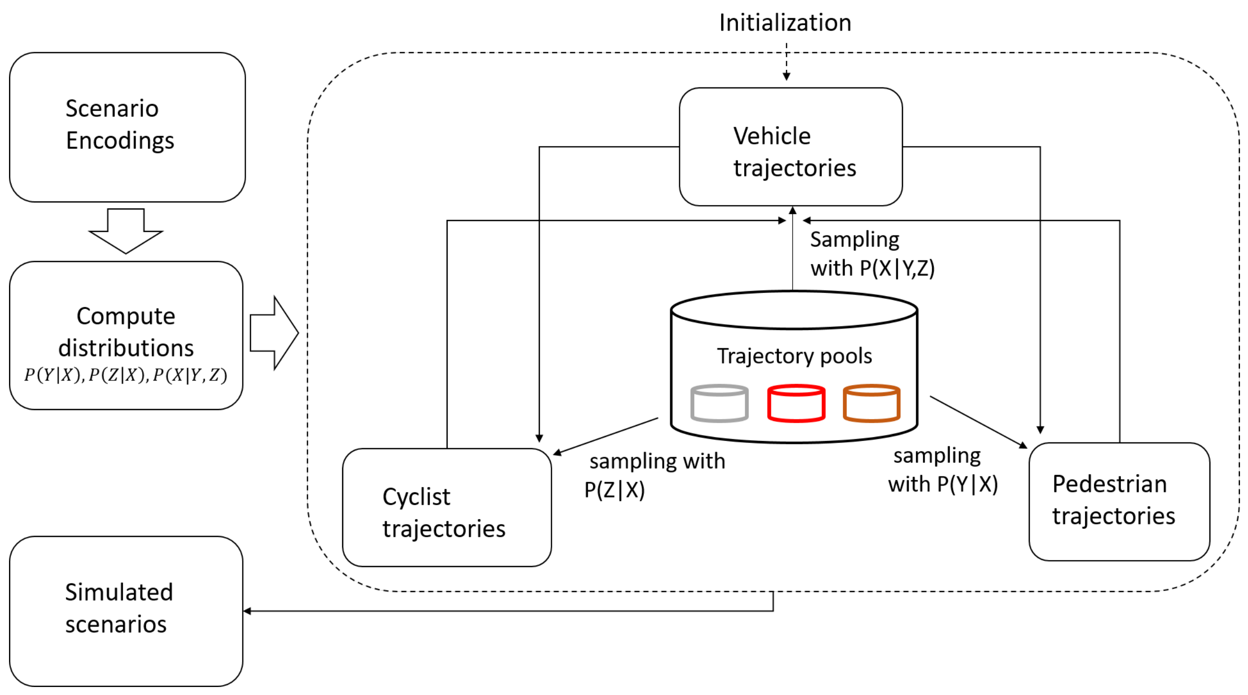


Figure 5.3: Scenario generation workflow

sample size of cyclist trajectories and the fact that areas in which cyclists ride rarely have conflicts with the areas in which pedestrians move, I assume Y and Z are independent of each other. With scenario encodings scene $= [X \in \{1, \cdots, k_V\}, Y \in \{1, \cdots, K_p\}, Z \in \{1, \cdots K_c\}]$ obtained using BGM models, the conditional distributions $P(Y|X)$, $P(Z|X)$, and $P(X|Y, Z)$ are then estimated. Note that X, Y, and Z are variables representing BGM component IDs for vehicles, pedestrians, and cyclists, respectively, defined in the previous section.

Let S_n^{sim} be the desired simulated dataset size, and a, b, and c be the maximum number of trajectories for vehicles, pedestrians, and cyclists in a simulated scenario, respectively. Denote the trained BGM model for vehicles as BGM_{veh}. The sampling algorithm is shown in Algorithm 3.

Note that in Algorithm 3, a simulated scenario affects the simulation of the next scenario. Once vehicle trajectories are sampled based on conditional probability distributions to form a scenario, it is used as the backbone to initialize the simulation of the next scenario recursively. The whole workflow for scenario simulation is depicted in Figure 5.3.

Algorithm 3 Socially acceptable scenario generation with Gibbs sampling

```
Data: $P(Y|X)$, $P(Z|X)$, $P(X|Y,Z)$; trajectory pools; $BGM_{veh}$
Result: Simulated scenario dataset
1  Initialization:
2  1. Simulated dataset $D_{\text{sim}} = \emptyset$;
3  2. Randomly select $a$ vehicle trajectories from the original dataset;
4  while $|D_{sim}| < S_n^{sim}$ do
5      1. Reshape vehicle trajectories to the $BGM_{veh}$ input shape;
6      2. Obtain the encoding $e_x$ on vehicle trajectories using $BGM_{veh}$;
7      3. Sample the pedestrian encoding category $e_y$ based on $P(Y|X)$ and $X = e_x$;
8      4. Sample the cyclist encoding category $e_z$ based on $P(Z|X)$ and $X = e_x$;
9      5. Sample the vehicle encoding category $e_x^{'}$ based on $P(X|Y,Z)$, $Y = e_y$ and $Z = e_z$;
10     6. Randomly sample $a$, $b$, and $c$ trajectories respectively from the vehicle, pedestrian,
         and cyclist trajectory pools with $X = e_x^{'}$, $Y = e_y$, $Z = e_z$;
11     7. Form a candidate scenario using sampled trajectories in step 6;
12     8. foreach Simulated time step in candidate scenario do
13         i). Remove vehicle trajectories that collide with other vehicle trajectories;
14         ii). Remove pedestrian trajectories that collides with the remaining trajectories from
             step i;
15         iii). Remove cyclist trajectories that collides with the resulting trajectories from steps
             i and ii.
16     9. Check if the remaining number of trajectories of each agent type is less than the
         corresponding desired minimum number in a scenario. If yes, discard current scenario,
         repeat steps 6-9 and generate another scenario;
17     10. Add simulated scenario to $D_{sim}$; continue to the next round with current vehicle
         trajectories.
18 Output: Simulated scenario dataset $D_{sim}$.
```

5.3.3 Predictive Models

By accurately predicting future states of surrounding traffic agents, the subject vehicle is able to conduct predictive planning. In this work I consider two kinds of predictive results that inform the decisions made by the subject vehicle: i) trajectory prediction of surrounding agents; and ii) driving risk prediction of surrounding vehicles. Prediction horizons for both predictive models are 5 seconds into the future. Since the predictive models are discussed from Chapter II through Chapter IV, in this section I only provide general descriptions of these models.

5.3.3.1 Trajectory Prediction

When an agent moves in a transportation network, its history trajectory reveals its hidden behavioral patterns. Learning such patterns can help predict its future positions. In my previous work I proposed a deep-learning based trajectory prediction model, which I called step attention, to accurately predict future trajectories of different types of agents (Zhang et al. (2022, 2021)). Step attention only utilizes one of the most basic information of agents, namely their history trajectories, which greatly reduces its information requirements compared to the majority of the literature, which exhaustively uses several pieces of information that might be difficult to obtain. In the step attention model, raw history trajectory information is firstly learned using an LSTM-RNN kernel, which consists of two LSTM layers. The LSTM-RNN kernel is designed to learn the time-related patterns in the raw history trajectory sequence by taking advantages of LSTM's ability to learn time-related patterns. Within each time step, features in previously learned patterns are further captured by a CNN-based time distributed kernel. An augmented attention mechanism is also introduced in the time distributed kernel to mitigate a drawback of CNN, which is limited to learning only local patterns. Finally, features are decoded to the position of the next time step with an GRU-RNN kernel. In my previous work, I demonstrated that the step attention model outperforms state-of-the-art models in terms of pedestrian trajectory prediction. Its final average displacement error (ADE) and final displacement error (FDE) on public benchmark datasets were 0.53 m and 1.72 m, respectively, for 4.8

seconds prediction horizon. Step attention is also able to accurately predict the future 5 seconds vehicle trajectories when trained using a real-world autonomous vehicle dataset– the same dataset used in this work (Zhang et al. (2021)). For vehicle trajectory prediction with a prediction horizon of 5 seconds, the resulting ADE/FDE are 2.95/7.03 m. Using transfer learning, I demonstrated that the step attention model could obtain promising performance when predicting cyclist trajectories even when the dataset was small. The corresponding ADE/FDE for cyclist trajectory predictions were 1.43/3.74 m in a 5 seconds prediction horizon (Zhang et al. (2021)).

Similar to the previous work, in this work I train a trajectory prediction model for each agent type. When planning begins, the subject vehicle uses the trained models to predict the future 5 seconds trajectories of all surrounding agents. Such predicted trajectories are then used to help construct related modules, which which will be discussed in the following sections.

5.3.3.2 Driving Risk Prediction

Driving behaviors can vary in different driving environments, posing different driving risks to surrounding traffic agents. In the previous work, I proposed an unsupervised learning method for driving risk prediction (Zhang et al. (2021, 2022)). Instead of identifying risk at the current time step, the method is able to predict driving risk of a vehicle in future several seconds based on the vehicle's history trajectory information. Training does not require manually labeled data, and the model is able to predict risk probabilities rather than only providing binary risky/non-risky results.

The driving risks are predicted as follows: 1) First, a dynamic time warping (DTW)-based time-series K-means is used to generate driving patterns from a multi-dimensional time series dataset, which consists of 10 seconds full length driving history of vehicles in the study region; 2) An isolation forest anomaly detection model is then trained on the dataset to detect all anomalous data points; 3) Based on the results in step 2, the anomaly rate within each driving pattern is obtained, which provides the risk probability for each driving pattern. I train a model using 10 seconds trajectories. During implementation, when the first 5 seconds of a trajectory is realized, I use the model to predict the risk level for the remaining 5 seconds of the trajectory.

5.3.4 Deep Reinforcement Learning-based Trajectory Planning

In this study I propose a deep RL-based trajectory planning framework. In this section I discuss details of the proposed deep RL learning designs and its learning strategy. I first model the problem as a POMDP, and represent traffic dynamics in driving scenarios as graphs. Graphs are then learned by a deep Q learning model with a custom-designed reward mechanism. The outcome of this offline training process is a trajectory planning policy, ready for use in real-time trajectory planning.

5.3.4.1 Problem Formulation

The purpose of this work is to identify a planning strategy, i.e., a series of acceleration and heading profiles, for the subject vehicles to follow in the future 5 seconds horizon. However, the surrounding traffic environment of the subject vehicle can dynamically change, making it challenging for the subject vehicle to plan its trajectory into a long look-ahead horizon. POMDP is known for its ability to find decision making policies in uncertain environments.

In a driving scenario, the surrounding environment of the subject vehicle continuously changes. To model the dynamic environment affecting the subject vehicle, in this proposed work I represent the surrounding traffic dynamics as a graph, where each agent, including the subject vehicle, is represented by a node. The subject vehicle node has the following properties: history trajectory information, desired destination in 5 seconds (i.e., goal), current acceleration and heading, and current position. Other vehicle nodes have the following properties: history trajectory information, predicted future trajectory information, predicted driving risk information, and current position. Different from vehicle nodes, pedestrian and cyclist nodes have the following properties: history trajectory information, predicted future trajectory information, and current position. A driving environment may have different numbers of surrounding traffic agents. During training, at each planning step, I select a fixed number of nodes for each agent type in the surrounding environment, based on their proximity to the subject vehicle, and use them to construct a directed graph. In this directed an edge is introduced from each surrounding node to the node corresponding to the subject vehicle (Figure 5.4). If the number of nodes for an agent type is less than the required number of nodes for that agent type, I add empty nodes to the graph to make sure all graphs have the same number of nodes for each type of agent, as deep RL models require standardized inputs.

The edge weights at time step t are computed as demonstrated in Eq. (5.4), where ps_t is the positional stressor, indicating the stress that an agent imposes to the subject vehicle at time t, r is the predicted driving risk obtained at the beginning of the planning process, and w^t_{sv}, w^t_{sp} and w^t_{sc} are the edge weights between the subject vehicle and surrounding vehicles, pedestrians, and cyclists, respectively. In this work the positional stressor is computed as the distance between the *predicted* position of the surrounding agent and the position of the subject vehicle at a planned time step.

$$\begin{aligned} w^t_{sv} &= \exp(-0.01\, ps_t + r) \\ w^t_{sp} &= \exp(-0.01\, ps_t) \\ w^t_{sc} &= \exp(-0.01\, ps_t) \end{aligned} \tag{5.4}$$

The scenario graph construction algorithm is demonstrated in Algorithm 4. Note that all graphs are time-dependent. The graph G_t constitutes the observation state of the RL problem. Note that the state space is infinite. An example process is illustrated in Figure 5.4, where 3 vehicle nodes, 2 pedestrian nodes and 2 cyclist nodes are sampled to construct the graph.

Since planning happens in a continuous space, for trajectory planning a reference path is always introduced by the existing literature to assist the subject vehicles in narrowing the exploration space (Fu et al. (2015); Li et al. (2015)). A reference path is a rough driving route for the subject vehicle from the starting location of the vehicle in the beginning of the planning horizon to its desired destination, and is typically provided by other vehicle modules, e.g., Geographic Information System (GIS) (Fu et al. (2015)). The reference path needs not be precise, and only kinematically feasible as well as reasonably consistent

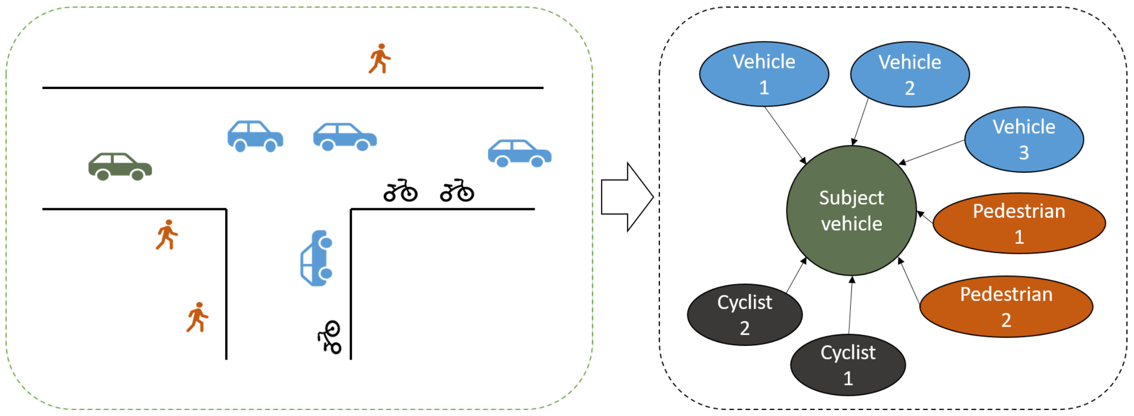


Figure 5.4: Illustration of extracting a graph from a scenario at a given time step. The green circle represents the subject vehicle and ovals linked to it represent the surrounding agents.

