Custom AI Architectures for Predictive Analytics Using Bayesian Statistics and Deep Learning

by

Mayuresh Savargaonkar

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Industrial and Systems Engineering)
in the University of Michigan-Dearborn
2023

Doctoral Committee:

Assistant Professor Abdallah Chehade, Chair
Assistant Professor Fred Feng
Assistant Professor Zhen Hu
Associate Professor Samir Rawashdeh

Mayuresh V. Savargaonkar

mayuresh@umich.edu

ORCID iD: 0000-0001-6958-4882

Dedication

This dissertation is dedicated to my parents, Dr. Ujwala Savargaonkar and Dr. Vijay Savargaonkar. This work would not be possible without their belief, love, and support.

Acknowledgments

The completion of this work was made possible through the assistance and support of several individuals, only some of whom can be mentioned here.

Foremost, I express my gratitude to Dr. Abdallah Chehade, my advisor and mentor throughout my Ph.D. program, for his invaluable guidance, wisdom, and unparalleled patience in supporting me. His extensive knowledge and insightful suggestions have had a profound impact on both my research and personal growth. I also extend my sincere appreciation to my committee members, namely Dr. Fred Feng, Dr. Zhen Hu, and Dr. Samir Rawashdeh, whose valuable advice and productive discussions were integral to the successful completion of this work. I am also grateful to the Department of Industrial and Manufacturing Systems Engineering at the University of Michigan - Dearborn for their financial and academic support during my doctoral studies. Additionally, I would like to thank the Ford Motor Company for allowing me to work on sponsored research projects, which provided valuable learning opportunities and greatly enhanced the quality of this work. Special thanks are due to Mr. Sari Kassar, and Mr. Mark Malone of the Ford Motor Company for their continuous support and valuable input. Personally, I wish to express my gratitude to my brother Mr. Prathmesh Savargaonkar and sister-in-law Dr. Srujana Goteti for their immense support, and to my friends Neeraj, Prakash, Abhishek, Sanket, Ruturaj, Archana, and Lilit for their unwavering encouragement throughout my studies.

Table of Contents

Dedication ii

Acknowledgments iii

List of Tables viii

List of Figures x

List of Appendices xiii

Abstract xiv

Chapter 1 Introduction 1

1.1 Motivation and Overview 1

1.2 Limitations of Existing Approaches 3

1.3 Goals and Objectives 5

Chapter 2 USAL: Uncorrelated Sparse LSTM Autoencoders for Robust Long-term Estimations 9

2.1 Introduction 9

2.2 Uncorrelated Sparse Autoencoder with Long-short-term Memory (USAL) 14

2.2.1 Sparse Autoencoders 14

2.2.2 Correlation Analysis 16

2.2.3 Prediction Network 17

2.3 Case Study 18

2.3.1 Datasets 18

2.3.2 Training and Tuning USAL 21

2.3.3 Comparison Across Battery Configurations 23

2.3.4 Comparisons Across Varying Charge, Discharge, and Temperature Conditions 25

2.3.5 Short-term SOC Estimations 28

2.3.6 Effects of Correlations on Long-term SOC Estimations 29

2.3.7 Robustness Evaluation 30

2.3.8 SOC Estimations for Dynamic Loading Profiles 31

2.4 Conclusion 34

Chapter 3 CGMM: Conditional Gaussian Mixture Model for Predictive Analytics Using Bayesian Statistics 35

3.1 Introduction 36

3.2 Gaussian Mixture Models for Warranty Claims Forecasting 39

3.3 Conditional Gaussian Mixture Model for Warranty Claims Forecasting 42

3.3.1 Problem Formulation 43

3.3.2 Conditional Predictive Distribution Using CGMM 44

3.3.3 Unbiased Conditional Predictive Distribution 45

3.4 Case Study Using Early Launched Automotive Parts 45

3.4.1 Description 45

3.4.2 Model Setup 47

3.4.3 Results 48

3.4.4 Comparison to Standard NHPP 50

3.4.5 Comparison to Benchmarks in Warranty Forecasting 53

3.4.6 Comparison to Recurrent Neural Networks 56

3.5 Conclusion 58

Chapter 4 BOSE: A Bayesian Architecture for Verification and Validation of Autonomous Vehicles 60

4.1 Introduction 61

4.2 Related Work 64

4.2.1 Scenario Generation Approaches 64

4.2.2 Simulation Platforms 66

4.3 Bayesian Optimization for Scenario Exploration (BOSE) 67

4.3.1 The Non-sequential Case 67

4.3.2 The Sequential Case 69

4.3.3 Sequential Design using Bayesian Deep Neural Networks 70

4.4 Case Study Using A Custom ADS 73

4.4.1 Low-Dimensional Scenario 74

4.4.2 High-Dimensional Scenario 77

4.5 Conclusion 81

Chapter 5 Collaborative Modeling using Infrastructure Information for Improved Predictive Analytics in Autonomous Vehicles 82

5.1 Introduction 83

5.2 Existing Datasets and Related Work 85

5.2.1 Real-world Driving Datasets 85

5.2.2 Synthetic Driving Datasets 87

5.3 The VTrackIt Dataset 88

5.3.1 Sensing Package, Lane Annotations, and Aerial BEV 89

5.3.2 Diversity in Weather and Typologies 93

5.3.3 Traffic and Ego Vehicle Customizations 94

5.3.4 Comparisons to Real-world Datasets 95

5.4 Case Study for Trajectory Predictions 96

5.4.1 InfraGAN: PGAN Module 97

5.4.2 InfraGAN: Correction Module (CM) 99

5.4.3 Trajectory Prediction Benchmarks 100

5.4.4 Ego Trajectory Prediction Benchmarks 103

5.5 Conclusion 104

Chapter 6 Summary of Original Contributions and Future Work 105

6.1 Summary of Original Contributions 105

6.2 Future Work 107

Appendices 109

Bibliography 116

List of Tables

Table 2-1 Summary of Accelerated Aging Datasets 18

Table 2-2 Summary of Train-Test Data 22

Table 2-3 Summary of results under varying cell configurations 24

Table 2-4 Cells and their Operating Conditions 24

Table 2-5 Summary of Results Under Varying Cell Configurations 25

Table 2-6 Summary of Results Under Varying Conditions 28

Table 2-7 Effectiveness of Correlation Loss 30

Table 2-8 Robustness of USAL 31

Table 2-9 Comparison of USAL with State-of-the-art 33

Table 3-1 Training and Validation Data 46

Table 3-2 Log-likelihood and AIC as a function of K Latent Clusters 47

Table 3-3 Selected Quantiles for the Error Distributions 48

Table 3-4 Comparison with Traditional Warranty Forecasting Models 54

Table 3-5 Selected Quantiles for the Error Distributions of the RNN Predictions for TIS 12 and 18 Months 57

Table 3-6 The MAPE and RMSE Values for the Predictions of the CGMM and RNN at TIS 12 and 18 Months 57

Table 4-1 Levels of Autonomy 61

Table 4-2 Comparison of Common Simulation Platforms used for Scenario Generation 66

Table 4-3 Algorithm I: BOSE 72

Table 4-4 Parameter Ranges for Scenario A 74

Table 4-5 Comparison of Performance for Scenario A 76

Table 4-6 Parameter Ranges for Scenario B.. 78

Table 4-7 Comparison of Performance for Scenario B 80

Table 5-1 Comparison of Major Driving Datasets for Autonomous Vehicle Applications 88

Table 5-2 Sensing Package 89

Table 5-3 Details of Variables Recorded at Every Frame in the VTrackIt Dataset 92

Table 5-4 Map and Road Features 93

Table 5-5 Diversity of VTrackIt in Comparison with Commonly Used Self-driving Datasets ... 96

Table 5-6 Trajectory Prediction Benchmarks Using Top k Predictions.................................... 102

Table 5-7 Comparison of Benchmark Models on the VTrackIt Test Set Per Map..................... 102

Table 5-8 Comparison of Benchmark Models on the VTrackIt Test Set Over Varying Prediction Horizons 102

Table 5-9 Ego Trajectory Prediction Benchmarks Using Top k Predictions on the VTRACKIT Test Set... 103

List of Figures

Figure 1-1 An overview of the larger picture of predictive analytics 1

Figure 1-2 An overview of the goals of this work. 5

Figure 1-3 USAL: A deep learning architecture for long-term estimations. 6

Figure 1-4 CGMM: A custom AI architecture for negotiating data drift. 6

Figure 1-5 BOSE: A novel data-driven frame that models and predicts black-swan events in an AI-driven systems. 7

Figure 1-6 InfraGAN: A custom DL model that performs collaborative modeling. 8

Figure 2-1 High-level representation of a battery management system in an intelligent vehicle. 10

Figure 2-2 Under-complete and over-complete autoencoders. 15

Figure 2-3 Uncorrelated Sparse Autoencoder with Long Short-Term Memory (USAL) 18

Figure 2-4 Charge and discharge profiles of all cells 21

Figure 2-5 Flow of information within USAL and its loss components 21

Figure 2-6 Long-term SOC estimation validation performance 23

Figure 2-7 Comparison of USAL with benchmarked models 26

Figure 2-8 Long-term SOC estimations for final cycle near EOL 27

Figure 2-9 Comparisons for short-term and long-term SOC estimations 29

Figure 2-10 SOC estimation performance of USAL 33

Figure 3-1 Effect of the warranty data maturation phenomenon 37

Figure 3-2 The Gaussian Mixture Model (GMM) for warranty forecasting 40

Figure 3-3 Proposed CGMM for warranty claims forecasting. 43

Figure 3-4 Example validation part used in this study 47

Figure 3-5 The distribution of PE 49

Figure 3-6 Cumulative number of claims for validation part 1 50

Figure 3-7 Cumulative number of claims for validation part 2 51

Figure 3-8 Cumulative number of claims for validation part 3 52

Figure 3-9 Cumulative number of claims for validation part 4 52

Figure 3-10 Cumulative number of claims for validation part 5 53

Figure 3-11 The distribution for MAPE of CGMM 54

Figure 3-12 The cumulative number of claims for several validation parts as a function of the TIS (months). 55

Figure 3-13 The distribution for the MAPE of the CGMM and RNN for TIS 18 months.......... 57

Figure 4-1 Scope of ISO 21448 and ISO 262262 for safety of ADSs.. 62

Figure 4-2 Formalized scenario definitions used in this work.. 63

Figure 4-3 Overview of the proposed BOSE framework in autonomous vehicles...................... 72

Figure 4-4 Underlying scenario typology for Scenario A. 74

Figure 4-5 Performance comparison of BOSE with other benchmarked methods for functional Scenario A.. 75

Figure 4-6 Progression of decision boundaries... 76

Figure 4-7 Underlying scenario typology for Scenario B... 78

Figure 4-8 Performance comparison of BOSE with other benchmarked methods for Scenario B. 79

Figure 4-9 Risky clusters with high probability of crashes identified by BOSE......................... 81

Figure 5-1 An overview of the VTrackIt dataset with infrastructure and pooled vehicle information.. 83

Figure 5-2 Co-ordinate system for data recorded and locations of sensors with their respective sensor codes in the VTrackIt dataset. 90

Figure 5-3 360-degree perception coverage around the ego vehicle using four RGB cameras. .. 90

Figure 5-4 Diverse set of weather and typologies included in the VTrackIt dataset.................. 93

Figure 5-5 Comparison of VTrackIt dataset 95

Figure 5-6 Overview of the InfraGAN 97

Figure 5-7 Performance comparison with and without infrastructure and pooled vehicle information 101

Figure 5-8 Comparison of error distributions with and without infrastructure and pooled vehicle information 103

Figure 5-9 Comparison of baseline models using the VTrackIt test set. 103

Figure B-1 Sample concrete scenarios and their predicted crash probability by BOSE for functional Scenario A and Scenario B from their respective test sets. 115

List of Appendices

Appendix A Expectation-Maximization (EM) in CGMM..110

Appendix B Experimental Evaluation of BOSE...114

Abstract

Predictive analytics has emerged as a vital field with significant potential in industries ranging from energy to mobility. As such, it has become a topic of considerable interest for research. Through the use of statistical models, predictive analytics helps reveal patterns and relationships in complex datasets, generating accurate predictions about future events or outcomes. The development of Artificial Intelligence (AI) architectures and data-driven frameworks has further revolutionized the way we perform predictive analytics. However, the broad adoption of AI for predictive analytics is limited due to the lack of custom architectures that can effectively handle the unique complexities of modern datasets and perform robust and accurate predictions.

As datasets grow increasingly complex, the need for Bayesian statistics and Deep Learning (DL) in predictive analytics has become increasingly evident. Bayesian statistics offers a versatile framework for incorporating prior knowledge and external knowledge into AI models. This can help mitigate problems such as data sparsity and improve long-term forecasts. Similarly, DL architectures, with their ability to identify and learn complex patterns within datasets, have the potential to unlock new insights and drive innovation in predictive analytics. However, the development of custom AI architectures that leverage such techniques for predictive analytics remains challenging due to their several inherent limitations.

This work aims to bridge this research gap by harnessing the power of Bayesian statistics and DL to advance the state-of-the-art in predictive analytics. Specifically, this work proposes custom AI architectures and data-driven frameworks that can (i) perform accurate long-term estimations, (ii) overcome data drift, (iii) provide uncertainty quantifications, (iv) model and predict anomalous behavior, (v) leverage concepts of Design of Experiments, and (vi) perform collaborative modeling. The proposed models and frameworks are evaluated using compelling case studies that demonstrate their effectiveness in improving the accuracy, reliability, and robustness of AI architectures for broader use in predictive analytics.

Chapter 1 Introduction

1.1 Motivation and Overview

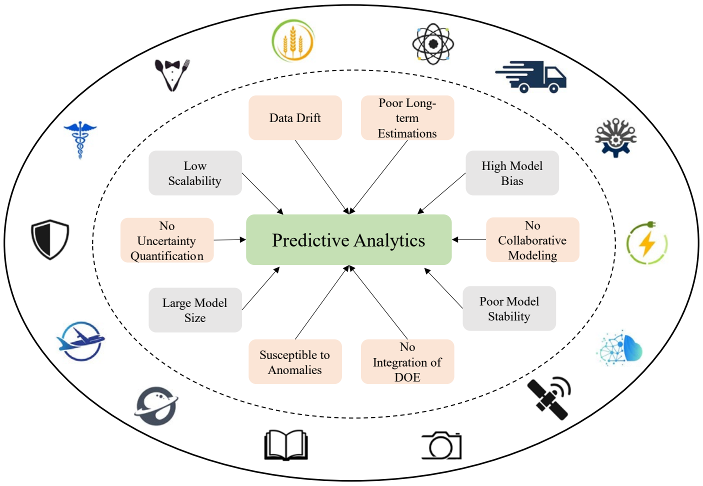


Figure 1-1 An overview of the larger picture of predictive analytics. While the outer-arc shows its multi-domain applications, the inner arc highlights some challenges. We focus on the six highlighted challenges in this work.