Algorithm 4 Scenario graph construction

```
Data: A simulated scenario, time step $t$
Result: Scenario graph $G_t$
1 Initialization: If $t = 0$ (planning begins), predict trajectories for all surrounding agents
    except the subject vehicle; predict driving risks for surrounding vehicles.
2 Build graph:
3 1. Build the target vehicle node at $t$ using: position at $t$, acceleration at $t$, heading at $t$, 5
    seconds history trajectory, desired destination;
4 2. Build surrounding vehicle nodes at $t$ using: predicted position at time $t$, history trajectory,
    predicted trajectory, predicted risk; select $I$ nodes closest to the subject vehicle using the
    $L^2$ norm (if the number of agents is less than $I$, add empty nodes to reach $I$);
5 3. Build surrounding pedestrian nodes at $t$ using: predicted position at time $t$, history
    trajectory, predicted trajectory; select $J$ nodes closest to the subject vehicle using the $L^2$
    norm (if the number of agents is less than $J$, add empty nodes to reach $J$);
6 4. Build surrounding cyclist nodes at $t$ using: predicted position at time $t$, history trajectory,
    predicted trajectory; select $V$ nodes closest to the subject vehicle using the $L^2$ norm (if the
    number of agents is less than $V$, add empty nodes to reach $V$);
7 5. Compute $w_{sv}^t$, $w_{sp}^t$ and $w_{sc}^t$;
8 6. Construct a graph $G_t$ based on nodes and edge weights obtained above.
```

with the road structure, i.e., the path should not drive off road. The reference path can serve as a rough guide to planners, and planning strategies work to correct/modify it. A reference path can be obtained using heuristics, e.g., lane centers from the origin location to the destination (goal), etc. In this work I adopt human driven paths with the same starting and destination locations as reference paths. In cases where a human-driven path may not be available, one can obtain a reference path through vehicle modules such as simulation and example trajectory generation modules (Fu et al. (2015); Tsuchiya et al. (2021)). I also conduct additional experiments, where I demonstrate the performance of the proposed framework using easier-to-generate reference paths in Appendix A. In this appendix, reference paths are generated assuming the subject vehicle approaches its desired destination with constant speed (adjusted to be kinematically feasible).

For a planning task $T_{\text{plan}} = (\text{start}, \text{goal}, P_r, Env)$, assume the reference path is $P_r = (p_r^1, p_r^2, ..., p_r^i, ..., p_r^n)$, where p_r^i is the location of the reference path at the i-th time step, and Env is the surrounding environment. Note that in this work T_{plan} is a variable, and it does not limit the maneuver from start to goal, e.g., driving forward, turning left, turning right, etc. Based on the reference path, the acceleration and heading of the reference path at each time step can be computed. At each planned time step, I define an action to be a tuple that dictates the adjustments to the acceleration and heading values of the reference path at that time step. To reduce complexity of the action space, I discretize acceleration and heading values. I define the acceleration adjustment space as $A_a = [-0.2, -0.19, ..., 0.19, 0.2]$ (unit: $m\ /s^2$), and the heading adjustment space as $A_h = [-5, -4, ..., 4, 5]$ (unit: degrees). Then the action space is defined as $A = A_a \times A_h$, with length 451.

The driving environment is only partially observable as I are not able to enumerate all factors and features that impact the behaviour of agents. Accordingly, I model this problem as a POMDP, as displayed in Figure 5.5. When the subject vehicle drives in environment Env_t at time t, graph G_t is constructed to represent the partially observable environment. Then, G_t is mapped into the belief state b_t , which generates the action policy a_t. Taking action a_t results in the reward R_t. The status of the subject vehicle changes, which is reflected by a graph update in G_{t+1} and a belief state update in b_{t+1}.

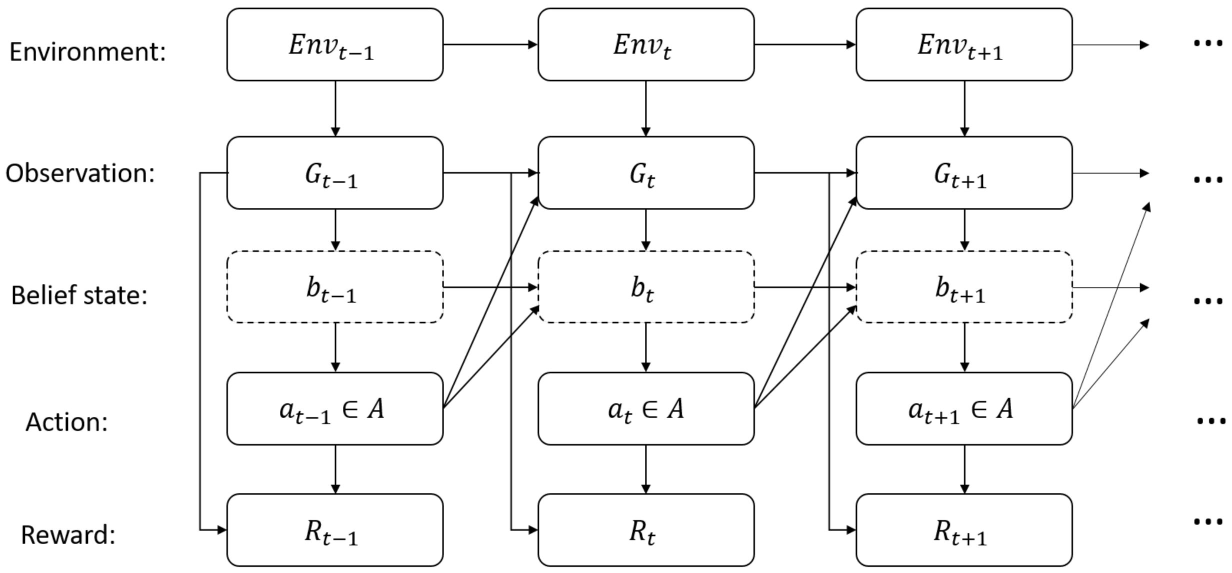


Figure 5.5: POMDP diagram of the formulated problem

5.3.4.2 Reward Design

The purpose of the work is to plan trajectories that lead to safe, comfortable and energy efficient trajectories. Accordingly, in this section I propose a custom-designed reward function for the RL problem. After taking action a_t, the graph is then updated. At a planned time step t, denote the edge weight set of graph G_t as W_t, and the planned trajectory of the subject vehicle up to time step t as S_t. Then, the reward is computed based on W_t and the average driving jerkiness in S_t, which is a widely used metric for measuring driving comfort. The reward function is indicated in Eq. (5.5), where η is the discount factor and τ is the number of maximum look-ahead steps of the RL exploration, which is set to $\tau = 10$ in this study. If based on the selected action the subject vehicle crashes into other traffic agents or drives off road (i.e., outside of geo-fenced area determined by the map), the planning process stops immediately and τ is shortened correspondingly.

$$\begin{aligned} r_t &= e^{-0.2c_s^t - c_j^t} \\ R_t &= r_1 + \sum_{t=2}^{\tau} \eta^{t-1} r_t \end{aligned} \tag{5.5}$$

where

$$\begin{aligned} c_s^t &= \max(W_t) \\ c_j^t &= \frac{2}{t}\sum_{i=1}^{t} |a_i - a_{i-1}| \end{aligned} \tag{5.6}$$

This reward function is designed to have actions (i.e., the adjustments in acceleration and heading relative to the reference path) indirectly affect the reward through graph weights. Recall that the graph weights capture the distance between agents as well as the risk associated with drivers. To maximize the reward return, the subject vehicle is led to maintain its distance from other agents and high-risk vehicles to reduce its exposure to positional stress, which corresponds to safer action policies, and is captured by the term c_s^t in Eq. (5.6). When taking an action, the resulting reward will also be penalized if driving comfort is low (high driving jerkiness), which is captured by the term c_j^t in Eq. (5.6).

5.3.4.3 Model Architecture

A deep Q-learning model is developed in this work to find a trajectory planning policy $\pi(a_t|G_t)$. Compared to the traditional Q-learning algorithm, which uses a Q-table to map an observation to a Q-value and is only able to handle small state spaces, deep Q learning uses neural networks (DQN) to approximate the Bellman function, which is:

$$Q^{new}(G_t, a_t) \leftarrow Q(G_t, a_t) + \alpha(R_t + \gamma \max_a Q(G_{t+1}, a) - Q(G_t, a_t)) \tag{5.7}$$

where G_t and a_t are observation and action at time t, α is the learning rate, and γ is the discount factor. In the deep Q-learning training process, two parallel networks are maintained: a target network and a prediction network. During learning, past experiences seen by the prediction network are stored in memory. When the prediction network output a Q-value, i.e., $Q(G_t, a_t)$, the target network conducts experience replay, randomly sampling a batch of past experiences from memory and using the average Q-value to approximate the target, which is $r_t + \gamma max_a Q(s_{t+1}, a)$ in Eq. (5.7). By minimizing the Q-value gap between the target network and the prediction network, Deep Q-learning

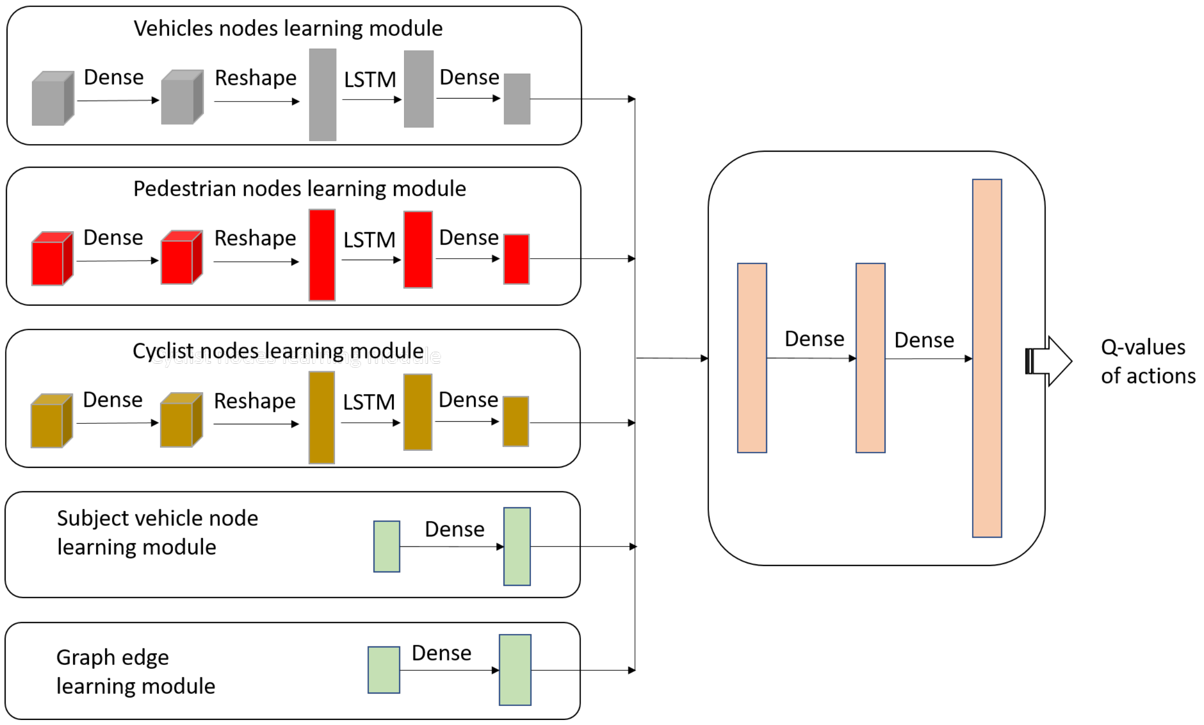

Vehicles nodes learning module
Dense
Reshape
LSTM
Dense
Pedestrian nodes learning module
Dense
Reshape
LSTM
Dense
Cyclist nodes learning module
Dense
Reshape
LSTM
Dense
Subject vehicle node learning module
Dense
Graph edge learning module
Dense
Dense
Dense
Q-values of actions

Figure 5.6: The proposed DQN architecture

approximates the Bellman equation gradually. This approximation design makes Deep Q-learning desirable for learning tasks with large or even infinite state spaces, and learning policies in uncertain environments.

The proposed DQN architecture in this work is shown in Figure 5.6. In this DQN architecture, five modules are designed to learn patterns in observations, G_t. For each module in this architecture, a learning module is established. For agent nodes, I focus on learning history patterns and motion properties of agents. In the vehicle node learning module, the input includes history trajectories of all vehicle nodes in G_t. In this work I set the number of vehicle nodes in each G_t to 5. Consequently, the input shape to the vehicle nodes learning module is [5, 10, 2] as the history trajectory length is 5s seconds long and one time step is 0.5s. The first element of the input tuple is the number of vehicles, the second element is the number of time steps, and the third element includes the vehicles' local geo-coordinates. For pedestrians and cyclists, the number of nodes are set to 3 in each G_t, resulting in input shape [3, 10, 2] for both the pedestrian nodes learning module and the cyclist nodes learning module. All three node learning modules have similar neural network architectures. After taking the input, first a fully connected layer is constructed to learn the pattern in the last dimension of the input tensor with output shape [5,10,1], [3, 10, 1] and [3, 10, 1] for vehicle, pedestrian and cyclist modules, respectively. Inside each module, the resulting tensor is then reshaped and fed into an LSTM network with 64 hidden units. Finally, the features are further learned using a fully connected layer with an output of shape [4]. To learn patterns embedded in the subject vehicle node, based on G_t, I extract the input feature as:

$$f_{\text{subject}} = [\text{goal}_x, \text{goal}_y, x_t, y_t, v_t, a_t, \theta_t] \tag{5.8}$$

where goal_x and goal_y are the coordinates of the desired destination, x_y and y_t indicate the subject vehicle's position at time t, and v_t, a_t, and θ_t are respectively the velocity, acceleration and heading of the subject vehicle at time t. Subject vehicle features are extracted through a fully connected layer with output shape [256]. The last component of G_t to be learned is weights. In this architecture, weights are formulated in the form of 1-D tensors as:

$$f_{\text{weight}} = [w^t_{sv_1}, ..., w^t_{sv_5}, w^t_{sp_1}, ..., w^t_{sp_3}, w^t_{sc_1}, ..., w^t_{sc_3}]. \tag{5.9}$$

Similar to the subject vehicle node learning module, edge weights are also learned with a fully connected layer with output shape [4]. All learned features across the five modules are then concatenated, and additional features are extracted sequentially using three fully connected layers with output shapes [256], [256], and [451]. The total number of parameters in the proposed DQN is 312,072. Final outputs of the model are Q-values of actions at time t. Based on the network architecture, the model is trained with the proposed learning approach in Algorithm 5. In this algorithm, the model is trained after every four actions are taken, and the target model is updated once every 10,000 actions are taken. The first 50,000 actions are randomly taken as model warm-up. Here, $\text{random}(0, 1)$ indicates choosing a random value between 0 and 1, and $\epsilon < 1$.

Algorithm 5 The proposed deep Q learning algorithm

```
Input: learning rate; batch size $bs$; experience replay memory size $M$; epsilon-greedy $\epsilon$ ;
optimizer; max episodes to train $E$
Initialization: model=DQN; target model=model; frame=0
Result: Trained DQN model
1  while episode number $< E$ and not converged do
2      1. Sample a simulated scenario;
3      2. Extract $G_t$ based on Algorithm 4; Reward=0;
4      while time step $< 10$ and not terminated do
5          frame = frame +1;
6          if current experience memory size $<$ 50,000 or random$(0,1) < \epsilon$ then
7          pick a random action
8          else
9              a. prepare model input based on $G_t$;
10             b. pass inputs in (a) to model, pick action with largest Q-value;
11         move to the next state by taking the action, get reward $r$;
12         Reward = Reward $+r$;
13         save states, action, reward to experience memory;
14         if frame $\mod 4 = 0$ and current memory $> bs$ then
15         a. randomly sample $bs$ frames from memory;
16         b. compute future reward with target model;
17         c. train the model on sampled frames based on rewards in sampled frames and
           future rewards by the target model;
18     if frame $\mod 10000 = 0$ then
19         replace target model with model
20     if episode ends then
21         terminate current loop; move to the next driving scenario.
22 Return model.
```

5.4 Experiments and Results

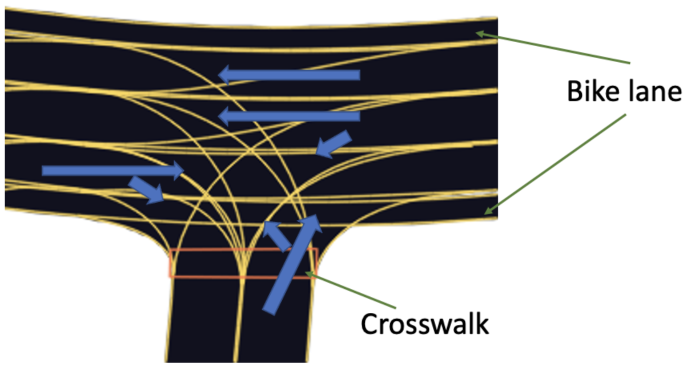
5.4.1 Experiment Data

The proposed work is evaluated using Lyft L5 motion prediction dataset. The dataset contains more than 1,000 hours of autonomous vehicle driving data, and has more than 170,000 driving scenarios. Each raw scenario is 25 seconds long, sampled at 10 Hz. (Houston et al. (2020)). I process the raw dataset in different ways as necessitated by modules that together form the framework–trajectory prediction for traffic agents, driving risk prediction, and driving scenario simulation. I focus the experiments on two driving environments: a non-signalized T-junction and a non-signalized lane-merge intersection. Both of these environments have more conflicting points compared to signalized intersections, resulting in more complex driving environments and making the planning problem more challenging. Illustrations of the two environments are shown in Figure 5.7, where arrows show the traffic movements in their corresponding lanes.

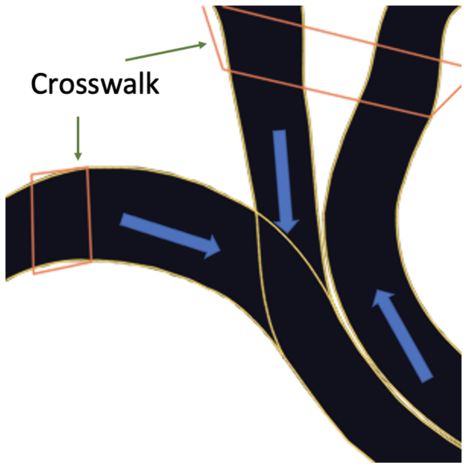
In the regions of interests, full trajectories of agents are first processed into 10 seconds long non-overlapping trips with data frequency of 2 Hz. Agent-specific trajectory datasets are extracted. For trajectory prediction purposes, I separated agent-specific trajectory datasets into training sets and test sets with ratio 8:2. As mentioned earlier, I use 5 second trajectories for training, and predict the position of the agent in the next time step. A sequential prediction strategy is used to predict a full future 5 seconds trajectory. For trajectory prediction models, with a sliding window technique, the size of the training sets for vehicles, pedestrians and cyclists are 100,000, 100,000, and 30,000, respectively. For driving risk prediction, 10,000 vehicle trajectories with lengths of 10 seconds each are extracted from the dataset. For scenario simulation, I only extract driving scenarios where all three agent types are present. Each simulated driving scenario is 10 seconds long, and the planning starting time ($t = 0$) is at the 5th second.

5.4.2 Calibrating the BGM Models

Recall that I train three BGM models, where the purpose of each model is to learn the underlying trajectory patterns of one agent type. In this section, I obtain the best



(a) Illustration of the non-signalized T-junction intersection



(b) Illustration of the lane merge intersection

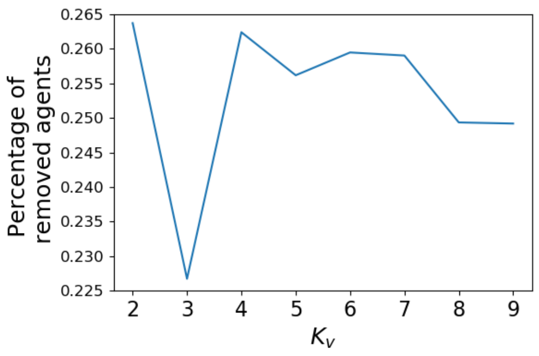
Figure 5.7: Illustration of the two intersections selected for experimentation.

combination for the number of components of the three BGM models so as to generate high-quality scenarios.

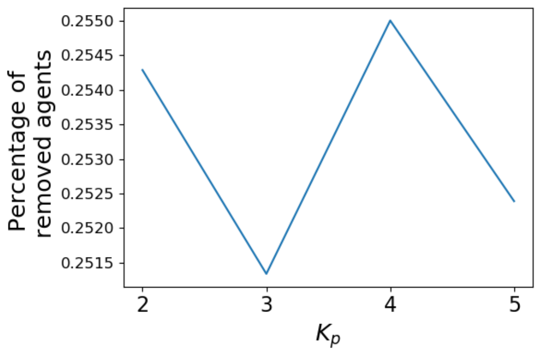
Recall that a raw simulated scenario is needed to be further cleaned (by removing agents) in order to become socially acceptable. To find the best combination of the number of components K_v, K_p, and K_c, I use the social acceptability metric. More precisely, I evaluate the performance of each combination of the number of components, (K_v, K_p, K_c), based on the percentage of agents that will need to be removed in order to achieve socially acceptable scenarios. The more appropriate a parameter combination setting is, the lower the agent removal rate would be. An extreme case would be to set K_v, K_p, and K_c respectively to the total number vehicles, pedestrians, and cyclists in the real-world driving scenarios in the original dataset. In this case, the removal rate will be 0, but the models with be severely overfitted and no new scenarios will be generated. In this work I investigate all combinations of the following values: $K_v \in [2, 9]$, $K_p \in [2, 5]$, and $K_c \in [1, 3]$, where $K_v, K_p, K_c \in \mathbb{N}$. These sets of values are selected based on the number of agents observed in real-world scenarios at intersections of interest. For each combination of (K_v, K_p, K_c), the agent removing rate is computed based on 1,000 simulated driving scenarios, where every simulated driving scenario has 10 vehicles, 5 pedestrians and 5 cyclists. The analysis is shown in Figure 5.8. Based on results demonstrated in Figure 5.8, in this work I set (K_v, K_p, K_c) to $(3, 3, 1)$ as it yields the least percentage of agent removals, which is 0.22.

5.4.3 Reduction of the Exploration Space

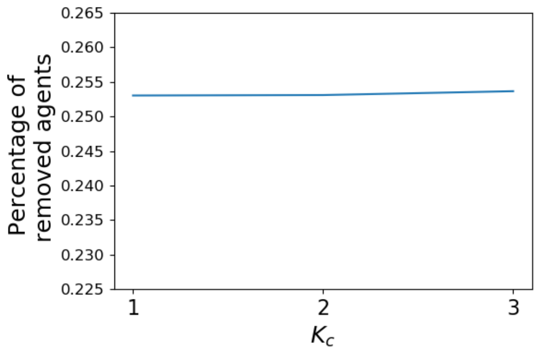
Based on the definition of the action space, the proposed model does not exhaustively search the entire space of continuous planning strategies. Instead, it explores areas in the vicinity of the reference path. This setting reduces random exploration of the model, making it converge faster to the optimal policy. Figure 5.9 illustrates the exploration region of the subject vehicle based on the proposed action space formulation. This region is designed to be large enough not to exclude the optimal policy, but at the same time restrict the action space to actions that are conducive to safer, more comfortable, and more energy efficient trajectories. In Figure 5.9, the subject vehicle, surrounding vehicles, pedestrians, and cyclists are respectively colored as red, blue, orange, and gray. End dots denote their



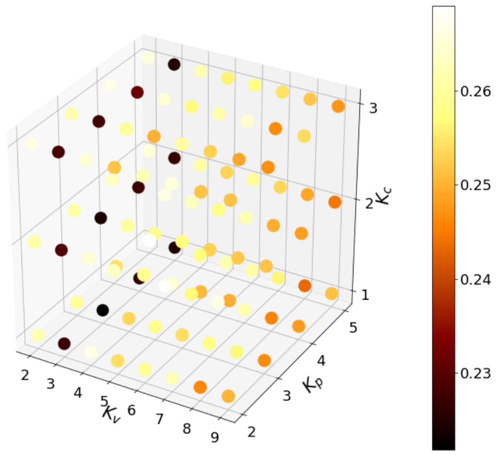
(a) Percentage of removed agents, as the number of components in the vehicle BGM model, K_v, changes, averaged over all simulated scenarios



(b) Percentage of removed agents, as the number of components in the pedestrian BGM model, K_p, changes, averaged over all simulated scenarios



(c) Percentage of removed agents, as the number of components in the cyclist BGM model, K_c, changes, averaged over all simulated scenarios



(d) Percentage of removed agents averaged over 1,000 scenarios w.r.t. different (K_v, K_p, K_c) tuples

Figure 5.8: Finding the best combination for the number of components in the three BGM models

corresponding positions at $t = 0$, while tail lines indicate history trajectories. Regions colored in cyan are regions to explore, based on the definition of the action space.

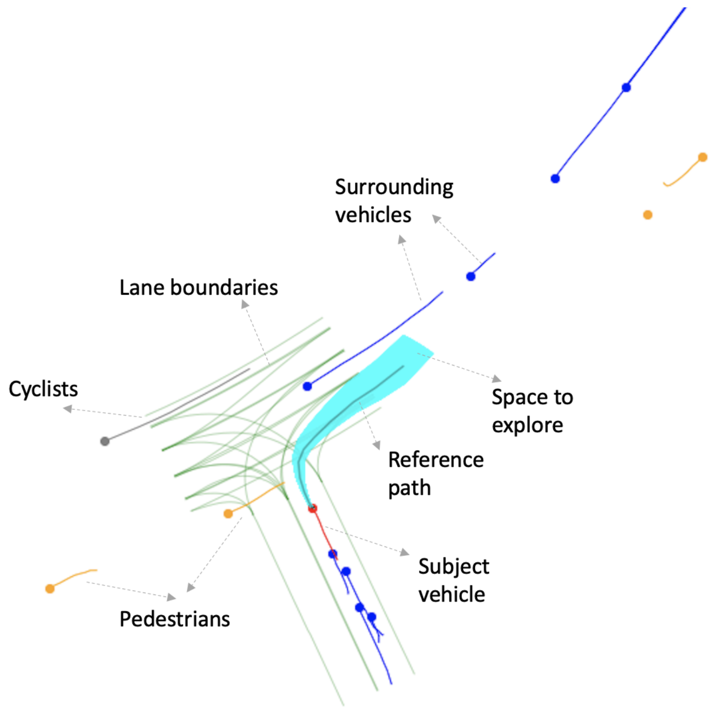
5.4.4 Performance

The reward trends during model training are shown in Figure 5.10, where shaded areas denote the raw reward for each training episode and red lines are reward trends averaged every 10,000 episodes. As indicated in this figure, for both intersections the proposed model converges after 30,000 training episodes (driving scenarios). After convergence, there are still small fluctuations in reward due to the epsilon greedy exploration mechanism used during training to allow the model to explore random policies rather than always using the best policy learned so far.

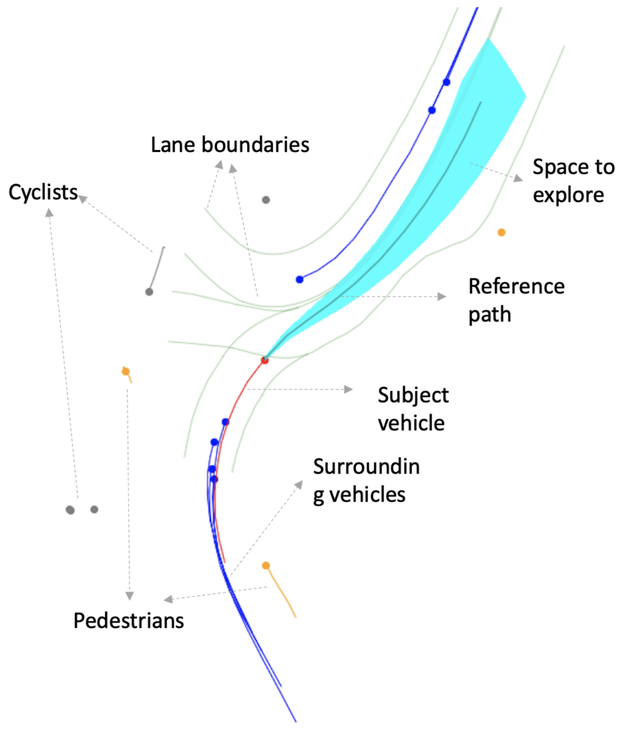
Figure 5.11 shows the performance of the proposed model at several example driving environments and with different planning tasks. As trajectory planning is time-dependent, I also demonstrate these example cases in video format. The video recording which can be accessed at: https://www.youtube.com/watch?v=ioTCsEZuoVc

In Figure 5.11(a), the subject vehicle is on the arterial and is about to drive through the intersection. At the same time, there are two surrounding vehicles who intend to use the same lane as the subject vehicle in the intersection: one vehicle is moving straight through the intersection ahead of the subject vehicle, and the other vehicle is turning right from the minor road into the arterial. The planning strategy generated by the model has the subject vehicle decelerate to let the right turn vehicle complete making the turn. During this process, the velocity of the subject vehicle changes from 12.1 m/s to 10.0 m/s.

In Figure 5.11(b), the subject vehicle attempts to turn right onto the arterial, while a surrounding vehicle is approaching on the arterial. Both vehicles have the same goal of joining the through movement in the same direction. At the same time, a cyclist is moving towards the crosswalk from a far distance. The planned trajectory allows the surrounding vehicle to pass through the intersection first, after which the subject vehicle completes making a right turn. Though the cyclist is far away from the subject vehicle when planning begins, the model has the subject vehicle maintain a wide distance from the cyclist in order to avoid a potential future collision.