Predictive analytics has a rich history dating back to the field of statistics, where models are developed to make inferences about future events based on past data [1]–[4]. Therefore, predictive analytics has become a ubiquitous tool for a variety of industries, including transportation, healthcare, energy, and manufacturing, among others [5]–[7]. Moreover, the rise of big data and the availability of advanced computing resources have led to a rapid evolution of predictive analytics, enabling organizations to develop more accurate predictions and improve decision-making processes [7]–[11]. By leveraging data from various sources, predictive analytics helps organizations to identify patterns and trends that would be difficult to discern otherwise. This, in turn, allows them to make more informed decisions and take proactive measures to mitigate risks and capitalize on opportunities [10]. In addition, the rise of Artificial Intelligence (AI) has enabled the development of powerful predictive models [12]–[14]. Moreover, the advent of Deep Learning (DL) has revolutionized how we approach predictive analytics, making it possible to process and analyze vast amounts of data in real-time and uncover new insights that would be difficult or impossible to identify using traditional statistical models [15]–[17].

While traditional AI architectures have shown remarkable success in many applications, their adoption in predictive analytics is limited due to several challenges [17]. One of the main challenges is the lack of custom AI architectures that can perform robust long-term predictive analytics [18]. While traditional DL models can identify and learn complex patterns within datasets, they require large amounts of data [19]–[22]. Such models are therefore limited when tasked to perform long-term estimations given limited training data [18]. Moreover, these models are often trained on historical data, which may not reflect the current or future state of the modeled system [23]. Such data drift leads to inaccurate predictions and limit the applicability of the developed models [24]. These problems are further worsened by their lack of ability to perform uncertainty quantifications [12], [25]. Issues such as lack of training data add fuel to this fire and the developed models are often exhibit anomalous behavior when exposed to unseen data [26]–[28]. This is primarily due to the lack use of a Design of Experiments (DOE) style frameworks that help verify and validate the developed models [26], [29]. Finally, predictive models that use traditional AI architectures often lack the ability to perform collaborative modeling [30]–[33]. They are thus unable to leverage and combine multimodal data from various sources for effective predictive analytics [34], [35]. This leads to inaccurate predictions and hampers the broad adoption of AI in predictive analytics.

To overcome these challenges, we propose several custom AI architectures that can (i) perform accurate long-term estimations, (ii) overcome data drift, (iii) provide uncertainty quantifications, (iv) identify and predict anomalous behavior, (v) leverage concepts of Design of

Experiments, and (vi) perform collaborative modeling, using concepts of Bayesian statistics and DL. As shown in Figure 1-1, the architectures proposed in this work focus on the six mentioned challenges that may be extended to multiple applications across several industries and domains.

1.2 Limitations of Existing Approaches

The development of AI has revolutionized the way we perform predictive analytics [36]–[38]. Although many AI architectures have been proposed, several challenges remain [17]. Some of these are shown in Figure 1-1.

One significant limitation of traditional AI architectures is their inability to perform long-term estimations [39]. While short-term estimation involves predicting the immediate future, long-term estimation often involves predictions near the End-of-Life (EOL) [18]. Therefore, traditional AI architectures such as Recurrent Neural Networks (RNNs) [40] and Long Short-Term Memory Networks (LSTMs) [41] fail when performing long-term estimations given limited training data [42], [43]. Although the concepts of Transfer Learning (TL) [44] attempt to partially solve this problem, these architectures often require hours of re-training on large amounts of training data [43]. Long-term estimations are essential for predicting the health and performance of various components, including batteries [45], [46] and other parts [21], [47], [48]. Therefore, it is critical to develop AI architectures that can perform accurate long-term estimations given limited training data while being robust and generalizable.

Manufacturers often collect and analyze data from various products and their components to enhance reliability and customer satisfaction [49], [50]. This allows them to improve the quality of the manufactured products by identifying patterns and trends in the millions of manufactured parts and components [51]. However, data trends are often a complex function that change with time [23]. This is commonly referred to as “data drift” [23]. This is especially the case in manufacturing environments where the products vary due to inherent variation in the manufacturing process [52]. Although some work has been done to address this problem, most predictive methods only work for a shorter horizon and fail to provide uncertainty quantifications [53], [54]. As such, there is a need for custom AI architectures that can effectively leverage Bayesian strategies to accommodate data drift and perform effective predictive analytics.

"Anomalous behavior" refers to the rare, unpredictable and untoward behavior that is impossible to predict using traditional statistical models [26], [75]–[77]. In the context of predictive analytics using AI, anomalous behavior or events can pose a significant challenge, as traditional AI architectures are often developed based on historical data that may not include such events [59], [78], [79]. Therefore, one has a tendency to build a false sense of confidence in the built models or systems [26], [59], [80]. Although some work has been done to build data-driven frameworks that can predict such behavior, they often use over-simplified test cases [59], [78], [79], [81]–[86]. As such, there is a need to develop custom data-driven frameworks that can effectively verify and validate AI-driven systems by modeling and predicting such behavior [87].

Modern machines such as Autonomous Vehicles (AVs) are expected to operate and perform predictive analytics in complex and dynamic environments [55]–[57]. Therefore, more recently, the use of Generative models have become quite prevalent in such applications [58]–[64]. As the name suggests, Generative models are trained to "generate" new data samples that are similar to the input data distribution [34], [65]. However, a significant limitation of Generative models is their lack of ability to integrate multimodal data from various sources [20], [66]–[69]. Unlike traditional statistical models, Generative models do not explicitly incorporate such data, thus making them vulnerable to anomalous behavior [70], [71]. This is particularly problematic in applications where the cost of a wrong decision can be significant, such as AVs [34], [35], [65],

[72]–[74]. Therefore, there is a need to develop custom AI architectures that can perform collaborative modeling by integrating multimodal data from several sources for improved predictive analytics, especially in AVs.

1.3 Goals and Objectives

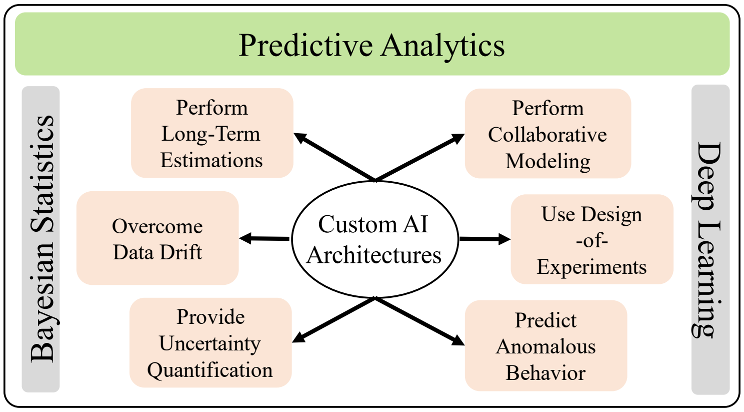

Predictive Analytics
Bayesian Statistics
Deep Learning
Custom AI Architectures
Perform Long-Term Estimations
Perform Collaborative Modeling
Overcome Data Drift
Use Design -of- Experiments
Provide Uncertainty Quantification
Predict Anomalous Behavior

Figure 1-2 An overview of the goals of this work.

The use of custom AI architectures have the potential to improve predictive analytics [88]. However, to fully realize this potential, there is a need to overcome the limitations of traditional AI architectures by developing custom robust, and accurate architectures that can (i) perform accurate long-term estimations, (ii) overcome data drift, (iii) provide uncertainty quantifications, (iv) identify and predict anomalous behavior, (v) leverage concepts of Design of Experiments, and (vi) perform collaborative modeling. Therefore, in this work, we attempt to propose custom AI architectures that leverage Bayesian statistics and DL to solve the six mentioned challenges in predictive analytics. As shown in Figure 1-2, the efficacy of the proposed architectures is validated using compelling case-studies for different applications in the automotive domain.

More specifically, in Chapter 2 of this work, we focus on developing a custom-built AI architecture to address the challenging task of long-term estimations. The proposed DL method, USAL [84], uses a multi-task learning strategy to force orthogonality in an autoencoder given limited training data. It works by penalizing highly correlated encodings in an autoencoder to efficiently transform measured inputs into a space of informative features. Finally, it leverages these uncorrelated learned encodings to support long-term State-of-Charge (SOC) estimations in Lithium-ion (Li-ion) batteries of electric vehicles. Figure 1-3 shows the architecture of USAL.

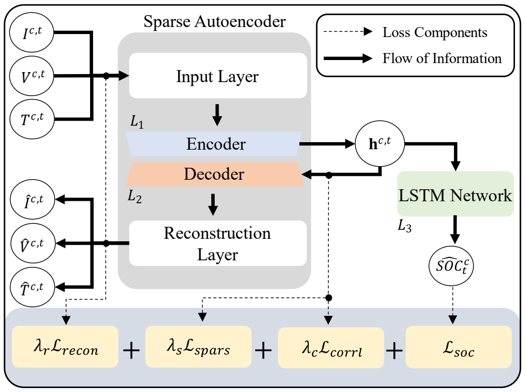


Figure 1-3 USAL: A deep learning architecture for long-term estimations.

In Chapter 3, we develop a custom AI architecture that leverages Bayesian theories to negotiate data drift [52]. Here, we explicitly consider the data drift in warranty data for a large automotive manufacturer and try to accommodate its effects by conditioning the input distributions in a Gaussian Mixture Model (GMM). The proposed approach, CGMM [52], is a successful predictive analytics tool that explicitly considers and models the effects of data drift. Furthermore, the proposed approach also provides an epistemic uncertainty quantification without forcing any time-based trends on the input data. Figure 1-4 shows the architecture of the proposed CGMM.

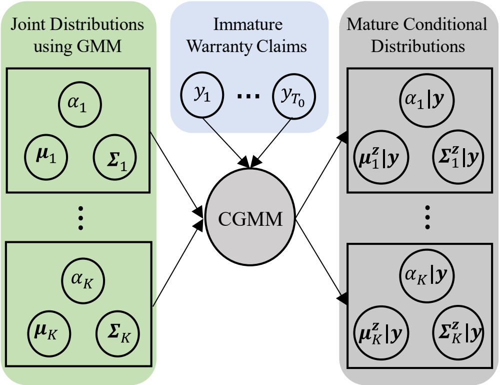

Joint Distributions using GMM
α_1
$\boldsymbol{\mu}_1$
$\boldsymbol{\Sigma}_1$
α_K
$\boldsymbol{\mu}_K$
$\boldsymbol{\Sigma}_K$
Immature Warranty Claims
y_1
y_{T_0}
CGMM
Mature Conditional Distributions
$\alpha_1|\mathbf{y}$
$\boldsymbol{\mu}_1^z|\mathbf{y}$
$\boldsymbol{\Sigma}_1^z|\mathbf{y}$
$\alpha_K|\mathbf{y}$
$\boldsymbol{\mu}_K^z|\mathbf{y}$
$\boldsymbol{\Sigma}_K^z|\mathbf{y}$

Figure 1-4 CGMM: A custom AI architecture for negotiating data drift.

In Chapter 4, we focus on developing a custom, robust, and scalable approach for the virtual verification and validation of AVs. In this work, we propose BOSE, a Bayesian experimentation framework that leverages design of experiments framework [90] to identify “anomalous behavior” also known as “emergent behavior” in AVs. The proposed framework uses Bayesian neural networks to identify the best experiments that help us completely understand the complex interactions in input variables that lead to emergent behavior in an Autonomous Driving System (ADS) of an AV. Figure 1-5 shows the BOSE framework.

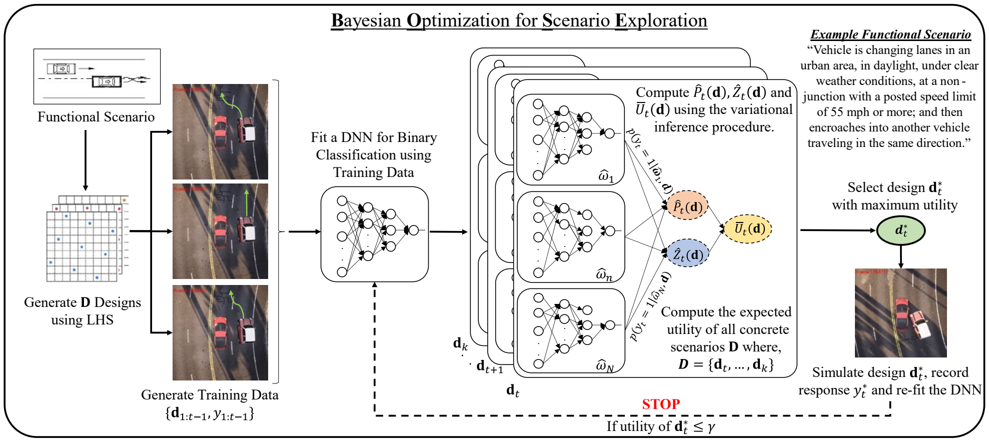
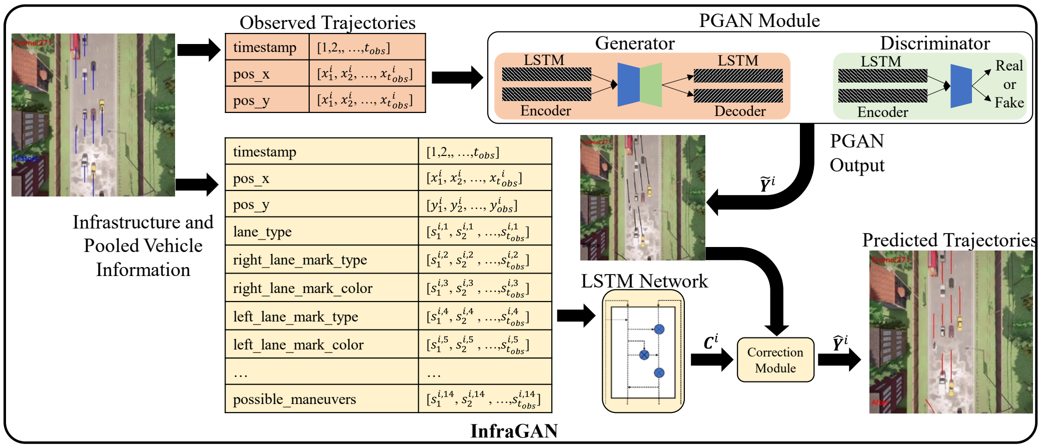

Bayesian Optimization for Scenario Exploration
Example Functional Scenario
“Vehicle is changing lanes in an urban area, in daylight, under clear weather conditions, at a non - junction with a posted speed limit of 55 mph or more; and then encroaches into another vehicle traveling in the same direction.”
Functional Scenario
Generate **D** Designs using LHS
Generate Training Data $\{\mathbf{d}_{1:t-1}, y_{1:t-1}\}$
Fit a DNN for Binary Classification using Training Data
Compute $\hat{P}_t(\mathbf{d}), \hat{Z}_t(\mathbf{d})$ and $\bar{U}_t(\mathbf{d})$ using the variational inference procedure.
$p(y_t = 1|\hat{\omega}_1, \mathbf{d})$
$p(y_t = 1|\hat{\omega}_N, \mathbf{d})$
$\hat{\omega}_1$
$\hat{\omega}_n$
$\hat{\omega}_N$
$\hat{P}_t(\mathbf{d})$
$\hat{Z}_t(\mathbf{d})$
$\bar{U}_t(\mathbf{d})$
Compute the expected utility of all concrete scenarios **D** where, $\boldsymbol{D} = \{\mathbf{d}_t, \ldots, \mathbf{d}_k\}$
$\mathbf{d}_k$
$\mathbf{d}_{t+1}$
$\mathbf{d}_t$
Select design $\mathbf{d}_t^*$ with maximum utility
$\boldsymbol{d}_t^*$
Simulate design $\mathbf{d}_t^*$, record response y_t^* and re-fit the DNN
STOP
If utility of $\mathbf{d}_t^* \leq \gamma$

Figure 1-5 BOSE: A novel data-driven frame that models and predicts black-swan events in an AI-driven systems.