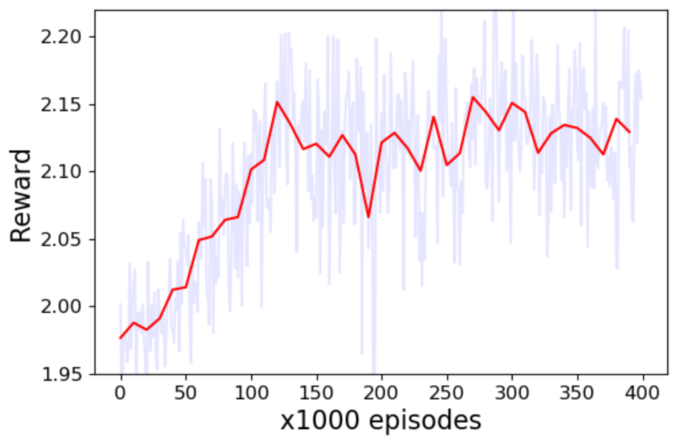


(a) An example of the exploration region at the non-signalized T-junction

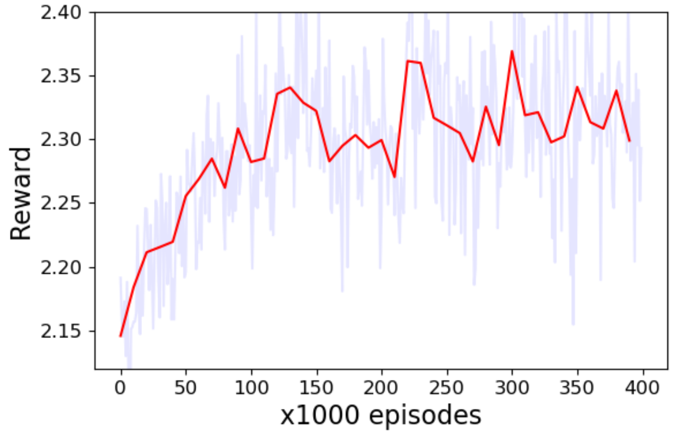


(b) An example of the exploration region at the non-signalized lane merge intersection

Figure 5.9: Illustrations of possible regions for the subject vehicles to explore within 5 seconds on example driving scenarios

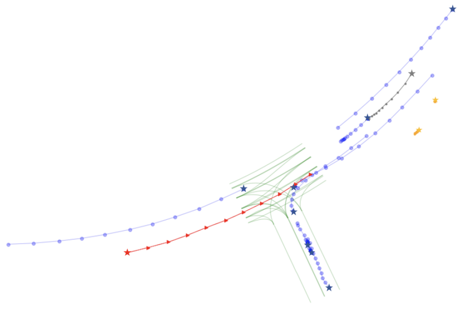


(a) Training reward at the T-junction

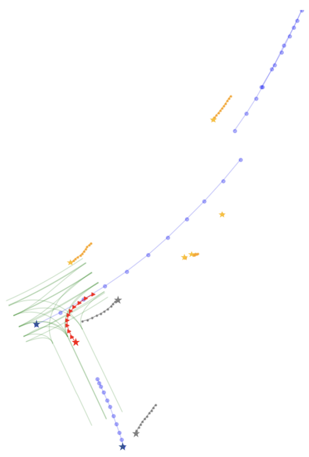


(b) Training reward at the lane merge intersection

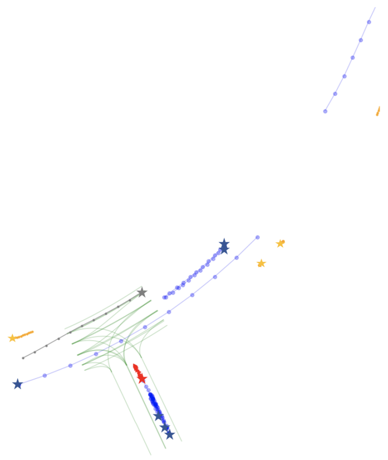
Figure 5.10: Reward trends of model training



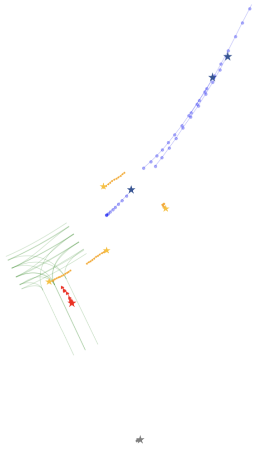
(a) Driving through the T-junction



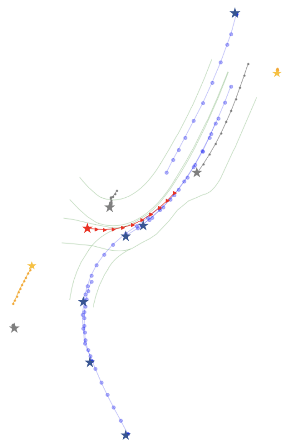
(b) Slowing down for through movement when turning



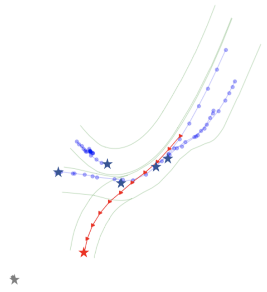
(c) Making a stop for through movement when turning



(d) Slowing down for pedestrian through movement



(e) Merging at the lane merge intersection



(f) Merging from another lane at the lane merge intersection

Figure 5.11: Planning results on example cases. Planned trajectories for the subject vehicle are in red. Ground truth of vehicle, pedestrian and cyclist trajectories are plotted in blue, yellow and gray, respectively. Stars mark the positions of agents at the planning start time.

In Figure 5.11(c) a surrounding vehicle is about to pass through the intersection from a further distance compared to Figure 5.11(b). The policy has the subject vehicle come to an stop, with an average planned velocity of 0.76 m/s.

In Figure 5.11(d) the subject vehicle plans to turn right into the arterial. At the same time, two pedestrians are near the intersection, and one of them ends up crossing the intersection during the planning horizon. The policy has the subject vehicle stop for the entirely of the planning horizon,i.e., 5 seconds. The average speed for the planned trajectory is 1.0 m/s.

Figures 5.11(e) and 5.11(f) show two different merging scenarios at the lane merge intersection. In figure 5.11(e) the subject vehicle is merging from the upper lane. At the beginning of the planning horizon, a leading vehicle V_A is ahead of the subject vehicle, and another vehicle V_B is following the leading vehicle, but in an adjacent lane. Recognizing that V_A and V_B are in two different lanes, the planned trajectory has the subject vehicle closing its gap with V_A, and positioning itself longitudinally ahead of V_B. During the process, the velocity of the subject vehicle initially increases from 3.8 m/s to a peak of 5.1 m/s, and then decreases from its peak to 4.7 m/s.

In 5.11(f) the subject vehicle merges at the intersection from the lower lane. A leading vehicle is ahead of the subject vehicle and a following vehicle is at the upper lane. However, in this scenario when planning begins, the following vehicle is still far from the merging point, traveling with a low speed. As such, the policy has the subject vehicle accelerate and place itself ahead of the following vehicle in the target lane. During this merging process, the velocity of the subject vehicle changes from 4.8 m/s to 6.4 m/s.

5.4.5 Comparison to Human Driving

I compare the performance of the proposed trajectory planning method against human driving in terms of three metrics: safety, driving comfort, and energy efficiency. In order to investigate the overall performance of the proposed method relative to human-driven trajectories, I generate 1,000 simulated driving scenarios for each of the two intersections to serve as test sets. These driving scenarios have different planning tasks and traffic conditions compared to the training set, and will allow us to assess the out-of-sample generalizablity of the framework. To measure safety performance, I use the following procedure for both planned and human-driven trajectories:

1. In each driving scenario and at each time step, compute the positional stressor (the greater, the safer) between the subject vehicle and other surrounding traffic agents.
2. Find the strongest (smallest) positional stressor within the planning horizon (i.e., future 5 seconds, as determined in step 1).
3. Average the smallest positional stressor obtained in step 2 over 1,000 driving scenarios.

The resulting average can be used as a metric to measure the safety risks of both planned and human-driven trajectories.

Following the literature (Zhou et al. (2020); Feng et al. (2017)), I measure driving comfort by the jerkiness in the planned/human-driven trajectories, indicated as c_j^t in Eq. (5.6). I compute the energy consumption of a trajectory based on the work proposed by Galvin (2017). Assuming the subject vehicle is an electric Ford vehicle, energy consumption can be computed as:

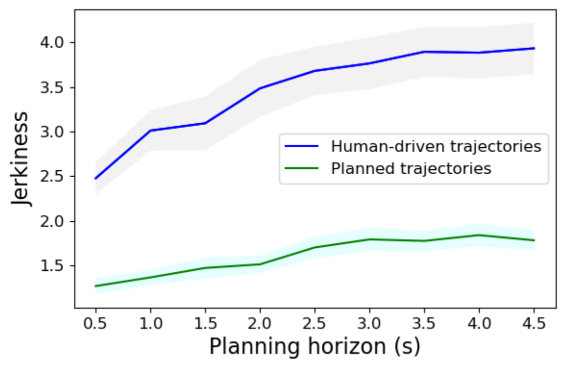
$$E = \sum_{t=1}^{T} (1100v_t - 96.61v_t^2 + 2.745v_t^3 + 1439v_t a_t) \times 0.5 \tag{5.10}$$

where v_t and a_t are respectively the velocity and acceleration at time t, and T is the time horizon, with a maximum of 10 seconds for both planned trajectories and human-driven trajectories.

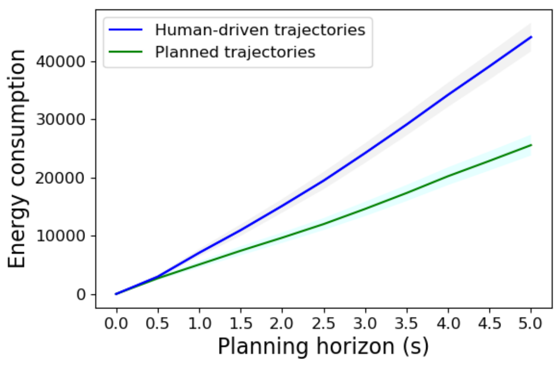
For each measure, I use two-sample t-tests to investigate whether there exists a statistically significant difference between the planned trajectories and human-driven ones. The values are reported in Table 5.1. Results indicate that planned trajectories perform superior to human-driven ones based on all three metrics.

Figure 5.12 shows the general trajectory planning performance of the proposed model compared to human driving. In this figure, blue lines demonstrate the mean values for human driving, while green lines indicate the mean values for planned trajectories. Shaded regions denote 95% confidence intervals.

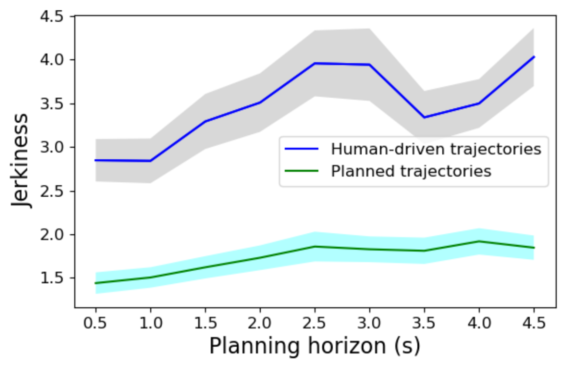
As demonstrated in Table 5.1 and Figure 5.12, planned trajectories by the proposed model demonstrate promising performance compared to human driving. Table 5.1 demonstrates that the proposed model is capable of outperforming human driving with



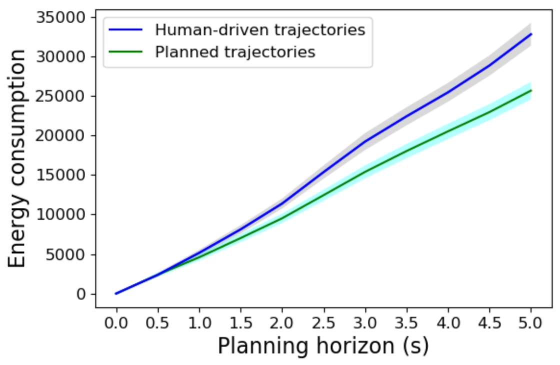
(a) Jerkiness in 5 seconds planning horizon at the T junction



(b) Energy consumption in a 5 seconds planning horizon at the T junction



(c) Jerkiness in a 5 seconds planning horizon at the lane merge intersection



(d) Energy consumption in a 5 seconds planning horizon at the lane merge intersection

Figure 5.12: Model performance compared to human driving

Metrics	Planned trajectories	Human-driven trajectories	p-value
T-junction average safety measure	**8.91**	8.69	5×10^{-10}
T-junction average jerkiness	**1.61**	3.47	8×10^{-8}
T-junction average energy consumption per trajectory	**25538.5**	44075.6	2×10^{-168}
Lane merge average safety measure	**5.73**	5.65	6×10^{-4}
Lane merge average jerkiness	**1.76**	3.57	4×10^{-8}
Lane merge average energy consumption per trajectory	**26511.2**	33654.1	2×10^{-63}

Table 5.1: Result metric statistics

respect to safety, comfort, and energy consumption for a planning horizon as long as 5 seconds with under a 0.5% type I error. Figure 5.12 further demonstrates that in both experimental regions, driving behavior planned by the proposed method outperforms human driving significantly over the planning horizon. At the T-junction, the average energy consumed by the proposed model and human drivers are 2553.8 and 4407.6 per time step (0.5s), respectively. At the lane merge intersection, the average energy consumed by the proposed model and human drivers are 2651.1 and 3365.4 per time step, respectively. Compared to human driving, energy savings for the above two experimental regions are 42.06% and 21.22%.

5.5 Conclusion

In this chapter I develop a predictive framework based on reinforcement learning for vehicle trajectory planning. The framework is capable of devising offline policies, in the form of acceleration and heading profiles, for a target vehicle to follow in complex and dynamically changing driving environments. I abstract the driving environment in a graph, where each agent in the driving scene constitutes a node. The weight of an edge connecting two agent nodes represents the positional stressor between the agents. Agents carry different information depending on their type: the target vehicle carries its history trajectory, while other agents carry information on their predicted trajectories and/or the level of risk they pose on other agents. When traffic dynamics change, nodes and graph edges are adjusted accordingly, resulting in state updates in the corresponding POMDP. I propose a deep Q network to capture patterns in graphs and learn offline planning policies. Since publicly available real-world autonomous driving datasets are scarce, I simulate socially acceptable driving scenarios through a simulation framework, which is based on real-world autonomous vehicle data. I show that the proposed model outperforms human driving when planning vehicle trajectories for horizons as long as five seconds. According to experimental results, the proposed model delivers safer driving trajectories. Statistically significant improvements in driving comfort and energy consumption are also obtained using the proposed model. Energy efficiency savings are specifically high, and they depend on the reference paths used in planning. Compared to human-driven trajectories, up to 42% energy savings can be obtained when using human-driven trajectories as reference paths. Kinematically feasible constant speed reference paths can provide up to 70% energy savings.

CHAPTER VI

Conclusion and Discussions

6.1 Research Findings and Conclusions

The rise of CAV technology has the potential to free drivers from traditional steering wheels and tedious daily driving tasks. Safer driving, less congestion, and more sustainable traveling are parts of the merits of the future application of CAVs. After decades' development, the field of CAV has evolved dramatically along with the advancing of communication and sensor technologies, and artificial intelligence. Although still in the early stage, CAVs are transformed from a theoretical exercise into real-world products. As the CAV technology keeps marching forward, along with appealing opportunities and benefits, the society sees challenges. More and more critical problems keep emerging due to the current immaturity of the technology. The successful development and implementation of CAVs and CAV-related transportation systems would not be achieved without the prioritization of safety. This dissertation puts forward a predictive-prescriptive framework in order to enhance the safety at urban intersections, with the consideration of connectivity and autonomy. The framework can be adopted by CAVs to mitigate real-time risks and plan future trajectories. It can also be utilized by other agencies, e.g., transportation management departments, to promote safety in transportation systems.

At the beginning of the dissertation, an overview on the field of CAVs is provided, which includes general backgrounds on CVs and AVs. Preliminary discussions about potentials and status-quo of CAVs are put forward. Most importantly, from the perspective of human driving, the merit of a predictive-prescriptive mechanism is introduced, which is leveraged by this dissertation to develop a safety-focused framework for CAVs. Four main challenges are analyzed regarding the development and implementation of such framework.

In Chapter II, the dissertation delves into the problem of risky driving prediction, and demonstrates an unsupervised learning approach, i.e., RDP, to predict driving risk in real-time. The model is established by generating driving patterns through clustering and using anomaly detection to identify risky driving in an offline dataset. RDP does not require manual labeling, and is able to provide non-binary prediction results on real-time driving risk of vehicles in a connected vehicle environment. An online assessment strategy is provided for the sake of real-world implementation of RDP.