Finally, in Chapter 5, we develop a custom Generative model called InfraGAN that performs collaborative modeling. InfraGAN is the world’s first generative model that leverages infrastructure information for accurate trajectory predictions in AVs. However, the input data to train models such as InfraGAN is often private and not publicly available. Therefore, in Chapter 5 of this work, we also introduce a large-scale, synthetically generated, publicly available autonomous driving dataset with tags for relevant infrastructure information named VTrackIt [89]. Figure 1-6 shows the architecture of the proposed InfraGAN.



Observed Trajectories

timestamp	$[1,2,, \dots, t_{obs}]$
pos_x	$[x_1^i, x_2^i, \dots, x_{t_{obs}}^i]$
pos_y	$[x_1^i, x_2^i, \dots, x_{t_{obs}}^i]$

timestamp	$[1,2,, \dots, t_{obs}]$
pos_x	$[x_1^i, x_2^i, \dots, x_{t_{obs}}^i]$
pos_y	$[y_1^i, y_2^i, \dots, y_{obs}^i]$
lane_type	$[s_1^{i,1}, s_2^{i,1}, \dots, s_{t_{obs}}^{i,1}]$
right_lane_mark_type	$[s_1^{i,2}, s_2^{i,2}, \dots, s_{t_{obs}}^{i,2}]$
right_lane_mark_color	$[s_1^{i,3}, s_2^{i,3}, \dots, s_{t_{obs}}^{i,3}]$
left_lane_mark_type	$[s_1^{i,4}, s_2^{i,4}, \dots, s_{t_{obs}}^{i,4}]$
left_lane_mark_color	$[s_1^{i,5}, s_2^{i,5}, \dots, s_{t_{obs}}^{i,5}]$
…	…
possible_maneuvers	$[s_1^{i,14}, s_2^{i,14}, \dots, s_{t_{obs}}^{i,14}]$

Figure 1-6 InfraGAN: A custom DL model that performs collaborative modeling.

Chapter 2 USAL: Uncorrelated Sparse LSTM Autoencoders for Robust Long-term Estimations

Today, most modern electric and battery-operated devices use Li-ion batteries. Consequently, for their safe and reliable operation, accurate SOC estimations are necessary. Unfortunately, existing methods often fail to identify patterns relevant to long-term SOC estimation due to complex battery cell characteristics such as aging. In this work, we propose the Uncorrelated Sparse Autoencoder with Long Short-Term Memory (USAL). USAL is a novel neural network that addresses the challenging task of long-term SOC estimation given a limited initial history of a cell's charge-discharge behavior. USAL uses a multi-task learning strategy to harness the advantages of sparse autoencoders and LSTM networks by enforcing correlation penalties. The USAL simultaneously learns to (i) generate a latent space of informative SOC encodings from commonly measured cell characteristics, (ii) penalize for high multicollinearity between encodings, and (iii) identify non-trivial long and short temporal correlations between the encodings using LSTM cells. USAL outperforms benchmarked models in our experiments when trained on five initial charge-discharge cycles across multiple battery cells using three publicly available accelerated aging datasets.

2.1 Introduction

Li-ion battery cells have been pivotal in the development of intelligent and EVs, drones, portable electronics, grid storage systems as well as many other systems and applications. Today, these battery cells are used in various electronic devices due to the humongous spectrum of advantages they offer over existing energy storage technologies. Some advantages of Li-ion battery cells include their high specific energy, high cycle life, high Coulombic efficiency (up to 98%), and low self-discharge rates. These are mainly of high benefit to EVs. Their high efficiency and power density make the use of Li-ion battery cells extremely lucrative in modern energy-efficient electric grids [91]. Li-ion battery cells also offer significant performance improvements and other environmental benefits [92]. Thus, Li-ion battery cells are considered to be a sufficient alternative to existing energy storage systems and technologies.

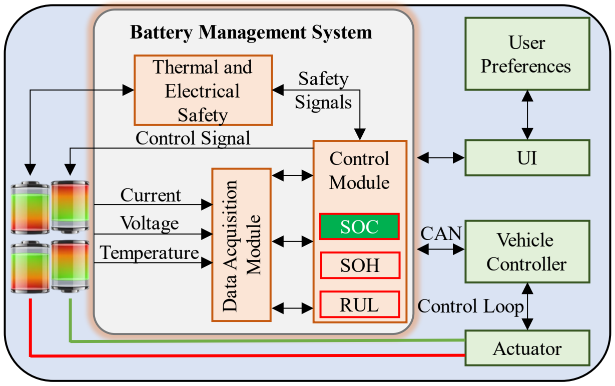


Figure 2-1 High-level representation of a battery management system in an intelligent vehicle.

Efficient management of Li-ion battery cells is necessary to ensure a safe operation, improved performance, optimized power management, and prolonged service life [93]. These operations are commonly performed using a battery management system (BMS). Figure 2-1 shows an overview of a BMS in an EV. For a BMS to work efficiently, it is crucial to measure the battery parameters accurately. While battery measurements such as voltage, current, and temperature can be accurately measured, metrics such as state-of-charge (SOC) and state-of-health (SOH) can only be estimated within the control module using the measured inputs. Based on these estimates, the BMS controls various operational aspects of the battery while considering user preferences. SOC is a metric that is used to describe the ratio between the remaining capacity and the maximum (nominal) capacity of a battery measured in Ampere-hours (Ah) at a specific temperature [19]. Inaccurate estimation of the SOC may lead to over-charge or discharge conditions forced upon the battery by the BMS control module. Such situations may lead to catastrophic failures of battery cells that can potentially harm human life.

A Li-ion battery is a complex electrochemical system that, upon demand, efficiently stores and distributes electrical energy. Thus, it is of high interest to check its operating parameters and metrics to make crucial decisions about its operation. SOC is one such metric that is of high importance. For example, the driving range in an EV is calculated based on the SOC of a battery. When the SOC of a battery nears 15%, the EV displays a 'Low-Charge' warning on the dashboard. This is very similar to a 'Low-Fuel' warning light in a standard gasoline vehicle. However, since Li-ion battery cells have non-linear characteristics that depend on various internal and external conditions, accurately estimating the SOC is a challenging task [94], [95].

Xiong *et al.* [93] categorize SOC estimation methods into four distinct categories, namely, (i) look-up table methods, (ii) integral methods, (iii) model-based methods, and (iv) data-driven methods. Look-up table methods are commonly used to map measured characteristics such as open-circuit voltage (OCV) to SOC. This OCV-SOC relationship is pre-mapped, and the SOC estimations are performed by referencing and locating the closest measured OCV value [96], [97]. However, while look-up table-based methods offer accurate SOC estimations, they cannot perform online estimations needed by EVs due to isolation requirements [94], [98], [99].

Integral methods such as Coulomb counting use integrals of measured cell current over time to calculate the SOC.

$$SOC_t = SOC_{t_0} + \frac{1}{C_N}\int_{t_0}^{t} \eta \text{ x } i \, dt \tag{2-1}$$

In eq. (2-1), SOC_t is the SOC estimated at time t and SOC_{t_0} is the reference SOC at some time t_0, C_N is nominal cell capacity, η is the cell efficiency, and i is the measured cell current in Amps.

Although this method can perform online estimations, it is highly dependent on the reference value SOC_{t_0}. Further, it is also limited by its ability to filter noise in the measured current i. Thus, due to robustness issues caused by their open-loop nature, integral methods are complemented with model-based methods [93].

The estimation models are defined using highly complex and non-linear state equations in model-based methods. SOC estimation methods using Kalman filter-based approaches are a typical example of these methods [100]. Whereas model-based methods are more robust in their SOC estimations, they are often limited by their ability to capture cell aging and degradation information [101]. Although significant research has improved these models' accuracy, latency remains an unaddressed challenge [102]. Thus, researchers have come up with data-driven methods. These approaches are often referred to as 'black-box' methods due to their lack of explainability [93]. Although data-driven methods can be used for robust and online SOC estimations, they often use highly complex Machine Learning (ML) training procedures that require large amounts of training data for accurate SOC estimations [18], [103]–[106]. Moreover, they often fail to account for cell aging and need frequent retraining [107].

Aging is a complex phenomenon experienced by Li-ion battery cells leading to capacity loss over time and use [108], [109]. Aging processes do not have one cause but result from several chemical processes and interactions [110]. Palacin [110] provides a detailed analysis of cell aging and its effects on capacity. Although cell aging is a well-known and researched phenomenon, little work has been done to adopt its effects on long-term SOC estimations. Short-term SOC estimation involves estimating the SOC for a few cycles in the immediate future. On the other hand, long-term SOC estimation involves estimating SOC for cycles near EOL, given some initial charge-discharge cycles. Researchers have recently tried to enhance the data-driven models using Transfer Learning (TL) with promising results for short-term SOC estimations [19], [43].

Although data-driven models show improvement upon other existing models, they often require hours of training on large amounts of battery aging data [19], [43]. Further, these models need frequent retraining to update their hyper-parameters. This helps them capture the latest aging-related temporal changes within the battery cell. Such models are also referred to as 'self-adaptive aging models.' A detailed review of such methods can be found in the work by Lucu *et al.* [111]. In similar work, researchers have also studied combining autoencoders and Long Short-Term Memory (LSTM) networks for short-term SOC estimations [112]. Although this study explores short-term SOC estimations in detail, it follows a two-step training procedure and fails to discuss the effect of correlated encodings on long-term SOC estimations.

This work proposes the Uncorrelated Sparse Autoencoder with Long Short-Term Memory (USAL). USAL is a unique combination of data-driven sparse autoencoders and an LSTM network that performs accurate long-term SOC estimations by penalizing high correlations between encodings. Correlation penalties within USAL force it to transform commonly measured inputs (e.g., voltage) into a space of informative encodings that represent unique statistical features. The LSTM network within USAL then identifies temporal trends in this encoded data, supporting online and accurate long-term SOC estimations. In our experiments, USAL achieves accurate long-term SOC estimations when trained only on the initial five charge-discharge cycles under diverse operating conditions. USAL also achieves state-of-the-art when evaluated using dynamic loading profiles using a publicly available dataset. To the best of our knowledge, this is one of the first works that introduces the problem of long-term SOC estimations. This is also one of the first works that analyzes the effects of multicollinearity between encodings generated by an autoencoder for SOC estimations in Li-ion battery cells. In summary, some significant contributions of this work include:

1. **Long-Term SOC estimations**: Exploration of the challenges in long-term SOC estimations and proposition of a novel data-driven solution.

2. **Multi-task learning**: Proposition of a multi-task learning strategy to simultaneously train a sparse autoencoder and an LSTM network given limited training data.

3. **Informative encodings**: Learning a diverse and informative set of encodings by penalizing for high multicollinearity between learned encodings.

4. **Age-mapping**: Identifying and mapping aging-related trends in Li-ion battery cells for accurate and reliable SOC estimations.

2.2 Uncorrelated Sparse Autoencoder with Long-short-term Memory (USAL)

USAL is a state-of-art deep neural network that simultaneously learns to reconstruct inputs and perform long-term SOC estimations. The following discussion explores each part within the USAL in more detail.

2.2.1 Sparse Autoencoders

Autoencoders are specialized neural networks that are designed to transform inputs into a latent space of learned representations (encodings) [113]–[115]. Formally, encodings are defined as the 'learned representation of data that can reconstruct inputs even in the presence of noise' [113], [116], [117].

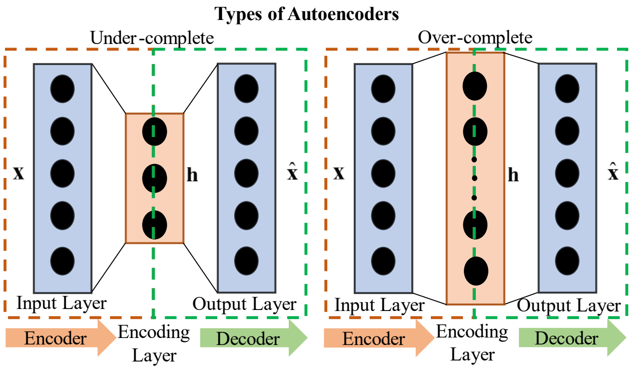


Figure 2-2 Under-complete and over-complete autoencoders.

Figure 2-1 shows two autoencoders, each with one hidden layer. In this figure, an m-dimensional input vector $\mathbf{x} \in \mathbb{R}^m = X$ is mapped to a p-dimensional encoding $\mathbf{h} \in \mathbb{R}^p = F$ in the encoder ($\phi: X \rightarrow F$). In the decoder, the p-dimensional encodings are then transformed to an m-dimensional reconstruction $\hat{\mathbf{x}} \in \mathbb{R}^m = \hat{X}$ ($\psi: F \rightarrow X$). USAL thus trains its autoencoder by trying to find mapping functions ϕ, and ψ that minimize the reconstruction error $\mathcal{L}_{recon}$ such that,

$$\phi, \psi = argmin_{\phi,\psi}(\mathcal{L}_{recon}) \tag{2-2}$$

where,

$$\mathcal{L}_{recon} = \sqrt{\sum_{c}^{C} \sum_{t}^{T} (\boldsymbol{x}^{c,t} - \hat{\boldsymbol{x}}^{c,t})^2} \tag{2-3}$$

In eq. (2-3), let $\mathbf{x}^{c,t} = [x_1^{c,t}, x_2^{c,t}, \ldots, x_m^{c,t}]$ be a vector representing a set of m inputs (e.g., voltage, current, temperature) given to USAL at some time t in a charge-discharge cycle c. Let $\hat{\mathbf{x}}^{c,t} = [\hat{x}_1^{c,t}, \hat{x}_2^{c,t}, \ldots, \hat{x}_m^{c,t}]$ be a vector representing reconstructed inputs generated by the autoencoder at its output at the same time t in a charge-discharge cycle c. Thus, eq. (2-3) forms the first part of USAL's loss function, whose objective is to train an efficient autoencoder within USAL.

For under-complete autoencoders $\mathbf{h} \epsilon \mathbb{R}^p$ where $p < m$ [113]. Typically, for the task of SOC estimations, current, voltage, and temperature are the only measured inputs. Thus, by choosing $p < m$, we may introduce a strong bias within USAL, limiting its SOC estimation capabilities. To overcome this problem, in this work, we use over-complete autoencoders where, $p \geq m$ [113], [114]. As per recommendations in literature [113], [116], [118], [119], to avoid learning identity functions, we enforce sparsity regularization over the encoding layer in the over-complete autoencoder within USAL. Identity function mapping can be avoided by limiting neuron activations in the encoding layer using L_1 regularization [68], [120]. Such sparsity-enforced over-complete autoencoders are known as 'Sparse Autoencoders' [119], [121]. In USAL, sparsity constraints are enforced using $\mathcal{L}_{spars}$ such that,

$$\mathcal{L}_{spars} = \sum_{c}^{C} \sum_{t}^{T} |\boldsymbol{h}^{c,t}| \tag{2-4}$$

Here, $\mathbf{h}^{c,t} = [h_1^{c,t}, h_2^{c,t}, \dots, h_p^{c,t}]$ represents the output of the encoding layer in USAL where, $h_p^{c,t}$ is the output of encoding p given some input $\mathbf{x}^{c,t}$.

2.2.2 Correlation Analysis

While L_1 regularization performs sparsity regularization, it offers little help with multicollinearity [122]. Multicollinearity is a phenomenon in which two or more variables are highly correlated [123]. Usage of such highly correlated encodings may severely limit the ability to extract aging information from the encodings [42]. In USAL, to ensure that the encodings learn a valuable representation of the input, we introduce the correlation loss $\mathcal{L}_{corrl}$, such that,

$$\mathcal{L}_{corrl} = \sum_{i} \sum_{j} |k_{i,j}| \tag{2-5}$$

where,

$$k_{i,j} = \frac{1}{C}\sum_{c}^{C} \frac{\sum_t^T (h_i^{c,t} - \bar{h}_i^c)(h_j^{c,t} - \bar{h}_j^c)}{\sqrt{\sum_t^T \left(h_i^{c,t} - \bar{h}_i^c\right)^2 \sum_t^T \left(h_j^{c,t} - \bar{h}_j^c\right)^2}} \tag{2-6}$$

In eq. (2-6), $k_{i,j}$ is the correlation between some encoding i and encoding j, and $\bar{h}_i^c$ is the mean output of some encoding i in cycle c. Note that $k_{i,j}$ is always equal to 1 if $i = j$. The correlation loss thus forms a crucial part of USAL and helps reduce multicollinearity within the encodings while simultaneously learning to transform inputs into outputs with minimal distortion.

2.2.3 Prediction Network

A prediction network with an LSTM layer forms the final part of USAL. LSTM networks are a variant of Recurrent Neural Networks (RNNs) that can learn temporal trends in the input data by developing and retaining memories from previous time steps [43], [112], [124]–[128]. USAL thus learns to perform accurate SOC estimations by minimizing $\mathcal{L}_{soc}$, such that,

$$\mathcal{L}_{soc} = \sqrt{\sum_{c}^{C}\sum_{t}^{T} \frac{\left(\widehat{SOC}_t^c - SOC_t^c\right)^2}{CT}} \tag{2-7}$$

In eq. (2-7), $\widehat{SOC}_t^c$is the SOC estimated by USAL and SOC_t^c is the true SOC given some input $\mathbf{x}^{c,t}$ in some cycle c at a time t.

In summary, given input $\mathbf{x}^{c,t}$ USAL simultaneously learns to recreate $\hat{\mathbf{x}}^{c,t}$ using a sparse autoencoder and perform accurate SOC estimations ($\widehat{SOC}_t^c$) using a prediction network while penalizing for high multicollinearity between the learned encodings. Equation (2-8) effectively summarizes the loss function used for training USAL where λ_r, λ_s and λ_c are the reconstruction, sparsity, and correlation regularization hyperparameters. Figure 2-3 helps visualize all components and their functions within USAL.

$$\mathcal{L}_{USAL} = \lambda_r \mathcal{L}_{recon} + \lambda_s \mathcal{L}_{spars} + \lambda_c \mathcal{L}_{corrl} + \mathcal{L}_{soc} \tag{2-8}$$

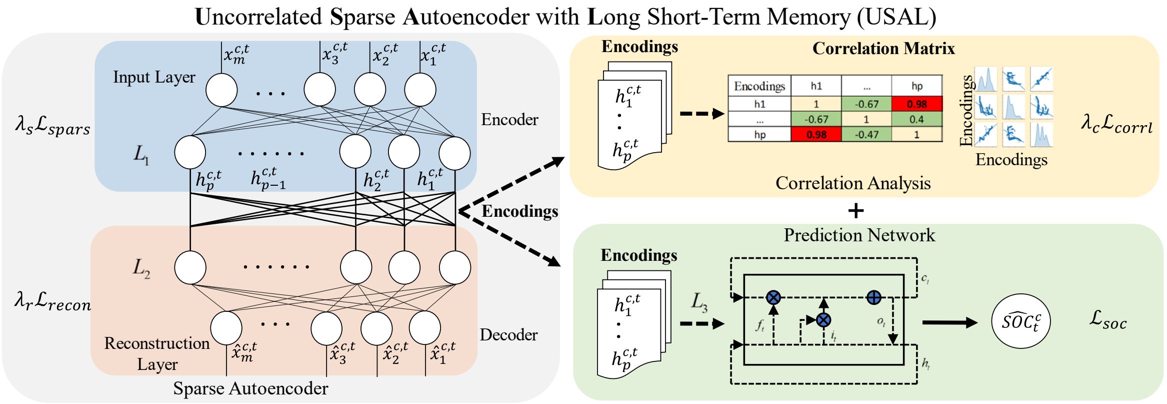


Figure 2-3 Uncorrelated Sparse Autoencoder with Long Short-Term Memory (USAL) for long-term SOC estimations.

2.3 Case Study

2.3.1 Datasets

Table 2-1 Summary of Accelerated Aging Datasets

Dataset	Purpose	Cell Type (Cathode/Anode)	Nominal Capacity (Ah)	Nominal Voltage (V)
Dataset-1	Accelerated Aging	18650 LiB (LiCoO2/LiC)	2.0	4.2
Dataset-2		APR18650M1A (LFP/Graphite)	1.1	3.3
Dataset-3		Pouch Cell (LiCoO2/Graphite)	3.36	3.82

This work uses three publicly available accelerated aging datasets that have a wide range of applications [99], [129]–[131]. Every considered dataset follows a unique accelerated aging process and uses a unique battery configuration. The three considered datasets are heavily studied in the literature. Details of each dataset are summarized in Table 2-1.

This work uses three publicly available accelerated aging datasets that have a wide range of applications [99], [129]–[131]. Every considered dataset follows a unique accelerated aging

process and uses a unique battery configuration. The three considered datasets are heavily studied in literature and acknowledged by industry experts for studying aging behavior in Li-ion battery cells. Details of each dataset and the battery cells used are summarized in Table 2-1. The datasets used in this study represent a comprehensive coverage of experimentation in accelerated aging for Li-ion cells. In our experiments, we extensively use these datasets to analyze the performance of USAL for long-term SOC estimations.

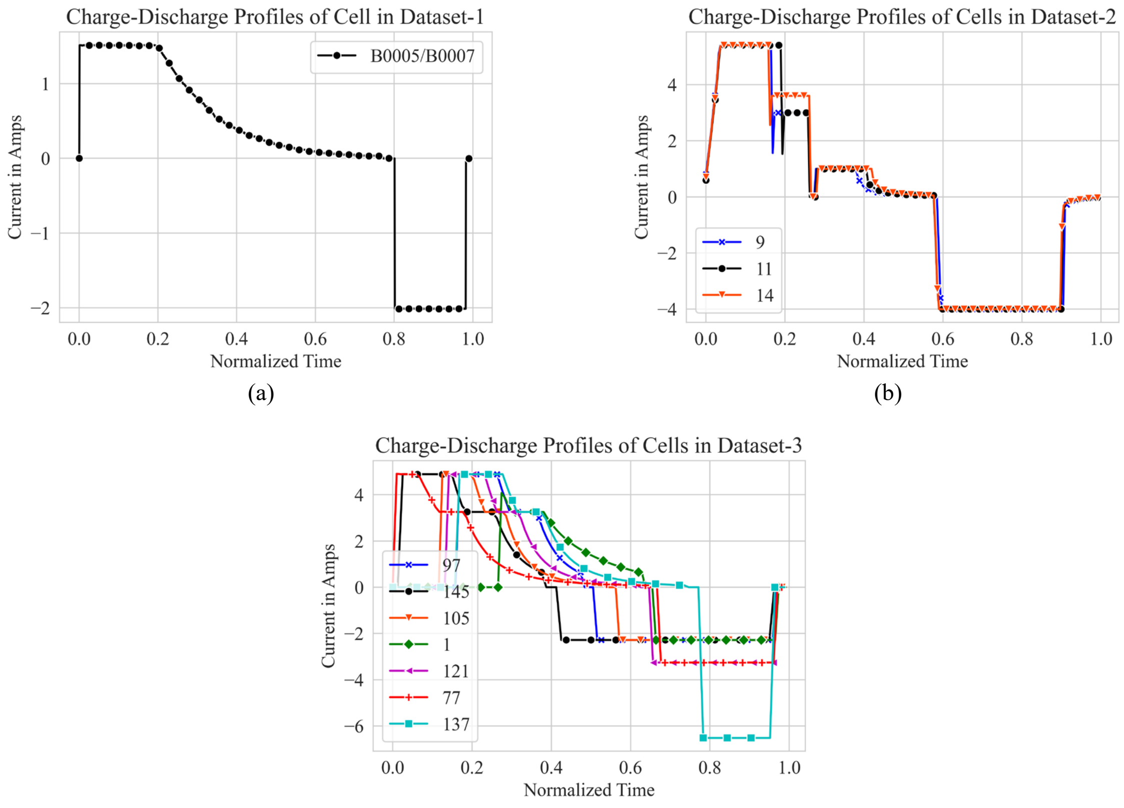
Dataset-1- The first dataset is provided by the NASA Ames Laboratory [129]. It collects batches of four Li-ion battery cells that are subject to accelerated aging using three different charge-discharge profiles. The recorded data includes measurements such as the current (in Amps), the voltage (in Volts), and the temperature (in ℃) of each cell. For this study, from this dataset, we use cell 'B0005' for validation and hyper-parameter tuning and 'B0007' for testing. For all Li-ion battery cells in this dataset, charging is carried out in Constant Current (CC) mode at 1.5A until the cell voltage reaches 4.2V. It is then continued in a Constant Voltage (CV) mode until the charge current drops to 20mA. The battery cells are then discharged at a Constant Current (CC) level of 2A until the cell voltage drops to 2.7V. Finally, repeated charging and discharging is implemented, which results in the accelerated aging of the battery cells. Experiments are stopped when the battery cells reach EOL- 30% fade in rated capacity. This dataset is referred to as 'Dataset-1.'

Dataset-2- The second dataset provided by Severson *et al.* [130] is a congregation of 124 LFP/Graphite battery cells (A123 Systems, model APR18650M1A). The recorded data includes measurements such as current (in Amps), voltage (in Volts), and temperature (in ℃) for each battery cell. This dataset aims to explore the effects of fast charging on cell aging. The battery cells are charged from 0% to 80% SOC with one of 72 different one-step and two-step charging policies.

They are then charged from 80% to 100% SOC with a uniform 1C CC-CV charging step to 3.6V and a current cut-off of C/50 where 1C is 1.1A. During the discharge process, the battery cells are discharged under the CC-CV conditions with discharge current set to 4C and 2.0V with a current cut-off of C/50. Repeated charging and discharging results in the accelerated aging of the battery cells. Experiments are stopped when the battery cells reach EOL- 20% fade in the rated capacity. This dataset is referred to as 'Dataset-2'.

Dataset-3- The third dataset provided by Diao *et al.* [131] is a collection of 192 Li-ion battery cells. In their experiments, Diao *et al.* [131] perform accelerated aging of the battery cells using three stress factors and a full factorial design of experiments framework. The recorded data includes measurements such as the current (in Amps) and each battery cell's voltage (in Volts). This dataset aims to analyze the effect of these stress factors on capacity fade by continuously cycling cells. The three stress factors used are- (i) ambient temperature (10 °C, 25 °C, 45 °C, 60 °C), (ii) charge cut-off C-rate (C/5, C/40), and (iii) discharge current rate (0.7C, 1C, 2C). In their experiments, the battery cells are charged at 1.5C to 4.2V. They are then held at 4.2V until the current decreases to 1C (3.36A). Charging is then continued up to 4.4V at 1C. The battery cells are held at 4.4V until the current drops to the given charge cut-off rate in stress factor (ii). The battery cells are rested for five minutes and finally discharged using the discharge rate given by the stress factor (iii). Repeated charging and discharging is implemented, which results in the accelerated aging of the battery cells. Experiments are stopped when the battery cells reach EOL- 50% fade in the rated capacity. Note that the tests are performed at ambient temperature values given by stress factor (i). This dataset is referred to as 'Dataset-3'.

The datasets used in this study represent a comprehensive coverage of experimentation in accelerated aging for Li-ion cells. Figure 2-4 shows the charge-discharge profiles of all test Li-ion battery cells used in this study. More details on the specific test conditions of every considered Li-ion battery cell are provided and discussed in the following sections.



(a)

(b)

(c)

Figure 2-4 Charge and discharge profiles of all cells used in this study in – (a) Dataset-1, (b) Dataset-2, and (c) Dataset-3.

2.3.2 Training and Tuning USAL

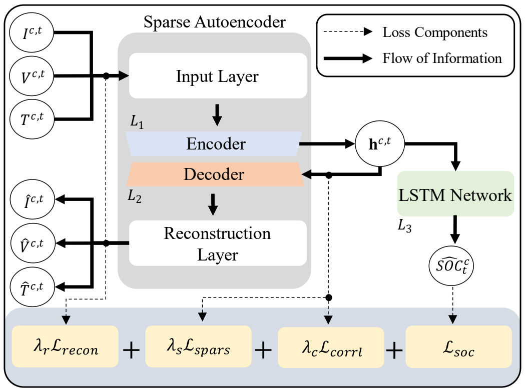


Figure 2-5 Flow of information within USAL and its loss components using commonly measured inputs in a typical Li-ion battery cell.

Table 2-2 Summary of Train-Test Data

Dataset	Train Cycles	Test Cycles	Inputs to USAL
Dataset-1	Initial 5 Cycles	Final 20 Cycles	$I^{c,t}, V^{c,t}, T^{c,t}$
Dataset-2			$I^{c,t}, V^{c,t}, T^{c,t}, I_{avg}^{c,t},\ T_{avg}^{c,t}$
Dataset-3			$I^{c,t}, V^{c,t}$

For long-term SOC estimations, USAL is trained using the initial five and tested on the final twenty charge-discharge cycles for every considered Li-ion battery cell, unless specified otherwise. Table 2-2 summarizes the inputs to USAL along with details of training and testing data for all three accelerated aging datasets used in this work. For Li-ion battery cells in 'Dataset-2', we introduce the moving average of the last five time steps for the measured current and voltage as two additional inputs based on the recommendations of Chemali *et al.* [19]. Figure 2-5 can be used to visualize the flow of information within USAL along with its loss components for a typical Li-ion battery cell.