In Chapter III, the dissertation proposes a deep learning-based model, namely, step attention, for long look-ahead pedestrian trajectory prediction. The model learns patterns in recent history trajectories, and estimates future pedestrian locations within the next several seconds. It is demonstrated that step attention outperforms benchmark methods and the resulting average displacement error is smaller than the typical range of a single step length of a pedestrian. Promising results are also obtained at urban intersections. Furthermore, step attention shows favorable results when utilized for trajectory prediction for heterogeneous traffic agents, as demonstrated in Chapter IV. Using transfer learning, step attention is able to accurately predict trajectories for less-seen traffic agents, such as cyclists, within a large prediction horizon.

On the basis of the predictive models discussed from Chapter II through Chapter IV, this dissertation proposes a predictive trajectory planning framework to generate high-quality driving polices for autonomous vehicles to follow within the next few seconds. A driving scenario generation module and a POMDP-based RL algorithm are proposed in search of high-quality planning strategies in uncertain dynamically changing traffic environments. Experiments are conducted at two different urban intersections, and I show that with the proposed approaches, the framework is able to plan safer, more comfortable, and more energy-efficient future trajectories compared to human-driven ones.

The establishment of methodologies and models in this dissertation meticulously takes the four main aforementioned challenges in Chapter I into consideration. To capture uncertainties in a dynamically changing environment, POMDP is introduced for the formation of the reinforcement learning graphical model, so as to mitigate the drawbacks due to uncertainty of environment observation. Uncertainties also play a role in the setup of RDP and step attention. For RDP, during online assessment, the uncertainty comes from partially-observed trajectory information. When more trajectory information of an approaching vehicle is revealed, its prediction can be updated in real-time to achieve better prediction results. As uncertainty resides in traffic agents' behaviors, step attention adopts a look-back strategy with deep learning, to examine and extract patterns embedded in recent history trajectories of traffic agents, so as to decrease the deficiencies caused by uncertainty and obtain insights on agent behaviors.

The work demonstrated in this dissertation is not limited to a specific region nor a special case. Different experiments are conducted to show the effectiveness and robustness of the proposed framework. When transferred from one location to another, the framework still provides promising results. Without rebuilding a whole new framework, the model can be easily scaled. Such scalability is further promoted by the settings of the framework's components, e.g., eliminating the time-consuming and subjective manual labeling process in RDP. The proposed models are responsive in real-time. Although training takes time, after offline completion of training, implementations of the proposed framework and its components discussed in this dissertation can take place in a timely manner, congruent with requirements of real-time applications.

6.2 Future Directions and Challenges

This dissertation only presents my investigations and findings in the past several years on limited topics within the vast field of connected and automated transportation. Diving deep into the CAV field, along the journey I have noted some future research directions and, more importantly, challenges for the future CAV-related systems:

- **Microscopic vs. macroscopic:** This dissertation focuses more on individual vehicles and other road users at intersections, which belongs to the microscopic paradigm. When microscopic behaviors are aggregated, they may affect macroscopic system-level outcomes and measures, such as traffic flow in a transportation system. Changes in macroscopic measures may in turn impact microscopic-level behaviour. A few topics worth further investigation include: i) the development of microscopic transportation models with the consideration of CAVs; ii) the mechanism through which CAVs impact macroscopic models; iii) the design of transportation systems given the existence of CAVs.

- **Privacy and security:** A CAV collects and shares information with other vehicles or data centers when traveling. Such data collection and sharing pose challenges to user privacy. For example, the RDP study mentioned in this dissertation utilizes connected vehicle information collected in Ann Arbor, MI. The raw BSMs contain real-time motion information of vehicles. Although real vehicle identifications are not revealed for the sake of maintaining privacy, such information may be reconstructed in the real world to identify vehicles. The privacy concern applies to other types of sensitive information. With more and more involvements from public and private sectors in this field, for long-term deployment, principles and regulations should be further established to inform how the data is collected, shared, used, and protected. The privacy concern usually accompanies discussions on security of CAVs. When connected, CAVs can be vulnerable to cyber attacks, whose consequences can be more severe as vehicles are equipped with higher levels of autonomy (Takahashi (2018); Othmane et al. (2015); Wang et al. (2020)). The successful development and deployment of CAVs is not going to occur without detailed investigation and research on privacy and security.

- **Equity:** The very basic principle of transportation is delivering people or goods from one location to another, and therefore it has tremendous social impacts by nature. Instead of only focusing on vehicles, this dissertation addresses the participation of different road users. One reason is that a transportation system is not built only for drivers/vehicles but for providing mobility and accessibility to everyone. Research has shown that if developed and implemented properly, CAV technology can be a solution to narrow the equity gap and promote societal equity (Guo et al. (2020); Bills

(2020)). How to properly develop such systems is still a challenging problem. For public agencies, corresponding management strategies and regulations should be re-investigated and established. For the private sectors, it may increase operational costs and lower profitability. For individual users, public acceptance and additional accessibility costs can contribute to the barriers. However, for the sustainability of the field and for a better society, equity is one of the important concerns to be addressed in future research and development of CAV-related systems.

APPENDIX

APPENDIX A

Supporting Materials For Chapter V

A.1 Constant Speed Reference Path

Here I analyze the performance of the proposed framework using a different reference path. When planning begins, a human driving reference path may be obtained through simulation or other trajectory generation modules. However, users may be interested in a simpler way to generate reference paths. A wildly adopted simple reference path in the literature is a path constructed by assuming the subject vehicle drives towards its goal with constant speed while staying within lane boundaries (Carvalho et al. (2013); Raffo et al. (2009)). When generating the reference path, the following information are required: the planning start time, the vehicle starting position, the desired destination of the vehicle, and the road structure. Given this information, I assume the subject vehicle drives towards its goal with constant speed within the next 5 seconds. Hence, a raw reference path can be denoted as $\text{Path}^1_{ref} = \{ref_1, ref_2, ..., ref_i, ..., ref_{10}\}$, where ref_i is the reference position at the i-th planning step. However, the raw Path^1_{ref} can be kinematically infeasible due to the fact that when transiting from history trajectory to the reference positions in Path^1_{ref}, unreasonable acute acceleration or deceleration may be required. Therefore, I update Path^1_{ref} by imposing kinematic constraints. Since with maximum performance most vehicles can reach 0-60 mph within 10-15 seconds, in this work I updated

Metrics	Planned trajectories	Human-driven trajectories	p-value
T-junction average safety measure	**8.76**	8.61	1.83×10^{-5}
T-junction average jerkiness	**0.70**	3.49	1×10^{-4}
T-junction average energy consumption	**11982.4**	44416.5	1.71×10^{-186}
Lane merge average safety measure	**5.93**	5.71	9.75×10^{-10}
Lane merge average jerkiness	**0.46**	3.21	4.96×10^{-5}
Lane merge average final energy consumption	**8910.2**	34673.1	9.21×10^{-245}

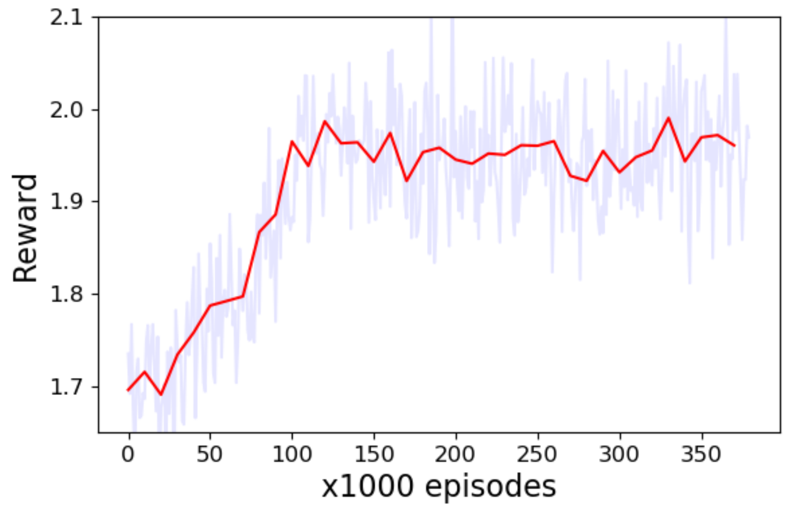
Table A.1: Result metric statistics using a constant speed reference path

the Path$_{ref}^{1}$ with a maximum acceleration limit of $2m/s^2$. When generating the reference path and during the transition from one position to the next in Path$_{ref}^{1}$, if the corresponding acceleration exceeds the maximum limit, the next stop on the path will be computed with an acceleration of $2m/s^2$. I denote the updated kinematically feasible reference path as Path$_{ref}$ $= \{ref_1^{'}, ref_2^{'}, ..., ref_i^{'}, ..., ref_{10}^{'}\}$. Similar to our settings in Section 3, I train the model with Path$_{ref}$.

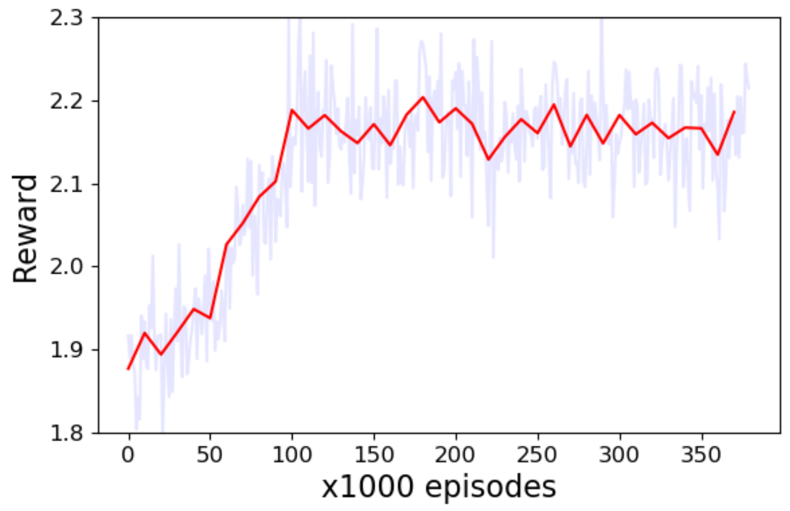
Figure A.1 displays the reward trends during training on both T-junction and lane merge intersections. This figure demonstrates that using a simple reference path the model convergences to similar reward values compared to the results obtained when using human-driven trajectories as reference paths in Chapter V.

The performance metrics are demonstrated in Table A.1. The values in this table are averaged over 1,000 simulated scenarios. As this table demonstrates, with the new reference path planned trajectories outperform human-driven ones based on all three metrics. At the T-junction, the average energy consumed by the proposed model and human drivers are 1198.3 and 4441.6 per time step (0.5s), respectively. At the lane merge intersection, the average energy consumed by the proposed model and human drivers are 891.1 and 3467.3 per time step, respectively. In both scenarios, planned trajectories obtain over 70% reduction in energy consumption compared to human-driven trajectories.

Figure A.2 demonstrates the general trajectory planning performance of the proposed model compared to human drivers. In the figure, blue lines demonstrate the mean values

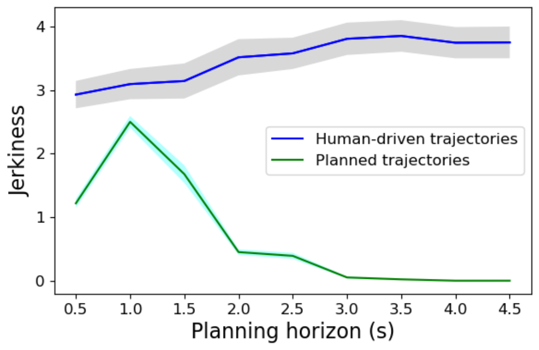


(a) Training reward at the T-junction using a constant speed reference path

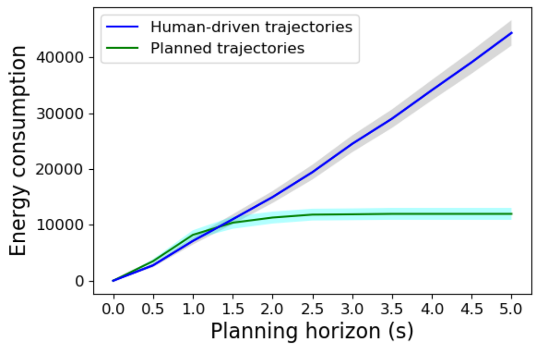


(b) Training reward at the lane merge intersection using a constant speed reference path

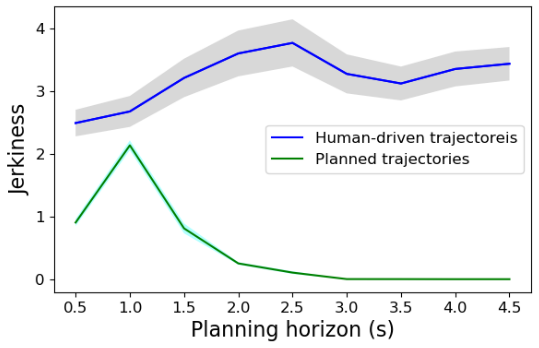
Figure A.1: Reward trends of model training



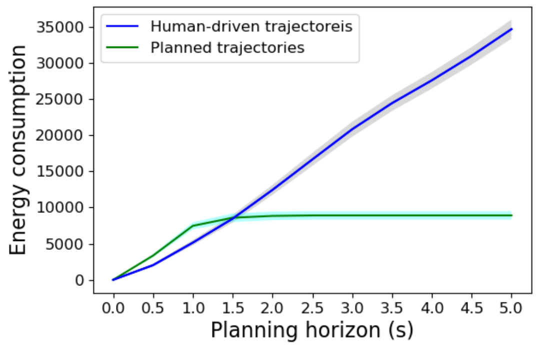
(a) Jerkiness in 5 seconds planning horizon at the T junction



(b) Energy consumption in 5 seconds planning horizon at the T junction



(c) Jerkiness in 5 seconds planning horizon at the lane merge intersection



(d) Energy consumption in 5 seconds planning horizon at the lane merge intersection

Figure A.**2:** Model performance compared to human driving for human driving, while green lines indicate the mean values for the planned trajectories. Shaded regions denote 95% confidence intervals.

As indicated in Table A.1 and Figure A.2, with a constant speed reference path the model plans trajectories that outperform human-driven trajectories in terms of safety, driving comfort, and energy efficiency. The planned trajectories show much less jerkiness and consume less energy within the 5 seconds planning horizon. When compared to Figure 5.12, I can also notice a difference between using a constant speed model to generate reference paths and using human-driven trajectories as reference paths. Compared to results shown in Figure 5.12, using Path_{ref} the model has better energy efficiency at both intersections. However, at the beginning the jerkiness can be high compared to using human-driven trajectories as reference paths. The reason is that since the utility of a reference path is to reduce the model exploration space during training, with Path_{ref} the model is intrigued to learn a planning strategy to reach the goal with the least acceleration/deceleration. This leads to lower energy consumption. However, to achieve that, at early steps within the planning horizon the model generates larger jerkiness compared to human-driven reference paths. This is due to a more turbulent acceleration profile at the beginning when the trajectory transitions from the history trajectory to a near-constant speed trajectory.

To sum up, with different reference paths, the proposed model can learn different planning strategies. Using a kinematically feasible constant speed reference path, the planned 5 seconds trajectories outperform human-driven trajectories in terms of safety, driving comfort and energy consumption. However compared to using human-driven trajectories as reference paths, which naturally provides kinematically feasible paths, the model shows higher jerkiness at the early planning steps and lower jerkiness toward the end of planning. With Path_{ref}, compared to using human-driven trajectories as reference paths, the model sacrifices driving comfort at the early time steps to achieve better energy efficiency.