USAL is a highly customizable deep neural network that can be modified according to the task on hand. In this section, this task is long-term SOC estimations. We use data from cell 'Cell-B0005' in 'Dataset-1' for tuning USAL's architecture and hyper-parameters. Although several architectures for USAL were tried in this case study, we only provide details for the best-performing architecture. All tested models were trained using the same initializations for a fair evaluation. The sparse autoencoder within USAL is an LSTM encoder-decoder network that consists of two key LSTM layers L_1 and L_2. In our experiments, although several values of p were tried, p, when set to 10, gave the best results unless specified otherwise. Thus, both L_1 and L_2 are defined as LSTM layers with 10 nodes each. Further, the prediction network within USAL consists of a single LSTM layer, L_3 with 10 nodes and uses the encodings generated by the encoder within USAL as inputs. The lookback steps for all LSTM layers within USAL are set to 50. While the outputs of all layers in USAL are transformed using a 'tangent hyperbolic' function, the outputs of the reconstruction layer are transformed using a 'linear' activation function.

As in any deep neural network, hyper-parameter tuning is an essential aspect of USAL. As such, in our experiments, we run an exhaustive grid search and train-test USAL by varying λ_r, λ_s and λ_c from 0-1 in eqs. (2-2) - (2-8). Figure 2-6 can be used to visualize the performance of USAL for long-term SOC estimations when using validation 'Cell-B0005'. As can be seen in this figure, USAL shows its best performance when λ_r, λ_s and λ_c are set to 0.06, 1e-4, and 1e-3, respectively. As can be observed in Figure 2-6, USAL shows poor performance when either λ_r, λ_s and λ_c are set to zero. This provides evidence that all losses defined in Section 2.2 play a crucial role in training USAL. Thus, we use the above-defined architecture, hyper-parameter values, and training strategy in all our experiments when performing SOC estimations using USAL. Note that we train USAL for 300 epochs in all our experiments using the 'Adam' optimizer on an Intel Xeon 5218R CPU and a single Nvidia RTX 2080 Ti GPU.

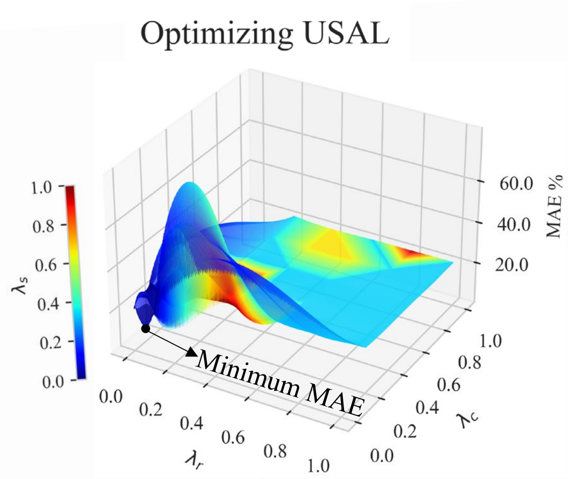


Figure 2-6 Long-term SOC estimation validation performance using different values of λ_r, λ_s and λ_c.

2.3.3 Comparison Across Battery Configurations

Proposition of models that estimate SOC only for specific battery cell configuration offers limited value to the industry. While one manufacturer may prefer a LiCoO2 cathode, the others

Table 2-3 Summary of results under varying cell configurations

Ambient Temperature (°C)	Cathode/Anode	Cell Id	MAE (%)	MAX Error (%)	RMSE (%)
Room Temp.	LiCoO2/LiC	B0007	3.60	68.99	6.63
25	LFP/Graphite	11	3.07	16.61	4.00
25	LiCoO2/Graphite	77	1.88	7.00	2.44

may prefer an LFP cathode. To this end, we train and test USAL on three Li-ion battery cells with varying configurations. Specifically, we use 'Cell-B0007' from 'Dataset-1', 'Cell-11' from 'Dataset-2', and 'Cell-77' from 'Dataset-3'. We compare the performance of USAL with three other data-driven models- a Deep Neural Network (DNN), a standard LSTM network, and another autoencoder-based SOC estimation framework, the Autoencod-LSTM [112]. Although several architectures for DNN and LSTM networks were tried, we provide details only for the best-performing architectures. The DNN modeled herein consists of three hidden layers with 16, 8, and 4 neurons each. The standard LSTM network used for comparisons consists of a single LSTM layer with 20 hidden nodes with 50 lookback time steps. The DNN and the standard LSTM network are trained using the mean squared error loss function for 300 epochs. We use the same architecture and strategy to train the Autoencod-LSTM as given by Fasahat *et al.* [112]. For a fair comparison, all benchmarked models use the same data used to train and test USAL.

Table 2-4 Cells and their Operating Conditions

Dataset	Cell Id	Charge	Discharge	Ambient Temperature
Dataset-2	9	5.4C(40%)-3C	4 C	30°C
	11	5.4C(50%)-3C		
	14	5.4C(60%)-3.6C		
Dataset-3	105	C/40	0.7 C	45°C
	121		1 C	
	137		2 C	
	1	C/5	0.7 C	10°C
	97			45°C
	145			60°C
	77	C/40	1 C	25°C

Figure 2-7(a) compares the performance of USAL with the three data-driven models for long-term SOC estimations for battery cells given in Table 2-5. Long-term SOC estimations near

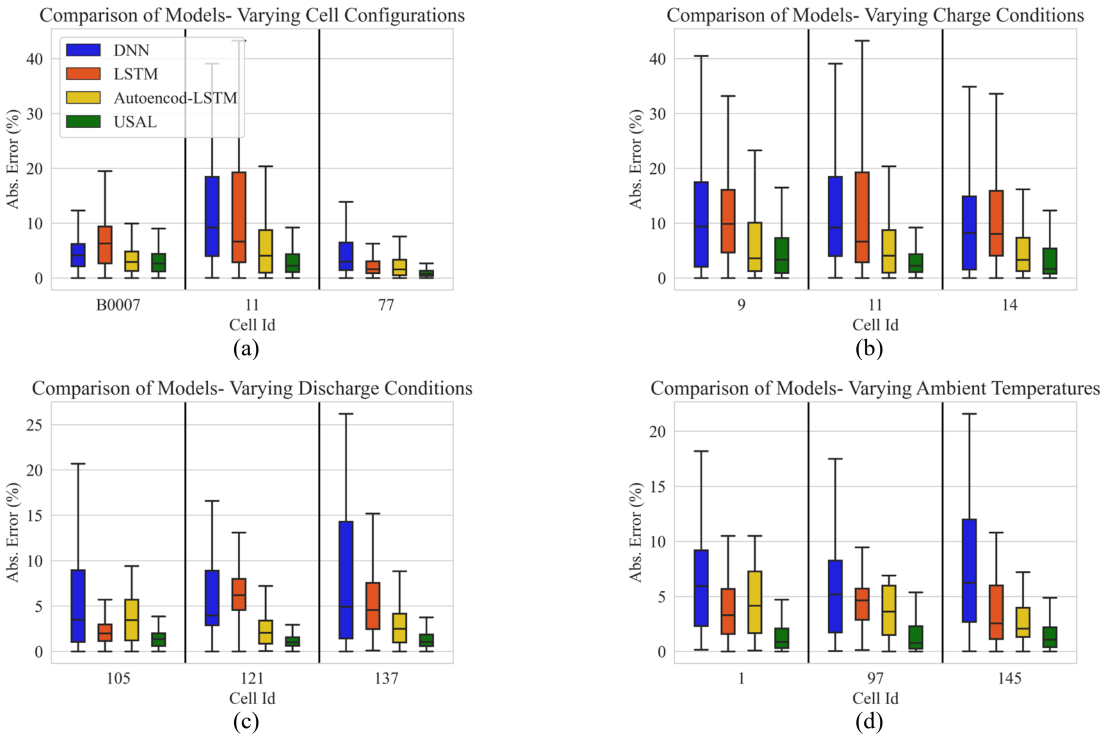
Table 2-5 Summary of Results Under Varying Cell Configurations

Ambient Temperature (°C)	**Cathode/Anode**	**Cell Id**	**MAE (%)**	**MAX Abs. Error (%)**	**RMSE (%)**
Room Temp.	LiCoO2/LiC	B0007	3.63	69.10	6.63
25	LFP/Graphite	11	3.08	16.12	4.06
25	LiCoO2/Graphite	77	0.88	3.80	1.10

the EOL are most challenging, given only a few charge-discharge cycles. Nevertheless, as shown in Figure 2-7(a), USAL comfortably exceeds the other data-driven models' performance. The poor performance of DNNs can be attributed to their lack of ability to capture temporal trends in the data. While the standard LSTM network and Autoencod-LSTM improve on this, they cannot learn the degradation and aging information from just five training cycles. Furthermore, due to a two-step training procedure used in the Autoencod-LSTM, there is no guarantee that the generated encodings will be tailored specifically for long-term SOC estimations. However, USAL applies a multi-task training procedure that penalizes simultaneously for (i) poor SOC estimations, (ii) poor reconstructions, and (iii) highly correlated encodings for accurate long-term SOC estimations. As a result, USAL outperforms other benchmarked methods. For example, while the Autoencod-LSTM achieves an MAE of 2.1%, the USAL achieves a significantly lower MAE of 0.88% for 'Cell-77'. Figure 2-8(a) can be used to visualize this difference in the performance for 'Cell-11.' Using this figure, we can also observe that estimations produced by USAL are less noisy than the other data-driven models. Hence, we can conclude that battery cell configurations have no adverse effect on USAL's ability to perform long-term SOC estimations. Table 2-5 summarizes the performance of USAL using varying cell configurations.

2.3.4 Comparisons Across Varying Charge, Discharge, and Temperature Conditions

Devices such as autonomous vehicles may frequently charge and discharge their battery cells under varying operational conditions. It is well-known that Li-ion battery cells age differently under varying charging, discharging, and ambient temperature conditions [110], [131]. Therefore, in this section, we perform long-term SOC estimations on battery cells aged under various operating conditions. Cells and their respective operational conditions are given in Figure 2-7.



(a)

(b)

(c)

(d)

Figure 2-7 Comparison of USAL with benchmarked models across varying: (a) Cell configurations, (b) Charge conditions, (c) Discharge conditions, and (d) Ambient temperatures across twenty test cycles near EOL.

Figure 2-7(b) and Figure 2-7(c) compare the performance of USAL with the other data-driven models developed in Section 2.3.3 for long-term SOC estimations. From these figures, it can be observed that while the other data-driven models are sensitive to changes in the charge-discharge patterns, USAL and its estimations are robust to any such changes. Extreme charge and discharge conditions age Li-ion battery cells quite quickly, leading to a steep drop in their capacity with time. As a result, it is hard to estimate SOC near EOL. This can be observed in the long-term SOC estimations for ‘Cell-14’ and Cell-137, which are showcased in Figure 2-8(b) and Figure 2-8(c), respectively. While other data-driven methods are unlikely to perform well in such situations, the encodings produced by USAL help overcome this drawback. This can be observed in estimations for ‘Cell-14’ in Figure 2-8(b). Thus, using results from Figure 2-7 and Figure 2-8, we can observe that USAL demonstrates sufficient ability to capture the Li-ion battery cell’s aging trends for long-term SOC estimations under varying charging and discharging conditions.

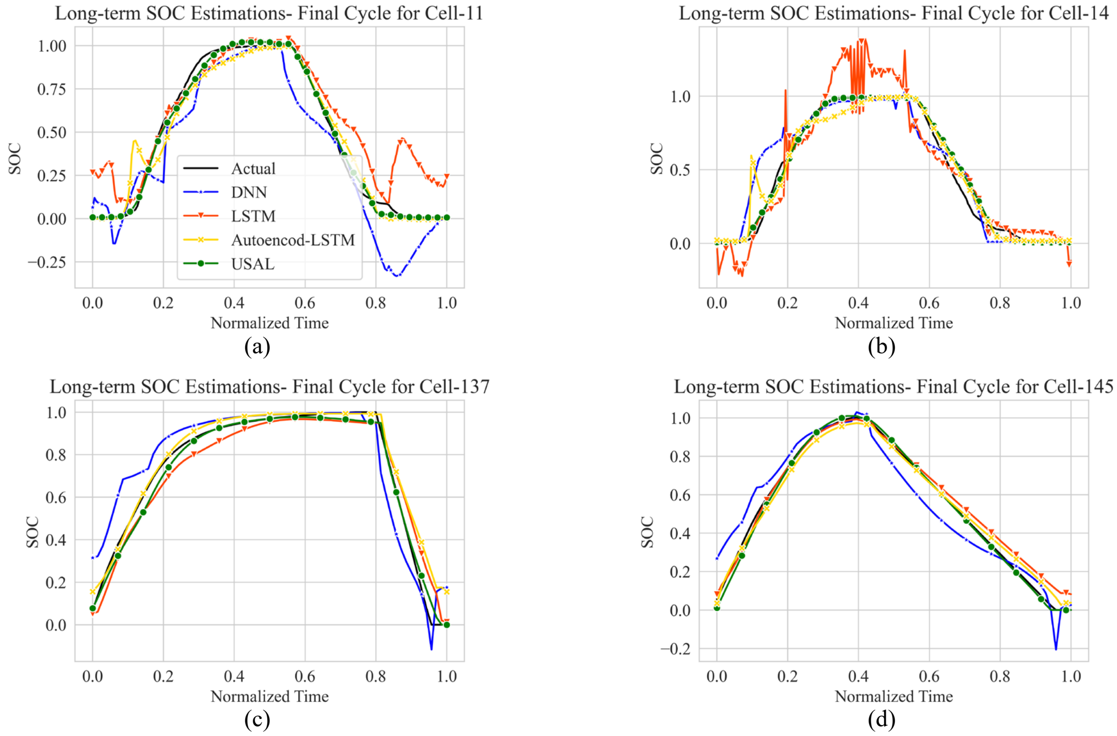


Figure 2-8 Long-term SOC estimations for final cycle near EOL for (a) Cell-11, (b) Cell-14, (c) Cell-137 and (d) Cell-145.

Li-ion battery cells are known to age quite significantly under higher ambient temperatures [108], [130], [131]. Therefore, it is of our high interest to validate the long-term SOC estimation performance of USAL aged under the same charge-discharge profiles but at different ambient temperatures. In our experiments across three Li-ion battery cells, a familiar pattern emerges, and USAL outperforms the other data-driven models for long-term SOC estimations under varying ambient temperatures. Out of three tested battery cells, ‘Cell-145’ ages quickly due to a higher ambient temperature (60 ℃). Thus, as shown in Figure 2-7(d) and Figure 2-8(d), it is much harder for the benchmarked data-driven models to perform accurate long-term SOC estimations. On the other hand, USAL shows overwhelming accuracy in its estimations, achieving a maximum absolute error of 5.78% in its SOC estimations over the twenty test cycles for ‘Cell-145’. Table 2-6 summarizes the performance of USAL across varying charging, discharging, and ambient temperature conditions.