BIBLIOGRAPHY

AACVTE (2020, Apr). Ann arbor connected vehicle test environment.

Abdolmaleki, M., M. Shahabi, Y. Yin, and N. Masoud (2019). Itinerary planning for cooperative truck platooning. *Available at SSRN*.

Alahi, A., K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese (2016). Social lstm: Human trajectory prediction in crowded spaces. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 961–971.

Ali, F., A. Ali, M. Imran, R. A. Naqvi, M. H. Siddiqi, and K.-S. Kwak (2021). Traffic accident detection and condition analysis based on social networking data. *Accident Analysis & Prevention 151*, 105973.

Aljaafreh, A., N. Alshabatat, and M. S. N. Al-Din (2012). Driving style recognition using fuzzy logic. In *2012 IEEE International Conference on Vehicular Electronics and Safety (ICVES 2012)*, pp. 460–463. IEEE.

Almasri, M. M., A. M. Alajlan, and K. M. Elleithy (2016). Trajectory planning and collision avoidance algorithm for mobile robotics system. *IEEE Sensors journal 16*(12), 5021–5028.

Alonso, F., C. Esteban, L. Montoro, and A. Serge (2019). Conceptualization of aggressive driving behaviors through a perception of aggressive driving scale (pad). *Transportation research part F: traffic psychology and behaviour 60*, 415–426.

Altché, F. and A. de La Fortelle (2017). An lstm network for highway trajectory prediction. In *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, pp. 353–359. IEEE.

Amer, M., M. Goldstein, and S. Abdennadher (2013). Enhancing one-class support vector machines for unsupervised anomaly detection. In *Proceedings of the ACM SIGKDD Workshop on Outlier Detection and Description*, pp. 8–15.

Aradi, S. (2020). Survey of deep reinforcement learning for motion planning of autonomous vehicles. *IEEE Transactions on Intelligent Transportation Systems*.

Arulkumaran, K., M. P. Deisenroth, M. Brundage, and A. A. Bharath (2017). Deep reinforcement learning: A brief survey. *IEEE Signal Processing Magazine 34*(6), 26–38.

Aycard, O., Q. Baig, S. Bota, F. Nashashibi, S. Nedevschi, C. Pantilie, M. Parent, P. Resende, and T.-D. Vu (2011). Intersection safety using lidar and stereo vision sensors. In *2011 IEEE Intelligent Vehicles Symposium (IV)*, pp. 863–869. IEEE.

Bagloee, S. A., M. Tavana, M. Asadi, and T. Oliver (2016). Autonomous vehicles: challenges, opportunities, and future implications for transportation policies. *Journal of modern transportation 24*(4), 284–303.

Bahdanau, D., K. Cho, and Y. Bengio (2014). Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Barbosa, N. M. and M. Chen (2019). Rehumanized crowdsourcing: a labeling framework addressing bias and ethics in machine learning. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, pp. 1–12.

Bello, I., B. Zoph, A. Vaswani, J. Shlens, and Q. V. Le (2019). Attention augmented convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 3286–3295.

Berndt, D. J. and J. Clifford (1994). Using dynamic time warping to find patterns in time series. In *KDD workshop*, Volume 10, pp. 359–370. Seattle, WA.

Bills, T. (2020). On transportation equity implications of connected and autonomous vehicles (cav) a review of methodologies.

Bissell, D. (2018). Automation interrupted: How autonomous vehicle accidents transform the material politics of automation. *Political Geography 65*, 57–66.

Bose, B., J. Dutta, S. Ghosh, P. Pramanick, and S. Roy (2018). Smartphone based system for real-time aggressive driving detection and marking rash driving-prone areas. In *Proceedings of the Workshop Program of the 19th International Conference on Distributed Computing and Networking*, pp. 1–6.

Burger, C. and M. Lauer (2018). Cooperative multiple vehicle trajectory planning using miqp. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pp. 602–607. IEEE.

Cai, P., Y. Sun, Y. Chen, and M. Liu (2019). Vision-based trajectory planning via imitation learning for autonomous vehicles. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pp. 2736–2742. IEEE.

Carvalho, A., Y. Gao, A. Gray, H. E. Tseng, and F. Borrelli (2013). Predictive control of an autonomous ground vehicle using an iterative linearization approach. In *16th International IEEE conference on intelligent transportation systems (ITSC 2013)*, pp. 2335–2340. IEEE.

Chandra, R., U. Bhattacharya, A. Bera, and D. Manocha (2019). Traphic: Trajectory prediction in dense and heterogeneous traffic using weighted interactions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8483–8492.

Cheng, H. and M. Sester (2018). Mixed traffic trajectory prediction using lstm–based models in shared space. In *The Annual International Conference on Geographic Information Science*, pp. 309–325. Springer.

Cheng, J., L. Dong, and M. Lapata (2016). Long short-term memory-networks for machine reading. *arXiv preprint arXiv:1601.06733*.

Chorowski, J. K., D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio (2015). Attention-based models for speech recognition. In *Advances in neural information processing systems*, pp. 577–585.

Claussmann, L., M. Revilloud, D. Gruyer, and S. Glaser (2019). A review of motion planning for highway autonomous driving. *IEEE Transactions on Intelligent Transportation Systems 21*(5), 1826–1848.

Collins, J. W. (2011). Assessing risk and driving risk mitigation for first-of-a-kind advanced reactors. Technical report, Idaho National Laboratory (INL).

Constantinou, E., G. Panayiotou, N. Konstantinou, A. Loutsiou-Ladd, and A. Kapardis (2011). Risky and aggressive driving in young adults: Personality matters. *Accident Analysis & Prevention 43*(4), 1323–1331.

Cronin, B. (2012, Oct). Vehicle based data and availability.

Danaf, M., M. Abou-Zeid, and I. Kaysi (2015). Modeling anger and aggressive driving behavior in a dynamic choice–latent variable model. *Accident Analysis & Prevention 75*, 105–118.

Das, S. (2021). Autonomous vehicle safety: Understanding perceptions of pedestrians and bicyclists. *Transportation research part F: traffic psychology and behaviour 81*, 41–54.

Deffenbacher, J. L., E. R. Oetting, and R. S. Lynch (1994). Development of a driving anger scale. *Psychological reports 74*(1), 83–91.

Dendorfer, P., A. Osep, and L. Leal-Taixé (2020). Goal-gan: Multimodal trajectory prediction based on goal position estimation. In *Proceedings of the Asian Conference on Computer Vision*.

Deo, N. and M. M. Trivedi (2017). Learning and predicting on-road pedestrian behavior around vehicles. In *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, pp. 1–6. IEEE.

Deo, N. and M. M. Trivedi (2018a). Convolutional social pooling for vehicle trajectory prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 1468–1476.

Deo, N. and M. M. Trivedi (2018b). Multi-modal trajectory prediction of surrounding vehicles with maneuver based lstms. In *2018 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1179–1184. IEEE.

DePasquale, J. P., E. S. Geller, S. W. Clarke, and L. C. Littleton (2001). Measuring road rage: Development of the propensity for angry driving scale. *Journal of Safety Research 32*(1), 1–16.

Dey, R. and F. M. Salemt (2017). Gate-variants of gated recurrent unit (gru) neural networks. In *2017 IEEE 60th international midwest symposium on circuits and systems (MWSCAS)*, pp. 1597–1600. IEEE.

Ding, W. and S. Shen (2019). Online vehicle trajectory prediction using policy anticipation network and optimization-based context reasoning. In *2019 International Conference on Robotics and Automation (ICRA)*, pp. 9610–9616. IEEE.

Ding, Z. and M. Fei (2013). An anomaly detection approach based on isolation forest algorithm for streaming data using sliding window. *IFAC Proceedings Volumes 46*(20), 12–17.

Droz, P.-Y. and J. Zhu (2014, August 21). Method to detect nearby aggressive drivers and adjust driving modes. US Patent App. 13/772,615.

Dula, C. S. and M. E. Ballard (2003). Development and evaluation of a measure of dangerous, aggressive, negative emotional, and risky driving 1. *Journal of Applied Social Psychology 33*(2), 263–282.

Dula, C. S. and E. S. Geller (2003). Risky, aggressive, or emotional driving: Addressing the need for consistent communication in research. *Journal of safety research 34*(5), 559–566.

Ellison, P. A., J. M. Govern, H. L. Petri, and M. H. Figler (1995). Anonymity and aggressive driving behavior: A field study. *Journal of Social Behavior and Personality 10*(1), 265.

Eren, H., S. Makinist, E. Akin, and A. Yilmaz (2012). Estimating driving behavior by a smartphone. In *2012 IEEE Intelligent Vehicles Symposium*, pp. 234–239. IEEE.

Etinger, A., N. Balal, B. Litvak, M. Einat, B. Kapilevich, and Y. Pinhasi (2013). Non-imaging mm-wave fmcw sensor for pedestrian detection. *IEEE Sensors Journal 14*(4), 1232–1237.

Fagnant, D. J. and K. Kockelman (2015). Preparing a nation for autonomous vehicles: opportunities, barriers and policy recommendations. *Transportation Research Part A: Policy and Practice 77*, 167–181.

Faisal, A., M. Kamruzzaman, T. Yigitcanlar, and G. Currie (2019). Understanding autonomous vehicles. *Journal of transport and land use 12*(1), 45–72.

Favarò, F. M., N. Nader, S. O. Eurich, M. Tripp, and N. Varadaraju (2017). Examining accident reports involving autonomous vehicles in california. *PLoS one 12*(9), e0184952.

Feng, F., S. Bao, J. R. Sayer, C. Flannagan, M. Manser, and R. Wunderlich (2017). Can vehicle longitudinal jerk be used to identify aggressive drivers? an examination using naturalistic driving data. *Accident Analysis & Prevention 104*, 125–136.

Fenton, R., G. Melocik, and K. Olson (1976). On the steering of automated vehicles: Theory and experiment. *IEEE Transactions on Automatic Control 21*(3), 306–315.

Fu, M., K. Zhang, Y. Yang, H. Zhu, and M. Wang (2015). Collision-free and kinematically feasible path planning along a reference path for autonomous vehicle. In *2015 IEEE Intelligent Vehicles Symposium (IV)*, pp. 907–912. IEEE.

Galata, A., N. Johnson, and D. Hogg (2001). Learning variable-length markov models of behavior. *Computer Vision and Image Understanding 81*(3), 398–413.

Galvin, R. (2017). Energy consumption effects of speed and acceleration in electric vehicles: Laboratory case studies and implications for drivers and policymakers. *Transportation Research Part D: Transport and Environment 53*, 234–248.

Gao, K., H. Wang, J. Nazarko, and G. Chobanov (2021). Indoor trajectory prediction algorithm based on communication analysis of built-in sensors in mobile terminals. *IEEE Sensors Journal 21*(22), 25234–25242.

Gardner, G. T. and P. C. Stern (1996). *Environmental problems and human behavior.* Allyn & Bacon.

Geary, T. and D. Danks (2019). Balancing the benefits of autonomous vehicles. In *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*, pp. 181–186.

Greenblatt, J. B. and S. Saxena (2015). Autonomous taxis could greatly reduce greenhouse-gas emissions of us light-duty vehicles. *nature climate change 5*(9), 860–863.

Greff, K., R. K. Srivastava, J. Koutník, B. R. Steunebrink, and J. Schmidhuber (2016). Lstm: A search space odyssey. *IEEE transactions on neural networks and learning systems 28*(10), 2222–2232.

Guo, Y., Z. Chen, A. Stuart, X. Li, and Y. Zhang (2020). A systematic overview of transportation equity in terms of accessibility, traffic emissions, and safety outcomes: From conventional to emerging technologies. *Transportation research interdisciplinary perspectives 4*, 100091.

Gupta, A., J. Johnson, L. Fei-Fei, S. Savarese, and A. Alahi (2018). Social gan: Socially acceptable trajectories with generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2255–2264.

Hasan, M. H. and P. Van Hentenryck (2021). The benefits of autonomous vehicles for community-based trip sharing. *Transportation Research Part C: Emerging Technologies 124*, 102929.

He, K., X. Zhang, S. Ren, and J. Sun (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778.

He, T., X. Tan, Y. Xia, D. He, T. Qin, Z. Chen, and T.-Y. Liu (2018). Layer-wise coordination between encoder and decoder for neural machine translation. In *Advances in Neural Information Processing Systems*, pp. 7944–7954.

Hewing, L., E. Arcari, L. P. Fröhlich, and M. N. Zeilinger (2020). On simulation and trajectory prediction with gaussian process dynamics. In *Learning for Dynamics and Control*, pp. 424–434. PMLR.

Hoeger, W. W., L. Bond, L. Ransdell, J. M. Shimon, and S. Merugu (2008). One-mile step count at walking and running speeds. *ACSM's Health & Fitness Journal 12*(1), 14–19.

Hong, J.-H., B. Margines, and A. K. Dey (2014). A smartphone-based sensing platform to model aggressive driving behaviors. In *Proceedings of the sigchi conference on human factors in computing systems*, pp. 4047–4056.

Hou, X. and J. Bergmann (2020). Pedestrian dead reckoning with wearable sensors: A systematic review. *IEEE Sensors Journal 21*(1), 143–152.

Houenou, A., P. Bonnifait, V. Cherfaoui, and W. Yao (2013). Vehicle trajectory prediction based on motion model and maneuver recognition. In *2013 IEEE/RSJ international conference on intelligent robots and systems*, pp. 4363–4369. IEEE.

Houston, J., G. Zuidhof, L. Bergamini, Y. Ye, L. Chen, A. Jain, S. Omari, V. Iglovikov, and P. Ondruska (2020). One thousand and one hours: Self-driving motion prediction dataset. *arXiv preprint arXiv:2006.14480*.

Houston, J., G. Zuidhof, L. Bergamini, Y. Ye, A. Jain, S. Omari, V. Iglovikov, and P. Ondruska (2020). One thousand and one hours: Self-driving motion prediction dataset. `https://level5.lyft.com/dataset/`.

Houston, J. M., P. B. Harris, and M. Norman (2003). The aggressive driving behavior scale: Developing a self-report measure of unsafe driving practices. *North American Journal of Psychology 5*(2), 269–278.

Huang, Y., H. Bi, Z. Li, T. Mao, and Z. Wang (2019). Stgat: Modeling spatial-temporal interactions for human trajectory prediction. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 6272–6281.

Huang, Z., J. Wang, L. Pi, X. Song, and L. Yang (2021). Lstm based trajectory prediction model for cyclist utilizing multiple interactions with environment. *Pattern Recognition 112*, 107800.

Hubmann, C., J. Schulz, M. Becker, D. Althoff, and C. Stiller (2018). Automated driving in uncertain environments: Planning with interaction and uncertain maneuver prediction. *IEEE transactions on intelligent vehicles 3*(1), 5–17.

Hulse, L. M., H. Xie, and E. R. Galea (2018). Perceptions of autonomous vehicles: Relationships with road users, risk, gender and age. *Safety science 102*, 1–13.

Hult, R., G. R. Campos, E. Steinmetz, L. Hammarstrand, P. Falcone, and H. Wymeersch (2016). Coordination of cooperative autonomous vehicles: Toward safer and more efficient road transportation. *IEEE Signal Processing Magazine 33*(6), 74–84.

Hussein, A., F. Garcia, J. M. Armingol, and C. Olaverri-Monreal (2016). P2v and v2p communication for pedestrian warning on the basis of autonomous vehicles. In *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, pp. 2034–2039. IEEE.