Table 2-6 Summary of Results Under Varying Conditions

Dataset	Cell Id	MAE (%)	MAX Abs. Error (%)	RMSE (%)
Varying Charge Conditions				
Dataset-2	9	4.55	30.91	6.22
	11	3.08	16.12	4.06
	14	3.51	17.49	5.12
Varying Discharge Conditions				
Dataset-3	105	1.43	5.29	1.74
	121	1.29	4.48	1.66
	137	1.67	16.71	2.54
Varying Ambient Temperatures				
Dataset-3	1	1.38	4.71	1.90
	97	1.51	5.93	1.22
	145	1.57	5.78	2.24

2.3.5 Short-term SOC Estimations

The SOC estimation module is responsible for long-term and short-term SOC estimations within the BMS. Therefore, a SOC estimation model that performs excellent long-term SOC estimations while suffering from poor performance in short-term SOC estimations is of little practical value. Thus, in this section, we perform short-term SOC estimations using ‘Cell-137’ in ‘Dataset-3’. Here, all models are tested on the 6th charge-discharge cycle. Figure 2-9 can be used to visualize the performance of all models for short-term SOC estimations. In Figure 2-9(a), we can observe that the standard LSTM shows acceptable performance while estimating the 6th cycle in ‘Cell-137’ with an MAE of 1.05%. The Autoencod-LSTM improves over this and achieves an MAE of 0.79%. The DNN cannot perform such accurate estimations due to its lack of ability to capture time-based trends. As seen in Figure 2-9(a), USAL improves upon the performance of all benchmarked models achieving an MAE of 0.65% for short-term SOC estimations. The higher dimensional encodings and USAL’s ability to negotiate multicollinearity within these encodings result in noise-free and accurate short-term estimations. Figure 2-9(b) showcases the estimation errors for all models for 'Cell-137.' As the disconnect between the training and test cycles increases, the performance of the benchmarked data-driven models worsens. USAL, on the other hand, is much more accurate and reliable. For example, while the Autencod-LSTM achieves an MAE of 6.1% on the last cycle near EOL for 'Cell-137,' USAL estimates this cycle much better, achieving an MAE of 2.5%. The SOC estimations near EOL and associated estimation errors for 'Cell-137' can be seen in Figure 2-9(c) and Figure 2-9(d), respectively.

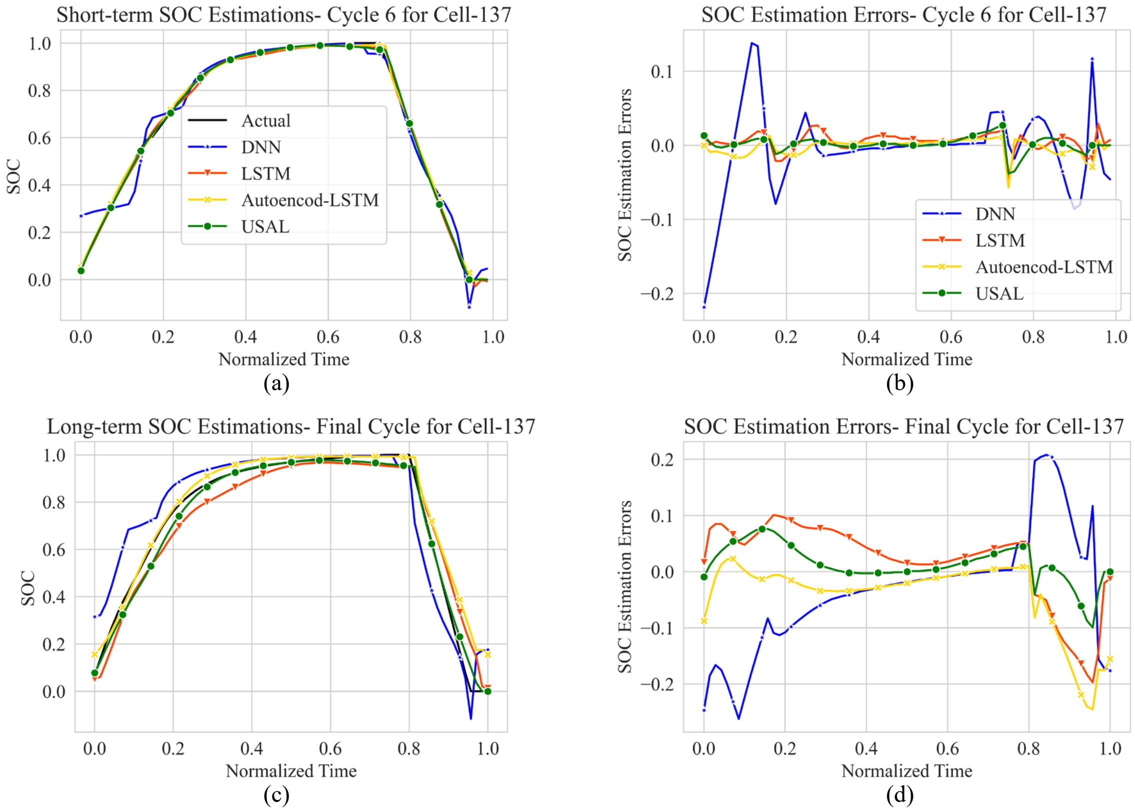


Figure 2-9 Comparisons for short-term and long-term SOC estimations in 'Cell-137' for USAL and other benchmarked models.

2.3.6 Effects of Correlations on Long-term SOC Estimations

In our experiments, although USAL is shown to exceed performance over other benchmarked models, it is of our high interest to validate that this performance gain is due to the reduced multicollinearity between the learned encodings. Consequently, we choose our most challenging cells – ‘Cell-14’, ‘Cell-137’ and ‘Cell-145’ given in Table 2-7 and compare the performance of USAL with and without the correlation loss. We specifically choose these three cells since they showcase the most rapid drop in capacity due to extreme test conditions. The models are then trained using the same architecture and training strategy as given above, and correlation loss is disabled by setting λ_c to zero in eq. (2-8). Note that we use the same initializations when training with and without correlation loss for a fair comparison.

As can be observed in Table 2-7, without the correlation loss, the performance of USAL worsens across all tested battery cells. As the correlation between the encodings reduces, the sparse autoencoder within USAL learns better representations of input data. Such unique data representations can then be leveraged to simultaneously reconstruct the inputs while identifying long-term aging effects for accurate SOC estimations. Learning such informative, higher-dimensional encodings is sensitive to the correlation between encodings. While the correlation loss increases the diversity between the learned encodings, the sparsity, reconstruction, and prediction losses force the network to learn significant aging trends when presented with limited training data.

Table 2-7 Effectiveness of Correlation Loss

Cell Id	MAE (%)	MAX Abs. Error (%)	RMSE (%)	Mean Abs. Correlation Between Encodings
Without Correlation Loss				
14	3.75	19.39	5.43	0.65
137	7.70	19.54	8.17	0.52
145	4.13	12.98	5.17	0.59
With Correlation Loss				
14	3.51	17.49	5.12	0.32
137	1.67	16.71	2.54	0.34
145	1.57	5.78	2.24	0.34

2.3.7 Robustness Evaluation

Typical, off-the-shelf measurement devices often exhibit noise levels ranging from 2% to 4% in their measurements [19]. Thus, any SOC estimation method must be robust against such noise in the measured inputs. As a result, we evaluate the robustness of USAL using the same approach as Chemali *et al.* [19]. In particular, we inject random Gaussian noise with zero mean and a standard deviation of 4% into the measured inputs. When presented with such noisy inputs, as seen in Table 2-8, USAL achieves its worst performance with an MAE of 3.88% for ‘Cell-14’. On the other hand, the Autoencod-LSTM achieves an MAE of 5.4% when no noise is injected in measured inputs for ‘Cell-14.’ A similar pattern is also observed for ‘Cell-137’ and ‘Cell-145’, where the MAE achieved by USAL with injected noise (3.87% and 2.55%) are lower than MAE achieved by Autoencod-LSTM with no injected noise (4.37% and 2.75%). We can thus conclude that USAL is indeed a robust method that performs accurate long-term SOC estimations even under the presence of noisy inputs.

Table 2-8 Robustness of USAL

Cell Id	**MAE (%)**	**MAE (%) with noise**	**RMSE (%)**	**RMSE (%) with noise**
14	3.51	3.88	5.12	5.50
137	1.67	3.87	2.54	4.93
145	1.57	2.55	2.24	3.40

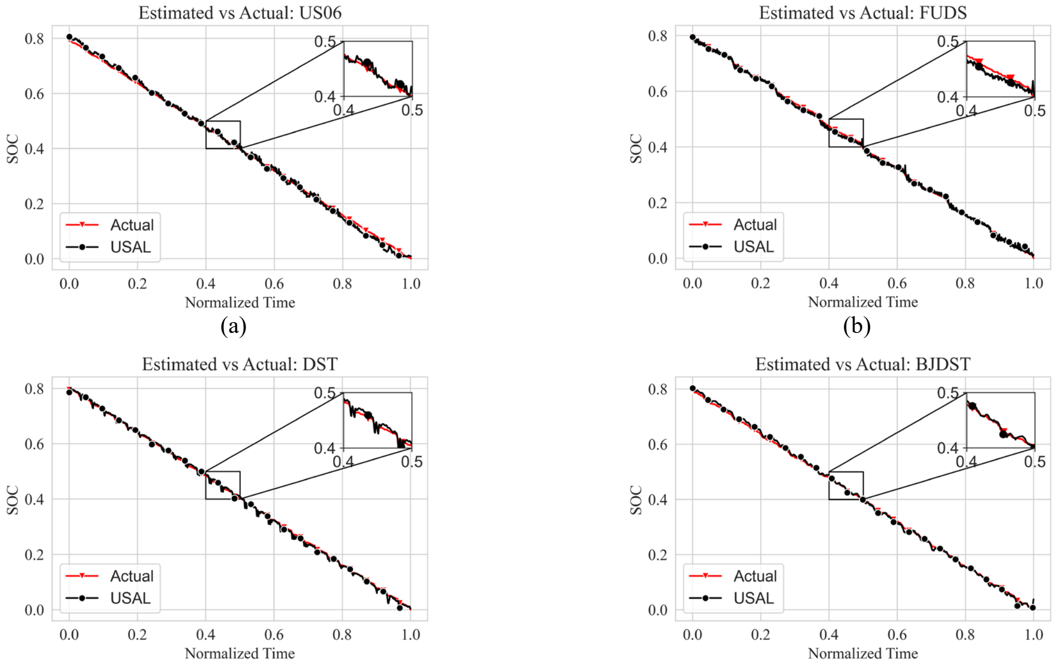
2.3.8 SOC Estimations for Dynamic Loading Profiles

State-of-art SOC estimation models are often benchmarked on dynamic loading profiles like Dynamic Stress Test (DST), Federal Urban Driving Schedule (FUDS), US06 Highway Driving Schedule (US06), and Beijing Dynamic Stress Test (BJDST). This helps prove the model’s estimation capabilities in real-life loading conditions. However, high input variability combined with random charging and discharging patterns make SOC estimation challenging for dynamic loading profiles. Although this work aims to develop a long-term SOC estimation model, evaluating its performance on dynamic loading profiles is crucial. To this effect, we use a fourth public dataset provided by the Center for Advanced Life Cycle Engineering (CALCE) at the University of Maryland [99]. This dataset uses the ‘INR 18650-20R’ cells (LiNiMnCo/Graphite) with a capacity rating of 2Ah. Initially, the cells are charged and then discharged so that the SOC reaches 80%. The cells are then loaded based on the selected loading profile. Current (in Amps) and voltage (in Volts) are logged every second in this test. The true SOC is calculated every second using the ‘Coulomb Counting’ method. Further, the architecture of USAL is optimized based on the task at hand, which in this case is SOC estimations under dynamic loading conditions. Although several architectures for USAL were tried, we only provide details for the best-performing architecture in this case study. For hyper-parameter tuning, USAL was trained using data from the DST cycle recorded at 45℃ and validated using data from the BJDST driving cycle recorded at 25℃. USAL achieved its best performance when p was set to 20. Hence both L_1 and L_2 are defined as LSTM layers with 20 nodes each. Further, the prediction network within USAL was reconfigured to have an LSTM layer (L_3) with 20 nodes. For all layers within USAL, the lookback steps are set to 50. USAL achieved its best results when no changes were made to output functions of neural network layers, λ_r, λ_s and λ_c. These values are thus set to the same values as given in Section 2.3.2.

Table 2-8 can be used to compare the performance of USAL with other state-of-art methods [127], [132], [133]. USAL was trained and tested on the same data as in Table 2-8 for a fair comparison with other benchmarked methods. Using this table, we can observe that USAL outperforms all other benchmarked methods in all our experiments. Figure 2-10 can be used to visualize this SOC estimation performance of USAL for the US06, FUDS, DST, and the BJDST driving cycle. These performance gains can be directly attributed to USAL’s ability to learn implicit, uncorrelated trends in input data. Autoencod-LSTM [24] is a step-by-step method, while the proposed method (USAL) is integrated where each part of the model is trained simultaneously. A comparison of Autoencod-LSTM with USAL (and correlation loss) in Table 2-9 helps illustrate

Table 2-9 Comparison of USAL with State-of-the-art

Method	Train Data	Test Data at 25℃	RMSE (%)	MAE (%)
GPR	FUDS- First 3000 samples at 25℃	US06- First 3000 samples	0.76	2.25
Recurrent GPR w/ two-tap delay			0.63	0.82
Autoregressive Recurrent GPR			0.24	0.81
USAL			**0.59**	**0.50**
Stacked LSTM	DST, US06 at 25℃	FUDS	1.07	0.84
Autoencod-LSTM			1.65	0.99
USAL			**0.60**	**0.48**
Stacked LSTM	FUDS, US06 at 25℃	DST	1.39	2.02
Autoencod-LSTM			1.00	0.66
USAL			**0.57**	**0.45**
SSMPI	DST at 0, 25, and 45℃	US06	-	0.94
USAL			**0.99**	**0.81**
SSMPI		BJDST	-	0.66
USAL			**0.66**	**0.53**



(a) (b) (c) (d)

Figure 2-10 SOC estimation performance of USAL for – (a) the US06 cycle, (b) the FUDS cycle, (c) the DST cycle, and (d) the BJDST cycle.

the contributions of this work in real-world applications. A comparison of Autoencod-LSTM [112] with USAL but without correlation loss ($\boldsymbol{\lambda_c = 0}$) in Table 2-9, helps illustrate the effectiveness of the proposed multi-task training strategy. Thus, we can conclude that USAL is a state-of-art method supporting SOC estimations even under dynamic driving conditions. Furthermore, USAL's size (<100KB) and its ability to capture aging and degradation trends make it a state-of-art algorithm that can be integrated into a modern BMS for accurate and reliable SOC estimations. In our experiments, the average training time for USAL was about 1200 secs (300 epochs), and the average estimation time was about 0.65ms per observation.