Jaipuria, N., G. Habibi, and J. P. How (2018). Learning in the curbside coordinate frame for a transferable pedestrian trajectory prediction model. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pp. 3125–3131. IEEE.

Jeong, H., Y. Jang, P. J. Bowman, and N. Masoud (2018). Classification of motor vehicle crash injury severity: A hybrid approach for imbalanced data. *Accident Analysis & Prevention 120*, 250–261.

Jo, K., J. Kim, D. Kim, C. Jang, and M. Sunwoo (2015). Development of autonomous car"part ii: A case study on the implementation of an autonomous driving system based on distributed architecture. *IEEE Transactions on Industrial Electronics 62*(8), 5119–5132.

Johnson, D. A. and M. M. Trivedi (2011). Driving style recognition using a smartphone as a sensor platform. In *2011 14th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, pp. 1609–1615. IEEE.

Josef, S. and A. Degani (2020). Deep reinforcement learning for safe local planning of a ground vehicle in unknown rough terrain. *IEEE Robotics and Automation Letters 5*(4), 6748–6755.

Kalra, N. and S. M. Paddock (2016). Driving to safety: How many miles of driving would it take to demonstrate autonomous vehicle reliability? *Transportation Research Part A: Policy and Practice 94*, 182–193.

Kamrani, M., R. Arvin, and A. J. Khattak (2018). Extracting useful information from basic safety message data: An empirical study of driving volatility measures and crash frequency at intersections. *Transportation research record 2672*(38), 290–301.

Kang, W. and Y. Han (2014). Smartpdr: Smartphone-based pedestrian dead reckoning for indoor localization. *IEEE Sensors journal 15*(5), 2906–2916.

Karaduman, O., H. Eren, H. Kurum, and M. Celenk (2013). An effective variable selection algorithm for aggressive/calm driving detection via can bus. In *2013 International Conference on Connected Vehicles and Expo (ICCVE)*, pp. 586–591. IEEE.

Kato, S., E. Takeuchi, Y. Ishiguro, Y. Ninomiya, K. Takeda, and T. Hamada (2015). An open approach to autonomous vehicles. *IEEE Micro 35*(6), 60–68.

Ketchen, D. J. and C. L. Shook (1996). The application of cluster analysis in strategic management research: an analysis and critique. *Strategic management journal 17*(6), 441–458.

Kim, B., B. Choi, S. Park, H. Kim, and E. Kim (2015). Pedestrian/vehicle detection using a 2.5-d multi-layer laser scanner. *IEEE Sensors Journal 16*(2), 400–408.

Kim, B., C. M. Kang, J. Kim, S. H. Lee, C. C. Chung, and J. W. Choi (2017). Probabilistic vehicle trajectory prediction over occupancy grid map via recurrent neural network. In *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, pp. 399–404. IEEE.

Kim, B., D. Necsulescu, and J. Sasiadek (2001). Model predictive control of an autonomous vehicle. In *2001 IEEE/ASME International Conference on Advanced Intelligent Mechatronics. Proceedings (Cat. No. 01TH8556)*, Volume 2, pp. 1279–1284. IEEE.

Kim, S., S. J. Guy, W. Liu, D. Wilkie, R. W. Lau, M. C. Lin, and D. Manocha (2015). Brvo: Predicting pedestrian trajectories using velocity-space reasoning. *The International Journal of Robotics Research 34*(2), 201–217.

Koopman, P. and M. Wagner (2017). Autonomous vehicle safety: An interdisciplinary challenge. *IEEE Intelligent Transportation Systems Magazine 9*(1), 90–96.

Krahé, B. and I. Fenske (2002). Predicting aggressive driving behavior: The role of macho personality, age, and power of car. *Aggressive Behavior: Official Journal of the International Society for Research on Aggression 28*(1), 21–29.

Lajunen, T. and D. Parker (2001). Are aggressive people aggressive drivers? a study of the relationship between self-reported general aggressiveness, driver anger and aggressive driving. *Accident Analysis & Prevention 33*(2), 243–255.

Lee, J. and K. Jang (2019). A framework for evaluating aggressive driving behaviors based on in-vehicle driving records. *Transportation research part F: traffic psychology and behaviour 65*, 610–619.

Lefèvre, S., D. Vasquez, and C. Laugier (2014). A survey on motion prediction and risk assessment for intelligent vehicles. *ROBOMECH journal 1*(1), 1–14.

Lerner, A., Y. Chrysanthou, and D. Lischinski (2007). Crowds by example. In *Computer graphics forum*, Volume 26, pp. 655–664. Wiley Online Library.

Li, H., A. Shrestha, H. Heidari, J. Le Kernec, and F. Fioranelli (2019). Bi-lstm network for multimodal continuous human activity recognition and fall detection. *IEEE Sensors Journal 20*(3), 1191–1201.

Li, X., Z. Sun, D. Cao, Z. He, and Q. Zhu (2015). Real-time trajectory planning for autonomous urban driving: Framework, algorithms, and verifications. *IEEE/ASME Transactions on mechatronics 21*(2), 740–753.

Liao, T. W. (2005). Clustering of time series data"a survey. *Pattern recognition 38*(11), 1857–1874.

Liu, C., S. Lee, S. Varnhagen, and H. E. Tseng (2017). Path planning for autonomous vehicles using model predictive control. In *2017 IEEE Intelligent Vehicles Symposium (IV)*, pp. 174–179. IEEE.

Liu, F. T., K. M. Ting, and Z.-H. Zhou (2008). Isolation forest. In *2008 eighth ieee international conference on data mining*, pp. 413–422. IEEE.

Liu, F. T., K. M. Ting, and Z.-H. Zhou (2012). Isolation-based anomaly detection. *ACM Transactions on Knowledge Discovery from Data (TKDD) 6*(1), 3.

Liu, X., N. Masoud, Q. Zhu, and A. Khojandi (2022). A markov decision process framework to incorporate network-level data in motion planning for connected and automated vehicles. *Transportation Research Part C: Emerging Technologies 136*, 103550.

Liu, X., G. Zhao, N. Masoud, and Q. Zhu (2020). Trajectory planning for connected and automated vehicles: Cruising, lane changing, and platooning. *arXiv preprint arXiv:2001.08620*.

Lu, N., N. Cheng, N. Zhang, X. Shen, and J. W. Mark (2014). Connected vehicles: Solutions and challenges. *IEEE internet of things journal 1*(4), 289–299.

Luong, M.-T., H. Pham, and C. D. Manning (2015). Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*.

Ma, L., J. Xue, K. Kawabata, J. Zhu, C. Ma, and N. Zheng (2015). Efficient sampling-based motion planning for on-road autonomous driving. *IEEE Transactions on Intelligent Transportation Systems 16*(4), 1961–1976.

Ma, Y., K. Tang, S. Chen, A. J. Khattak, and Y. Pan (2020). On-line aggressive driving identification based on in-vehicle kinematic parameters under naturalistic driving conditions. *Transportation Research Part C: Emerging Technologies 114*, 554–571.

Ma, Y., X. Zhu, S. Zhang, R. Yang, W. Wang, and D. Manocha (2019). Trafficpredict: Trajectory prediction for heterogeneous traffic-agents. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Volume 33, pp. 6120–6127.

Mannion, P., J. Duggan, and E. Howley (2016). An experimental review of reinforcement learning algorithms for adaptive traffic signal control. *Autonomic road transport support systems*, 47–66.

Martinelli, A., H. Gao, P. D. Groves, and S. Morosi (2017). Probabilistic context-aware step length estimation for pedestrian dead reckoning. *IEEE Sensors journal 18*(4), 1600–1611.

Masoud, N. and R. Jayakrishnan (2016). Formulations for optimal shared ownership and use of autonomous or driverless vehicles. In *Proceedings of the Transportation Research Board 95th Annual Meeting*, pp. 1–17.

Masoud, N. and R. Jayakrishnan (2017). Autonomous or driver-less vehicles: Implementation strategies and operational concerns. *Transportation research part E: logistics and transportation review 108*, 179–194.

Mathiassen, J. R., A. Skavhaug, and K. Bø (2002). Texture similarity measure using kullback-leibler divergence between gamma distributions. In *European conference on computer vision*, pp. 133–147. Springer.

Metz, D. (2018). Developing policy for urban autonomous vehicles: Impact on congestion. *Urban Science 2*(2), 33.

Miglani, A. and N. Kumar (2019). Deep learning models for traffic flow prediction in autonomous vehicles: A review, solutions, and challenges. *Vehicular Communications 20*, 100184.

Mizell, L., M. Joint, D. Connell, et al. (1997). Aggressive driving: Three studies.

Mnih, V., K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, et al. (2015). Human-level control through deep reinforcement learning. *nature 518*(7540), 529–533.

Møgelmose, A., M. M. Trivedi, and T. B. Moeslund (2015). Trajectory analysis and prediction for improved pedestrian safety: Integrated framework and evaluations. In *2015 IEEE Intelligent Vehicles Symposium (IV)*, pp. 330–335. IEEE.

Moody, J., N. Bailey, and J. Zhao (2020). Public perceptions of autonomous vehicle safety: An international comparison. *Safety science 121*, 634–650.

Mosenia, A., X. Dai, P. Mittal, and N. K. Jha (2017). Pinme: Tracking a smartphone user around the world. *IEEE Transactions on Multi-Scale Computing Systems 4*(3), 420–435.

Moukafih, Y., H. Hafidi, and M. Ghogho (2019). Aggressive driving detection using deep learning-based time series classification. In *2019 IEEE International Symposium on INnovations in Intelligent SysTems and Applications (INISTA)*, pp. 1–5. IEEE.

NHTSA (2017). Automated driving systems (ads): a vision for safety 2.0. *National Center for Statistics and Analysis, US Department of Transportation.[Online]. Available: https://www. nhtsa. gov/press-releases/us-dot-releases-new-automated-driving-systems-guidance*.

Nikhil, N. and B. Tran Morris (2018). Convolutional neural network for trajectory prediction. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 0–0.

Ntousakis, I. A., I. K. Nikolos, and M. Papageorgiou (2016). Optimal vehicle trajectory planning in the context of cooperative merging on highways. *Transportation research part C: emerging technologies 71*, 464–488.

Nunes, A., B. Reimer, and J. F. Coughlin (2018). People must retain control of autonomous vehicles.

Nurvitadhi, E., J. Sim, D. Sheffield, A. Mishra, S. Krishnan, and D. Marr (2016). Accelerating recurrent neural networks in analytics servers: Comparison of fpga, cpu, gpu, and asic. In *2016 26th International Conference on Field Programmable Logic and Applications (FPL)*, pp. 1–4. IEEE.

Occupancy-LSTM (2016). Model performance results - occupancy lstm on trajnet. *http://trajnet.stanford.edu/results.php?tid=250*.

Othman, K. (2021). Public acceptance and perception of autonomous vehicles: a comprehensive review. *AI and Ethics 1*(3), 355–387.

Othmane, L. B., H. Weffers, M. M. Mohamad, and M. Wolf (2015). A survey of security and privacy in connected vehicles. In *Wireless sensor and mobile ad-hoc networks*, pp. 217–247. Springer.

Pammer, K., C. Gauld, A. McKerral, and C. Reeves (2021). "they have to be better than human drivers!" motorcyclists" and cyclists" perceptions of autonomous vehicles. *Transportation research part F: traffic psychology and behaviour 78*, 246–258.

Pan, S. J. and Q. Yang (2009). A survey on transfer learning. *IEEE Transactions on knowledge and data engineering 22*(10), 1345–1359.

Pandey, V., E. Wang, and S. D. Boyles (2020). Deep reinforcement learning algorithm for dynamic pricing of express lanes with multiple access locations. *Transportation Research Part C: Emerging Technologies 119*, 102715.

Parikh, A. P., O. Täckström, D. Das, and J. Uszkoreit (2016). A decomposable attention model for natural language inference. *arXiv preprint arXiv:1606.01933*.

Park, S. H., B. Kim, C. M. Kang, C. C. Chung, and J. W. Choi (2018). Sequence-to-sequence prediction of vehicle trajectory via lstm encoder-decoder architecture. In *2018 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1672–1678. IEEE.

Patel, E. and D. S. Kushwaha (2020). Clustering cloud workloads: K-means vs gaussian mixture model. *Procedia Computer Science 171*, 158–167.

Pellegrini, S., A. Ess, and L. Van Gool (2010). Improving data association by joint modeling of pedestrian trajectories and groupings. In *European conference on computer vision*, pp. 452–465. Springer.

Penmetsa, P., E. K. Adanu, D. Wood, T. Wang, and S. L. Jones (2019). Perceptions and expectations of autonomous vehicles–a snapshot of vulnerable road user opinion. *Technological Forecasting and Social Change 143*, 9–13.

Perdisci, R., G. Gu, W. Lee, et al. (2006). Using an ensemble of one-class svm classifiers to harden payload-based anomaly detection systems. In *ICDM*, Volume 6, pp. 488–498. Citeseer.

Petitjean, F. and P. Gançarski (2012). Summarizing a set of time series by averaging: From steiner sequence to compact multiple alignment. *Theoretical Computer Science 414*(1), 76–91.

Petrović, ., R. Mijailović, and D. Pešić (2020). Traffic accidents with autonomous vehicles: type of collisions, manoeuvres and errors of conventional vehicles'' drivers. *Transportation research procedia 45*, 161–168.

Pool, E. A., J. F. Kooij, and D. M. Gavrila (2017). Using road topology to improve cyclist path prediction. In *2017 IEEE Intelligent Vehicles Symposium (IV)*, pp. 289–296. IEEE.

Prevost, C. G., A. Desbiens, and E. Gagnon (2007). Extended kalman filter for state estimation and trajectory prediction of a moving object detected by an unmanned aerial vehicle. In *2007 American control conference*, pp. 1805–1810. IEEE.

Quan, R., L. Zhu, Y. Wu, and Y. Yang (2021). Holistic lstm for pedestrian trajectory prediction. *IEEE transactions on image processing 30*, 3229–3239.

Raffo, G. V., G. K. Gomes, J. E. Normey-Rico, C. R. Kelber, and L. B. Becker (2009). A predictive controller for autonomous vehicle path tracking. *IEEE transactions on intelligent transportation systems 10*(1), 92–102.

Rahman, M. T., K. Dey, S. Das, and M. Sherfinski (2021). Sharing the road with autonomous vehicles: A qualitative analysis of the perceptions of pedestrians and bicyclists. *Transportation research part F: traffic psychology and behaviour 78*, 433–445.

Raju, V. M., V. Gupta, and S. Lomate (2019). Performance of open autonomous vehicle platforms: Autoware and apollo. In *2019 IEEE 5th International Conference for Convergence in Technology (I2CT)*, pp. 1–5. IEEE.

Rasouli, A., I. Kotseruba, T. Kunic, and J. K. Tsotsos (2019). 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*, pp. 6262–6271.

Rasouli, A. and J. K. Tsotsos (2019). Autonomous vehicles that interact with pedestrians: A survey of theory and practice. *IEEE Transactions on Intelligent Transportation Systems 21*(3), 900–918.

Recht, B. (2019). A tour of reinforcement learning: The view from continuous control. *Annual Review of Control, Robotics, and Autonomous Systems 2*, 253–279.

Rehder, E., F. Wirth, M. Lauer, and C. Stiller (2018). Pedestrian prediction by planning using deep neural networks. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1–5. IEEE.

Rousseeuw, P. J. and K. V. Driessen (1999). A fast algorithm for the minimum covariance determinant estimator. *Technometrics 41*(3), 212–223.