2.4 Conclusion

This work introduces the Uncorrelated Sparse Autoencoder with Long Short-Term Memory (USAL) for long-term SOC estimations. Unlike existing methods [19], [43], [112], the use of a multi-task training strategy in USAL helps it learn efficient higher-dimensional encodings given limited training data. Such encodings may then be used to identify the effect of temporal and aging patterns for accurate SOC estimations. While existing methods fail to identify cell aging and degradation-related changes, the correlation, reconstruction, sparsity, and prediction penalties within USAL force diversity in the learned encodings by penalizing for high multicollinearity. Based on our experiments, USAL routinely outperforms other data-driven models for long-term SOC estimations. Significant improvements are observed when evaluated under varying datasets, cell configurations, charge-discharge profiles, and ambient temperature conditions. While the proposed method seems to have a high implementation cost, this cost is only a one-time cost. In addition, the benefits that can be achieved by implementing this method in a BMS are priceless, especially with rise propositions such as 'cloud-based' BMS systems in intelligent vehicles [132], [134]. For future work, variations of USAL for Remaining Useful Life (RUL) and battery reliability predictions are of high interest [46], [52], [104].

Chapter 3 CGMM: Conditional Gaussian Mixture Model for Predictive Analytics Using Bayesian Statistics

Modern data collection practices allow manufacturers to keep track of vehicle and part health in a modern vehicle. Every time a vehicle is serviced under warranty, a claim is reported in the manufacturer's database. This helps the manufacturer develop robust predictive analytics for forecasting future warranty claims. However, forecasting warranty claims for modern vehicles is a reliability challenge for most manufacturers. Several factors increase the complexity of warranty claims forecasting, including the limited number of claims reported at the early launch stage, reporting delays, dynamic change in the fleet size, and design/manufacturing adjustments for the production line. The aggregated data drift of those complexities is henceforth referred to as the "warranty data maturation" phenomenon. Unfortunately, most of the existing models for warranty claims forecasting fail to consider warranty data maturation explicitly. This work address warranty data maturation by proposing the Conditional Gaussian Mixture Model (CGMM). CGMM uses historical warranty data from similar products to develop a robust prior joint Gaussian mixture distribution of warranty trends at both the current and future maturation levels. CGMM then utilizes Bayesian theories to estimate the conditional posterior distribution of the warranty claims at the future maturation level conditional on the warranty data available at the current maturation level. The CGMM identifies non-parametric temporal warranty trends and automatically clusters products into latent groups to establish (learn) an effective prior joint distribution.

3.1 Introduction

Warranty performance reflects customer satisfaction and instills a sense of product reliability in consumers' minds. Often warranty can be a deal-breaker for many consumers when all other product features are comparable. Although warranties help market the product better, traditionally, offering long-term warranties is costly for the manufacturer. The cost of warranty and recalls for a defective product can be quite considerable, deteriorating the company's profits. Due to associated costs, warranty claims forecasting has been an important topic of discussion for a long time [135]–[138]. As explained by Krivtsov [53], warranty claims forecasting can either follow a 'Univariate Approach' (considering either time or usage) or a 'Bivariate Approach' (considering both time and usage). Models in either of these approaches can further be divided into five types. More details about each approach can be found in [47], [50], [139]–[146].

Researchers have successfully tried forecasting warranty claims using 'Time-series models' such as ARIMA [21]. Further, more complex approaches like ones using Kalman filters have also been tried [147]. Wasserman and Sudjianto [148] use three approaches viz., Orthogonal series, Artificial Neural Networks (ANN), and Kalman filters to forecast warranty claims only to conclude that ANNs offer the best results. Wasserman and Sudjianto [148] employ ANNs for forecasting warranty claims in one of the earliest works. Wu and Akbarov [149] introduce the weighted SVR for forecasting warranty claims when repair rates are available. Recently, Khoshkangini *et al.* [150] used the gradient boost classifiers for forecasting warranty claims using logged vehicle data. In recent work, Shokouhyar *et al.* [151] have used the random forest method for forecasting warranty claims using social media data. A detailed discussion of various Machine learning (ML) methods for forecasting warranty claims can be seen in [152], [153].

Most mentioned models like the NHPP consider a parametric or a semi-parametric approach. Although they achieve promising results, literature proves that there is a better chance for a non-parametric approach in forecasting claims with higher accuracy when newly launched products are under consideration [154]. In real-life scenarios, during the early stages (of a new product), a component's actual seasonality (another significant factor affecting failure rate trends) is unknown. Hence, forcing a seasonal trend based on a parametric model seems farfetched. Alternatively, there could be multiple seasonal patterns. Therefore, choosing one particular seasonality pattern might be inadequate. Furthermore, all survival models and ANN's are estimated/trained on a dataset specific to a particular family of products. This may cause the survival models to force a product-family-specific trend and be less generalizable. ANN's too, are prone to overfitting due to the same problem. To remedy this issue, researchers traditionally have been training and deploying these models only on 'similar products.' Unfortunately, the concept of 'similar products' is ill-defined. Fredette and Lawless [155] define products' similarity as the Pearson correlation based on claim rates. Although this works well for the authors (Fredette and Lawless [155]), it should be noted that their case study is based on data that shares most of its components and hence the failure characteristics.

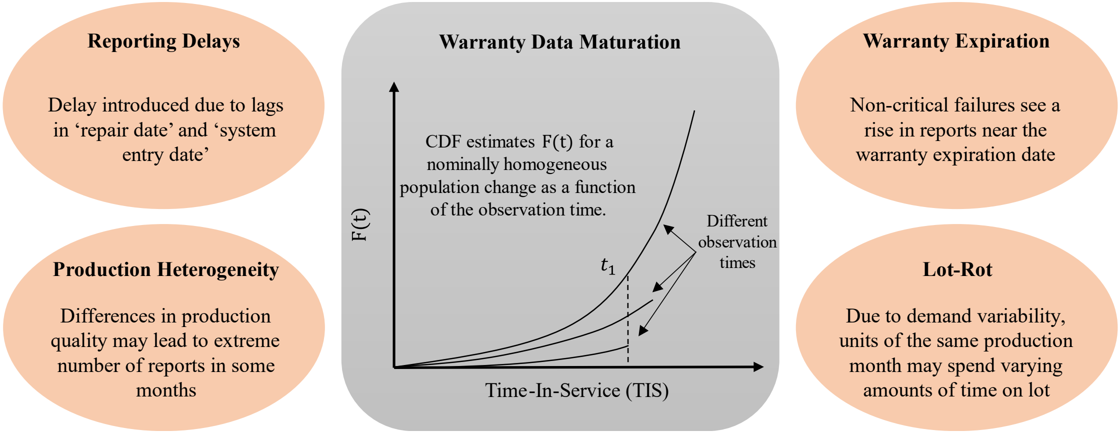

Reporting Delays
Delay introduced due to lags in 'repair date' and 'system entry date'
Production Heterogeneity
Differences in production quality may lead to extreme number of reports in some months
Warranty Data Maturation
CDF estimates F(t) for a nominally homogeneous population change as a function of the observation time.
F(t)
t_1
Different observation times
Time-In-Service (TIS)
Warranty Expiration
Non-critical failures see a rise in reports near the warranty expiration date
Lot-Rot
Due to demand variability, units of the same production month may spend varying amounts of time on lot

Figure 3-1 Effect of the warranty data maturation phenomenon on observed claims data and the common root causes.

‘Warranty Data Maturation’ is a significant problem in the field of warranty claims forecasting [53], [54]. This phenomenon causes the claim trends for a nominally homogeneous population to change as a function of the observation time [53]. Figure 3-1 shows this effect and some common reasons that lead to the warranty data maturation phenomenon. Although some work has been done to address this problem [53], [54], most mentioned warranty claims forecasting methods only work for a shorter horizon (<12 months). Although Gupta *et al.* [142] perform a bivariate claims analysis for up to 90 weeks, they assume no reporting delays. As a result, problems related to warranty data maturation are not so severe. While trying to address warranty data maturation issues, Kalbfleisch *et al.* [48] propose correcting reporting delays by adjusting the risk set of the underlying hazard function, proportional to the probability of the lag time. Krivtsov and Yevkin [156] compensate for the seasonality in warranty data by introducing a time-dependent covariate into the underlying survival model. Kleyner and Elmore [24] construct a parametric structure over the location-scale parameters of the underlying survival distributions, evaluated at different observation times.

In this work, to overcome the limitations of the existing parametric methods and to address problems with warranty data maturation, we propose a novel approach under the Bayesian framework known as the Conditional Gaussian Mixture Model (CGMM). Whereas traditional warranty claims forecasting models are used for short-horizon warranty claims forecasting [157], the CGMM performs much better, with a median error of about 2% for forecasts over 18 months. The main contribution of the proposed approach lies in the usage of the Bayesian updating scheme used to map the current warranty maturation levels to the future warranty maturation levels. CGMM offers the following advantages:

1. **Immature-to-mature warranty mapping**: CGMM maps the immature warranty claims trends to the mature warranty claims trends by modeling the joint distribution of mature and immature warranty trends of historical products as a Gaussian Mixture Model (GMM).
2. **Non-parametric**: CGMM does not force a time-dependent parametric function for forecasting the number of claims.
3. **Product-to-Product Variability**: CGMM uses conditional Bayesian inference when considering the similarity of products in a cluster.
4. **Robust to over-fitting**: CGMM effectively leverages historical warranty data. This reduces over-fitting because the training data comprises several products rather than one individual product.
5. **Uncertainty Quantification (UQ)**: Survival and stochastic models provide an UQ measure through the prediction interval using known statistical distributions. One step further, the CGMM provides an epistemic UQ after updating the GMM parameters (Bayesian update) and correcting for bias (conditional distribution). The credible interval for the posterior predictive distribution is used as an UQ measure in the CGMM.

3.2 Gaussian Mixture Models for Warranty Claims Forecasting

GMMs are commonly used for unsupervised learning [86], [158], [159]. In the field of warranty claims forecasting, the commonly constructed GMMs assume that all products are represented by a set of K latent clusters and that each product belongs to only one true cluster; however, since the true cluster is not available, the products are modeled as a mixture of the K latent clusters where, the number of claims at different time-in-service (TIS) of each latent cluster is jointly modeled using a multivariate Gaussian distribution. Consequently, the number of claims at different TIS points of any product are jointly modeled as a mixture of K multivariate Gaussian distributions. Note that these claims at different TIS points of any product represent a varying degree of warranty maturation. The mixing coefficients are then calculated using normalized likelihoods given by the observed warranty data (number of claims at different TIS points) for the product of interest. Figure 3-2 summarizes the GMM framework for modeling the number of claims at pre-specified TIS points. The joint random variable $\begin{bmatrix} \boldsymbol{Y} \\ \boldsymbol{Z} \end{bmatrix}$ corresponds to the observed warranty claims at an immature ($\boldsymbol{Y}$) and a mature warranty level ($\boldsymbol{Z}$).

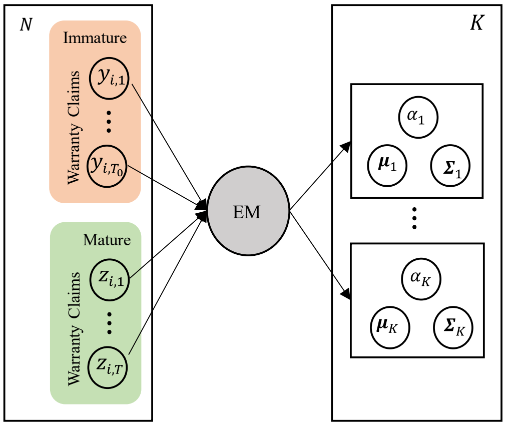


Figure 3-2 The Gaussian Mixture Model (GMM) for warranty forecasting. N is the number of historical training products, and K is the number of latent clusters.

Definition 1: Any intermediate warranty maturation level (immature warranty level) is defined by the latest observed TIS for the product of interest. For example, Ford Fusion 2018 in May 2019 shows a similar immature warranty level as Ford Fusion 2019 in May 2020.

Definition 2: A product reaches full warranty maturation (mature warranty level) when all sold manufactured items are past the warranty period. For example, Ford Fusion 2015, after 36 months from the last sold Ford Fusion 2015.

From the theory of mixture models [159] and the law of total probability, the joint distribution of $\boldsymbol{X} = \begin{bmatrix} \boldsymbol{Y} \\ \boldsymbol{Z} \end{bmatrix}$ is expressed as,

$$f_{\boldsymbol{X}}(\boldsymbol{x}_i') = \sum_{k=1}^{K} \alpha_k f_{\boldsymbol{X}|m}(\boldsymbol{x}_i' |\, m_i = k) \tag{3-1}$$

Here, $m_i \in \{1 \dots K\}$ is the latent variable representing the mixture component of $\boldsymbol{X}$, α_k is the mixing coefficient of the k^{th} cluster, $\boldsymbol{x}_i' = \left[y_{i,1}, \dots, y_{i,T_0}, z_{i,1}, \dots, z_{i,T}\right]'$ is the observed vector of the number of claims at different TIS points for the i^{th} training product at both the immature (y) and mature warranty levels (z), T_0 is the last observed TIS point at the immature warranty level, and T is the last observed TIS at the mature warranty level.

Without loss of generality, we assume that the normalized number of claims follows a normal distribution. Such assumption is statistically justifiable for case studies where the sample size in each cluster is sufficiently large (>200); otherwise, a closer investigation is required to verify the normality assumption. Under the normality assumption, the conditional probability distribution for $\boldsymbol{X}$ is given by,

$$f_{\boldsymbol{X}|m}(\boldsymbol{x}|m = k) = \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k). \tag{3-2}$$

Thus, the distribution of $\boldsymbol{X}$ can be expressed as,

$$f_{\boldsymbol{X}}(\boldsymbol{x}) = \sum_{k=1}^{K} \alpha_k \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \tag{3-3}$$

Given the recorded claims of the historical products $\boldsymbol{x}_1' \dots, \boldsymbol{x}_N'$, the log-likelihood of the GMM is expressed as,

$$\log\left(f_{\boldsymbol{X}}(\boldsymbol{X}'|\boldsymbol{\Theta})\right) = \log\left(\sum_{k} f_{\boldsymbol{X},m}(\boldsymbol{X}', m = k|\boldsymbol{\Theta})\right) \tag{3-4}$$

where $\boldsymbol{\theta}$ are unknown parameters of the GMM such that, $\boldsymbol{\theta} = \{\boldsymbol{\mu}_1 \dots \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_1 \dots \boldsymbol{\Sigma}_k, \alpha_1 \dots \alpha_k\}$. There exist no closed-form solution for the maximum likelihood estimator of $\boldsymbol{\theta}$ [159]. Thus, the

Expectation-Maximization (EM) algorithm is utilized to find the mean and covariance of the K clusters. More details on the EM algorithm can be seen in our work [52] and Appendix A.

To decide on the optimal number of clusters K, the EM algorithm is applied for different choices of K and the value of K that maximizes the Akaike Information Criterion (AIC) is selected. The AIC metric penalizes for (i) low likelihood, and (ii) an increased number of parameters (clusters); therefore, it provides a reasonable balance to maintain an acceptable performance (acceptable likelihood) without overfitting (over-complicating the model with excessive parameters). Other metrics are also commonly used for model selection, including the Bayesian Information Criterion (BIC) [160]–[162]. In this work, we use the AIC because it is shown to be effective even if the true model is not in the pool of potential GMMs.