Rudenko, A., L. Palmieri, M. Herman, K. M. Kitani, D. M. Gavrila, and K. O. Arras (2020). Human motion trajectory prediction: A survey. *The International Journal of Robotics Research 39*(8), 895–935.

Sadat, A., S. Casas, M. Ren, X. Wu, P. Dhawan, and R. Urtasun (2020). Perceive, predict, and plan: Safe motion planning through interpretable semantic representations. In *European Conference on Computer Vision*, pp. 414–430. Springer.

Sadeghian, A., V. Kosaraju, A. Gupta, S. Savarese, and A. Alahi (2018). Trajnet: Towards a benchmark for human trajectory prediction. *arXiv preprint*.

Sak, H., M. Shannon, K. Rao, and F. Beaufays (2017). Recurrent neural aligner: An encoder-decoder neural network model for sequence to sequence mapping. In *Interspeech*, Volume 8, pp. 1298–1302.

Sakoe, H. and S. Chiba (1978). Dynamic programming algorithm optimization for spoken word recognition. *IEEE transactions on acoustics, speech, and signal processing 26*(1), 43–49.

Saleh, K., M. Hossny, and S. Nahavandi (2017). Towards trusted autonomous vehicles from vulnerable road users perspective. In *2017 Annual IEEE International Systems Conference (SysCon)*, pp. 1–7. IEEE.

Saleh, K., M. Hossny, and S. Nahavandi (2018). Cyclist trajectory prediction using bidirectional recurrent neural networks. In *Australasian Joint Conference on Artificial Intelligence*, pp. 284–295. Springer.

Schulz, J., C. Hubmann, J. Löchner, and D. Burschka (2018). Multiple model unscented kalman filtering in dynamic bayesian networks for intention estimation and trajectory prediction. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pp. 1467–1474. IEEE.

Shi, X., Y. D. Wong, C. Chai, and M. Z.-F. Li (2020). An automated machine learning (automl) method of risk prediction for decision-making of autonomous vehicles. *IEEE Transactions on Intelligent Transportation Systems*.

Shi, X., Y. D. Wong, M. Z.-F. Li, C. Palanisamy, and C. Chai (2019). A feature learning approach based on xgboost for driving assessment and risk prediction. *Accident Analysis & Prevention 129*, 170–179.

Shiller, Z., Y.-R. Gwo, et al. (1991). Dynamic motion planning of autonomous vehicles. *IEEE Transactions on Robotics and Automation 7*(2), 241–249.

Shinar, D. and R. Compton (2004). Aggressive driving: an observational study of driver, vehicle, and situational variables. *Accident Analysis & Prevention 36*(3), 429–437.

Siegel, J. E., D. C. Erb, and S. E. Sarma (2017). A survey of the connected vehicle landscape‘‘architectures, enabling technologies, applications, and development areas. *IEEE Transactions on Intelligent Transportation Systems 19*(8), 2391–2406.

Social-GAN (2018). Model performance results - social gan on trajnet. *http://trajnet.stanford.edu/results.php?tid=638*.

Social-LSTM (2016). Model performance results - social lstm_v2 on trajnet. *http://trajnet.stanford.edu/results.php?tid=693*.

Song, S., X. Hu, J. Yu, L. Bai, and L. Chen (2018). Learning a deep motion planning model for autonomous driving. In *2018 Ieee Intelligent Vehicles Symposium (IV)*, pp. 1137–1142. IEEE.

Song, X., K. Chen, X. Li, J. Sun, B. Hou, Y. Cui, B. Zhang, G. Xiong, and Z. Wang (2020). Pedestrian trajectory prediction based on deep convolutional lstm network. *IEEE Transactions on Intelligent Transportation Systems 22*(6), 3285–3302.

Staats, A. W. and C. K. Staats (1963). Complex human behavior: A systematic extension of learning principles.

Sun, L., S. Versteeg, S. Boztas, and A. Rao (2016). Detecting anomalous user behavior using an extended isolation forest algorithm: an enterprise case study. *arXiv preprint arXiv:1609.06676*.

Sun, L., Z. Yan, S. M. Mellado, M. Hanheide, and T. Duckett (2018). 3dof pedestrian trajectory prediction learned from long-term autonomous mobile robot deployment data. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1–7. IEEE.

Sutskever, I., O. Vinyals, and Q. V. Le (2014). Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pp. 3104–3112.

Takahashi, J. (2018). An overview of cyber security for connected vehicles. *IEICE TRANSACTIONS on Information and Systems 101*(11), 2561–2575.

Taylor, J., X. Zhou, N. M. Rouphail, and R. J. Porter (2015). Method for investigating intradriver heterogeneity using vehicle trajectory data: A dynamic time warping approach. *Transportation Research Part B: Methodological 73*, 59–80.

Tian, Q., I. Kevin, K. Wang, and Z. Salcic (2019). A low-cost ins and uwb fusion pedestrian tracking system. *IEEE Sensors Journal 19*(10), 3733–3740.

Toghi, B., R. Valiente, D. Sadigh, R. Pedarsani, and Y. P. Fallah (2021). Cooperative autonomous vehicles that sympathize with human drivers. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4517–4524. IEEE.

Tran, H., V. Le, and T. Tran (2021). Goal-driven long-term trajectory prediction. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 796–805.

Tsuchiya, C., S. Takei, K. Sakai, and A. Khiat (2021). Exemplar trajectory generation for prior driving experience re-usage in autonomous driving. In *2021 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1001–1007. IEEE.

USDOT (2018, Jul). Intersection safety.

van Wyk, F., A. Khojandi, and N. Masoud (2019). A path towards understanding factors affecting crash severity in autonomous vehicles using current naturalistic driving data. In *Proceedings of SAI Intelligent Systems Conference*, pp. 106–120. Springer, Cham.

van Wyk, F., A. Khojandi, and N. Masoud (2020). Optimal switching policy between driving entities in semi-autonomous vehicles. *Transportation Research Part C: Emerging Technologies 114*, 517–531.

Vaswani, A., N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin (2017). Attention is all you need. In *Advances in neural information processing systems*, pp. 5998–6008.

Vemula, A., K. Muelling, and J. Oh (2018). Social attention: Modeling attention in human crowds. In *2018 IEEE international Conference on Robotics and Automation (ICRA)*, pp. 4601–4607. IEEE.

Wali, B., A. J. Khattak, H. Bozdogan, and M. Kamrani (2018). How is driving volatility related to intersection safety? a bayesian heterogeneity-based analysis of instrumented vehicles data. *Transportation research part C: emerging technologies 92*, 504–524.

Wang, F., M. Jiang, C. Qian, S. Yang, C. Li, H. Zhang, X. Wang, and X. Tang (2017). Residual attention network for image classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3156–3164.

Wang, J., W. Xu, and Y. Gong (2010). Real-time driving danger-level prediction. *Engineering Applications of Artificial Intelligence 23*(8), 1247–1254.

Wang, N., X. Wang, P. Palacharla, and T. Ikeuchi (2017). Cooperative autonomous driving for traffic congestion avoidance through vehicle-to-vehicle communications. In *2017 IEEE Vehicular Networking Conference (VNC)*, pp. 327–330. IEEE.

Wang, R. and S. M. Lukic (2011). Review of driving conditions prediction and driving style recognition based control algorithms for hybrid electric vehicles. In *2011 IEEE Vehicle Power and Propulsion Conference*, pp. 1–7. IEEE.

Wang, X., Z. Yu, and S. Mao (2018). Deepml: Deep lstm for indoor localization with smartphone magnetic and light sensors. In *2018 IEEE international conference on communications (ICC)*, pp. 1–6. IEEE.

Wang, Y., N. Masoud, and A. Khojandi (2020). Real-time sensor anomaly detection and recovery in connected automated vehicle sensors. *IEEE transactions on intelligent transportation systems 22*(3), 1411–1421.

Wang, Y., W. Xu, Y. Zhang, Y. Qin, W. Zhang, and X. Wu (2017). Machine learning methods for driving risk prediction. In *Proceedings of the 3rd ACM SIGSPATIAL Workshop on Emergency Management using*, pp. 1–6.

Woodhouse, K. (2014, Apr). U-m is turning ann arbor into the world's largest lab for wireless vehicle communication.

Wu, F., R. E. Stern, S. Cui, M. L. Delle Monache, R. Bhadani, M. Bunting, M. Churchill, N. Hamilton, B. Piccoli, B. Seibold, et al. (2019). Tracking vehicle trajectories and fuel rates in phantom traffic jams: Methodology and data. *Transportation Research Part C: Emerging Technologies 99*, 82–109.

Xiangguo Liu, Guangchen Zhao, N. M. Q. Z. (2021). Trajectory planning for connected and automated vehicles: Cruising, lane changing, and platooning. *SAE Intl. J CAV 4*(4), 315–333.

Xiao, L., Y. Zheng, W. Tang, G. Yao, and L. Ruan (2013). Parallelizing dynamic time warping algorithm using prefix computations on gpu. In *2013 IEEE 10th International Conference on High Performance Computing and Communications & 2013 IEEE International Conference on Embedded and Ubiquitous Computing*, pp. 294–299. IEEE.

Xie, G., H. Gao, L. Qian, B. Huang, K. Li, and J. Wang (2017). Vehicle trajectory prediction by integrating physics-and maneuver-based approaches using interactive multiple models. *IEEE Transactions on Industrial Electronics 65*(7), 5999–6008.

Xu, W., J. Wang, T. Fu, H. Gong, and A. Sobhani (2022). Aggressive driving behavior prediction considering driver‘‘s intention based on multivariate-temporal feature data. *Accident Analysis & Prevention 164*, 106477.

Xu, Y., Z. Piao, and S. Gao (2018). Encoding crowd interaction with deep neural network for pedestrian trajectory prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5275–5284.

Xue, H., D. Huynh, and M. Reynolds (2019). Location-velocity attention for pedestrian trajectory prediction. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 2038–2047. IEEE.

Xue, H., D. Q. Huynh, and M. Reynolds (2018). Ss-lstm: A hierarchical lstm model for pedestrian trajectory prediction. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 1186–1194. IEEE.

Yan, C., Y. Tu, X. Wang, Y. Zhang, X. Hao, Y. Zhang, and Q. Dai (2019). Stat: spatial-temporal attention mechanism for video captioning. *IEEE transactions on multimedia 22*(1), 229–241.

Yao, Y., E. Atkins, M. Johnson-Roberson, R. Vasudevan, and X. Du (2021). Bitrap: Bi-directional pedestrian trajectory prediction with multi-modal goal estimation. *IEEE Robotics and Automation Letters 6*(2), 1463–1470.

Ye, F., S. Zhang, P. Wang, and C.-Y. Chan (2021). A survey of deep reinforcement learning algorithms for motion planning and control of autonomous vehicles. In *2021 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1073–1080. IEEE.

Ye, Y., Z. Zhao, Y. Li, L. Chen, J. Xiao, and Y. Zhuang (2017). Video question answering via attribute-augmented attention network learning. In *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*, pp. 829–832.

Yoon, S. and D. Kum (2016). The multilayer perceptron approach to lateral motion prediction of surrounding vehicles for autonomous vehicles. In *2016 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1307–1312. IEEE.

Yu, N., Y. Li, X. Ma, Y. Wu, and R. Feng (2019). Comparison of pedestrian tracking methods based on foot-and waist-mounted inertial sensors and handheld smartphones. *IEEE Sensors Journal 19*(18), 8160–8173.

Zacharias, J. (2001). Pedestrian behavior pedestrian behavior and perception in urban walking environments. *Journal of Planning Literature 16*(1), 3–18.

Zanlungo, F., T. Ikeda, and T. Kanda (2011). Social force model with explicit collision prediction. *EPL (Europhysics Letters) 93*(6), 68005.

Zernetsch, S., S. Kohnen, M. Goldhammer, K. Doll, and B. Sick (2016). Trajectory prediction of cyclists using a physical model and an artificial neural network. In *2016 IEEE Intelligent Vehicles Symposium (IV)*, pp. 833–838. IEEE.

Zhang, E. and N. Masoud (2020). Increasing gps localization accuracy with reinforcement learning. *IEEE Transactions on Intelligent Transportation Systems 22*(5), 2615–2626.

Zhang, E. and N. Masoud (2021). Increasing gps localization accuracy with reinforcement learning. *IEEE Transactions on Intelligent Transportation Systems 22*(5), 2615–2626.

Zhang, E., N. Masoud, M. Bandegi, J. Lull, and R. K. Malhan (2022). Step attention: Sequential pedestrian trajectory prediction. *IEEE Sensors Journal*.

Zhang, E., N. Masoud, M. Bandegi, and R. K. Malhan (2022). Predicting risky driving in a connected vehicle environment. *IEEE Transactions on Intelligent Transportation Systems*.

Zhang, E., N. Masoud, W. Zhang, and R. K. Malhan (2021, April 29). System for predicting aggressive driving. US Patent App. 17/006,470.

Zhang, E., S. Pizzi, and N. Masoud (2021). A learning-based method for predicting heterogeneous traffic agent trajectories: Implications for transfer learning. In *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, pp. 1853–1858. IEEE.

Zhang, E., A. Tafreshian, and N. Masoud (2019). Parallel computing algorithm for real-time mapping between large-scale networks. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pp. 4087–4092. IEEE.

Zhang, P., W. Ouyang, P. Zhang, J. Xue, and N. Zheng (2019). Sr-lstm: State refinement for lstm towards pedestrian trajectory prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 12085–12094.

Zhang, S., W. Deng, Q. Zhao, H. Sun, and B. Litkouhi (2013). Dynamic trajectory planning for vehicle autonomous driving. In *2013 IEEE International Conference on Systems, Man, and Cybernetics*, pp. 4161–4166. IEEE.

Zhang, Z., F. Liu, B. Wolshon, and Y. Sheng (2020). Virtual traffic signals: Safe, rapid, efficient and autonomous driving without traffic control. *IEEE Transactions on Intelligent Transportation Systems 22*(11), 6954–6966.

Zhang, Z., A. Tafreshian, and N. Masoud (2020). Modular transit: Using autonomy and modularity to improve performance in public transportation. *Transportation Research Part E: Logistics and Transportation Review 141*, 102033.

Zhao, C., L. Li, X. Pei, Z. Li, F.-Y. Wang, and X. Wu (2021). A comparative study of state-of-the-art driving strategies for autonomous vehicles. *Accident Analysis & Prevention 150*, 105937.

Zhao, T., Y. Xu, M. Monfort, W. Choi, C. Baker, Y. Zhao, Y. Wang, and Y. N. Wu (2019). Multi-agent tensor fusion for contextual trajectory prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 12126–12134.

Zhou, J., R. He, Y. Wang, S. Jiang, Z. Zhu, J. Hu, J. Miao, and Q. Luo (2020). Autonomous driving trajectory optimization with dual-loop iterative anchoring path smoothing and piecewise-jerk speed optimization. *IEEE Robotics and Automation Letters 6*(2), 439–446.

Zhou, Y., M. E. Cholette, A. Bhaskar, and E. Chung (2018). Optimal vehicle trajectory planning with control constraints and recursive implementation for automated on-ramp merging. *IEEE Transactions on Intelligent Transportation Systems 20*(9), 3409–3420.

Zhu, M., X. Wang, and Y. Wang (2018). Human-like autonomous car-following model with deep reinforcement learning. *Transportation research part C: emerging technologies 97*, 348–368.

Ziebart, B. D., N. Ratliff, G. Gallagher, C. Mertz, K. Peterson, J. A. Bagnell, M. Hebert, A. K. Dey, and S. Srinivasa (2009). Planning-based prediction for pedestrians. In *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3931–3936. IEEE.