While the GMM provides a suitable framework to estimate the joint distribution of the number of claims at different TIS points, it cannot be applied for recently launched products before the end of the warranty. This is because the GMM requires knowing the number of claims for all TIS points, which is not readily available before the end of the warranty period. Therefore, the following section extends the GMM to enable warranty claims forecasting for recently launched products that did not reach the end of the warranty period.

3.3 Conditional Gaussian Mixture Model for Warranty Claims Forecasting

To enable using GMMs for warranty claims forecasting, we first need to distinguish between the observed TIS points and the TIS points to be forecasted for the recently launched products. We then utilize a multivariate Gaussian distribution to calculate the conditional distribution of the number of claims of the TIS points to be forecasted, given the number of claims at the observed TIS points. Figure 3-3 shows the operational framework of the CGMM.

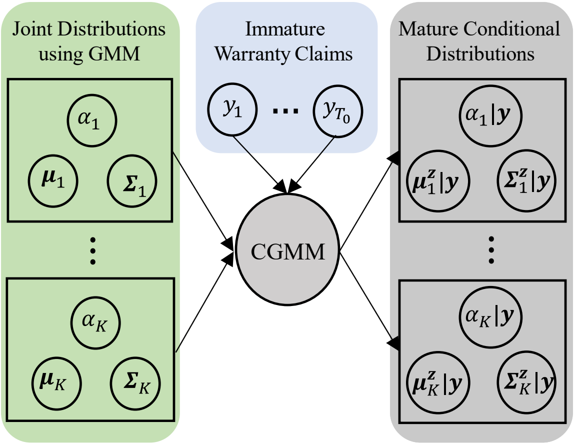

Joint Distributions using GMM
α_1
$\boldsymbol{\mu}_1$
$\boldsymbol{\Sigma}_1$
α_K
$\boldsymbol{\mu}_K$
$\boldsymbol{\Sigma}_K$
Immature Warranty Claims
y_1
y_{T_0}
CGMM
Mature Conditional Distributions
$\alpha_1|\boldsymbol{y}$
$\boldsymbol{\mu}_1^z|\boldsymbol{y}$
$\boldsymbol{\Sigma}_1^z|\boldsymbol{y}$
$\alpha_K|\boldsymbol{y}$
$\boldsymbol{\mu}_K^z|\boldsymbol{y}$
$\boldsymbol{\Sigma}_K^z|\boldsymbol{y}$

Figure 3-3 Proposed CGMM for warranty claims forecasting.

3.3.1 Problem Formulation

First, we split the number of claims into two sets. The first set corresponds to the observed TIS points (immature warranty level), and the second set corresponds to the TIS points that need to be forecasted (mature warranty level). Furthermore, the joint distribution of $\begin{bmatrix}\boldsymbol{Y}\\ \boldsymbol{Z}\end{bmatrix}$ is expressed as a mixture of multivariate Gaussian distribution and is given by,

$$\boldsymbol{X} = \begin{bmatrix}\boldsymbol{Y}\\ \boldsymbol{Z}\end{bmatrix} \sim \sum_{k=1}^{K} \alpha_k \mathcal{N}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k), \tag{3-5}$$

Here, $\boldsymbol{Y}$ represents the number of claims for the observed TIS points, and $\boldsymbol{Z}$ represents the number of claims for the TIS points to be forecasted at the end of the warranty period. The training input $\boldsymbol{X}$ used to estimate the prior GMM is designed at the same historical maturation level to account for warranty maturation. Consequently, the distribution of the number of claims for the observed TIS points is given by,

$$\boldsymbol{Y} \sim \sum_{k=1}^{K} \alpha_k \mathcal{N}(\boldsymbol{\mu}_k^{\boldsymbol{y}}, \boldsymbol{\Sigma}_k^{\boldsymbol{y}}) \tag{3-6}$$

where,

$$\boldsymbol{\Sigma}_k = \begin{bmatrix} \boldsymbol{\Sigma}_k^{\boldsymbol{y}} & \boldsymbol{\Sigma}_k^{\boldsymbol{y},\boldsymbol{z}} \\ \boldsymbol{\Sigma}_k^{\boldsymbol{z},\boldsymbol{y}} & \boldsymbol{\Sigma}_k^{\boldsymbol{z}} \end{bmatrix} \tag{3-7}$$

$$\boldsymbol{\mu}_k = \begin{bmatrix} \boldsymbol{\mu}_k^{\boldsymbol{y}} \\ \boldsymbol{\mu}_k^{\boldsymbol{z}} \end{bmatrix} \tag{3-8}$$

Here, $\boldsymbol{\Sigma}_k$ is the covariance matrix of the observed and to be forecasted TIS points and is given by eq. (3-7).

3.3.2 Conditional Predictive Distribution Using CGMM

In CGMM, we utilize the Bayesian version of the multivariate Gaussian distribution to calculate the conditional predictive distribution, $f_{\boldsymbol{Z}|\boldsymbol{Y}}(\boldsymbol{z}_i|\boldsymbol{y}_i)$. Proofs for deriving eq. (3-9) can be seen in our work [52] and Appendix A.

$$f_{\boldsymbol{Z}|\boldsymbol{Y}}(\boldsymbol{z}_i|\boldsymbol{y}_i) = \sum_{k=1}^{K} (\alpha_k|\boldsymbol{y}_i)\, \mathcal{N}(\boldsymbol{z}_i; \boldsymbol{\mu}_k^{\boldsymbol{z}}|\boldsymbol{y}_i, \boldsymbol{\Sigma}_k^{\boldsymbol{z}}|\boldsymbol{y}_i), \tag{3-9}$$

where,

$$\alpha_k|\boldsymbol{y}_i = \frac{\alpha_k * f_{\boldsymbol{Y}}(\boldsymbol{y}_i|\boldsymbol{\mu}_k^{\boldsymbol{y}}, \boldsymbol{\Sigma}_k^{\boldsymbol{y}})}{\sum_{r=1}^{K}\{\alpha_r * f_{\boldsymbol{Y}}(\boldsymbol{y}_i|\boldsymbol{\mu}_r^{\boldsymbol{y}}, \boldsymbol{\Sigma}_r^{\boldsymbol{y}})\}} \tag{3-10}$$

$$\boldsymbol{\mu}_k^{\boldsymbol{z}}|\boldsymbol{y}_i = \boldsymbol{\mu}_k^{\boldsymbol{z}} - \left(\boldsymbol{\Sigma}_k^{\boldsymbol{y},\boldsymbol{z}}\right)' \left(\boldsymbol{\Sigma}_k^{\boldsymbol{y}}\right)^{-1}\left(\boldsymbol{\mu}_k^{\boldsymbol{y}} - \boldsymbol{y}_i\right) \tag{3-11}$$

$$\boldsymbol{\Sigma}_k^{\boldsymbol{z}}|\boldsymbol{y}_i = \boldsymbol{\Sigma}_k^{\boldsymbol{z}} - \left(\boldsymbol{\Sigma}_k^{\boldsymbol{y},\boldsymbol{z}}\right)' \left(\boldsymbol{\Sigma}_k^{\boldsymbol{y}}\right)^{-1}\left(\boldsymbol{\Sigma}_k^{\boldsymbol{y},\boldsymbol{z}}\right) \tag{3-12}$$

The conditional Bayesian inference partially accounts for the product-to-product variability by updating the posterior distribution, given the number of claims of the observed TIS points for the product of interest. Furthermore, the CGMM provides the conditional distribution for the number of claims to be forecasted, given the available history of the observed number of claims. Unlike existing models, such advantages of the CGMM make it attractive for warranty claims forecasting.

3.3.3 Unbiased Conditional Predictive Distribution

In many practical applications, assuming that products are a mixture of latent clusters introduces an algorithmic bias. This is due to the natural cluster-to-cluster variability. Therefore, it is natural to consider only the most likely cluster to avoid such algorithmic biases. Mathematically, we search for cluster k^* with the maximum normalized likelihood given by,

$$k^*|\boldsymbol{y}_i = \underset{k}{\operatorname{argmax}}(\log f_{\boldsymbol{Y}}(\boldsymbol{y}_i|\boldsymbol{\mu}_k^{\boldsymbol{y}}, \boldsymbol{\Sigma}_k^{\boldsymbol{y}})) \tag{3-13}$$

Given the most likely cluster $k^*|\boldsymbol{y}_i$, the enhanced predictive conditional distribution of the CGMM is expressed as,

$$f_{\boldsymbol{Z}|\boldsymbol{Y}}(\boldsymbol{z}_i|\boldsymbol{y}_i) = \mathcal{N}(\boldsymbol{z}_i; \boldsymbol{\mu}^{\boldsymbol{z}}|\boldsymbol{y}_i, \boldsymbol{\Sigma}^{\boldsymbol{z}}|\boldsymbol{y}_i) \tag{3-14}$$

where,

$$\boldsymbol{\mu}^{\boldsymbol{z}}|\boldsymbol{y}_i = \boldsymbol{\mu}_{k^*}^{\boldsymbol{z}} - \left(\boldsymbol{\Sigma}_{k^*}^{\boldsymbol{y},\boldsymbol{z}}\right)' \left(\boldsymbol{\Sigma}_{k^*}^{\boldsymbol{y}}\right)^{-1}\left(\boldsymbol{\mu}_{k^*}^{\boldsymbol{y}} - \boldsymbol{y}_i\right) \tag{3-15}$$

$$\boldsymbol{\Sigma}^{\boldsymbol{z}}|\boldsymbol{y}_i = \boldsymbol{\Sigma}_{k^*}^{\boldsymbol{z}} - \left(\boldsymbol{\Sigma}_{k^*}^{\boldsymbol{y},\boldsymbol{z}}\right)' \left(\boldsymbol{\Sigma}_{k^*}^{\boldsymbol{y}}\right)^{-1}\left(\boldsymbol{\Sigma}_{k^*}^{\boldsymbol{y},\boldsymbol{z}}\right) \tag{3-16}$$

3.4 Case Study Using Early Launched Automotive Parts

3.4.1 Description

Most automotive companies host a reporting platform for dealers to report warranty claims. Although these platforms' primary intent is to keep track of reported warranty claims, they also contain information on failure modes and failure time and/or usage that can be used for subsequent engineering and statistical analysis. Note that claims are sometimes delayed for verification and depend on the individual dealership's reporting practices. In addition, the number of reported claims is expected to increase with more vehicles being continuously produced and sold. This increases the challenge of estimating the warranty claims as a function of the TIS because of the heterogeneous production quality, usage variability, seasonal effects, and other warranty data maturation factors. Furthermore, various failure modes and a highly non-linear interaction between the parts of the vehicle introduce some non-trivial behavior for the trajectory of the number of claims as a function of TIS. Such non-linear trajectories often lead to poor long-term (>12 months) forecasting using traditional parametric survival models (such as Weibull at a part level) or stochastic models (such as the NHPP at a system or vehicle level).

The automotive warranty claims dataset (scaled for confidentiality) considered in this case study includes more than 15,000 vehicle parts installed on more than 10 million vehicles for four historical model years (2010-2013). Note that some parts may be only available on one model year but not the other; however, there is still a non-trivial correlation that links different vehicle parts and different model years. The claims are recorded as events reported from multiple dealerships. We aggregate those events monthly at the part level for every unique combination of part code, vehicle code, and model year for 36 months. Parts from model years 2010, 2011, and 2012 are used for training, and parts from the model year 2013 are considered for validation as described in Table 3-1.

Table 3-1 Training and Validation Data

Dataset	Number of unique parts
Training (2010, 2011 and 2012)	10,980
Validation (2013)	4,663
Total	15,643

In this case study, August 31 (AUG 31) of the year following the model year represents 18 months after the first sold manufactured vehicle. It thus represents the cumulative claims at an immature warranty level since not all sold vehicles have passed the warranty period. Figure 3-4 shows the cumulative number of claims from thousands of vehicles of a randomly selected validation part from the model year 2013 at AUG 31-2014 in red circles (claims at immature warranty level), the cumulative number of claims after all the sold vehicles pass the warranty period in black stars (claims at mature warranty level), and the cumulative number of claims predicted by the CGMM as green triangles.

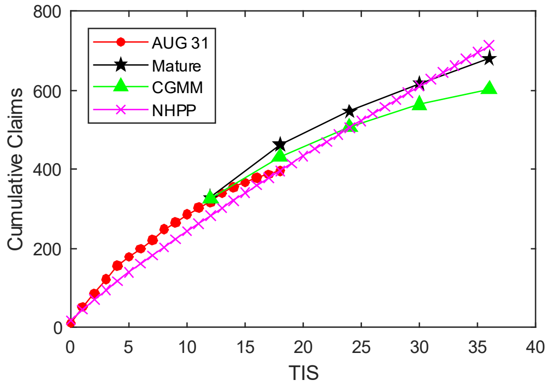


Figure 3-4 Example validation part used in this study.

3.4.2 Model Setup

Table 3-2 Log-likelihood and AIC as a function of K Latent Clusters

Number of clusters ($\boldsymbol{K}$)	**Log-likelihood**	**AIC**
1	-8.1999	1.6406
2	-7.1400	1.4293
3	-6.9445	1.3908
4	-6.9223	1.3871
5	-6.8175	1.3667
6	-6.7054	1.3450
7	-6.7612	1.3568
8	**-6.6288**	**1.3309**
9	-6.6333	1.3325
10	-6.7136	1.3492

The observed cumulative number of claims for the first 18 months-in-service are denoted by $\boldsymbol{y_i} = [y_{i,0}, y_{i,1}, y_{i,2}, \dots, y_{i,18}]'$ and the forecasted cumulative number of claims for the TIS 12, 18, 24, 30, and 36 months after all the sold vehicles pass the warranty period are denoted by $\boldsymbol{z_i} = [z_{i,12}, z_{i,18}, z_{i,24}, z_{i,30}, z_{i,36}]'$. Note that $y_{i,12}$ is not necessarily equal to $z_{i,12}$ due to warranty data maturation issues. For the forecasted months-in-service, the CGMM outputs the distribution of possible trajectories for $\boldsymbol{z_i}|\boldsymbol{y_i} \sim \mathcal{N}(\boldsymbol{\mu^z}|\boldsymbol{y_i}, \boldsymbol{\Sigma^z}|\boldsymbol{y_i})$ as shown in eq. (3-14). The CGMM thus provides uncertainty quantification $\boldsymbol{\Sigma}^{\boldsymbol{z}}|\boldsymbol{y}_i$ around the expected value of $\boldsymbol{z}_i|\boldsymbol{y}_i$. The number of latent clusters K for the CGMM is varied between 1 and 10, and the number of clusters with the lowest Akaike Information Criterion (AIC) is selected. Table 3-2 summarizes the log-likelihood and AIC for all the trained CGMMs with a different number of latent clusters.

3.4.3 Results

Using results from Table 3-2, we define $K = 8$ and train the CGMM on parts from the model years 2010, 2011, and 2012. The distributions of the percentage error (PE) calculated using eq. (3-17) between the actual (mature) and predicted number of claims for the five TIS points (12, 18, 24, 30, and 36 months) are as shown in Figure 3-5 (a-e).

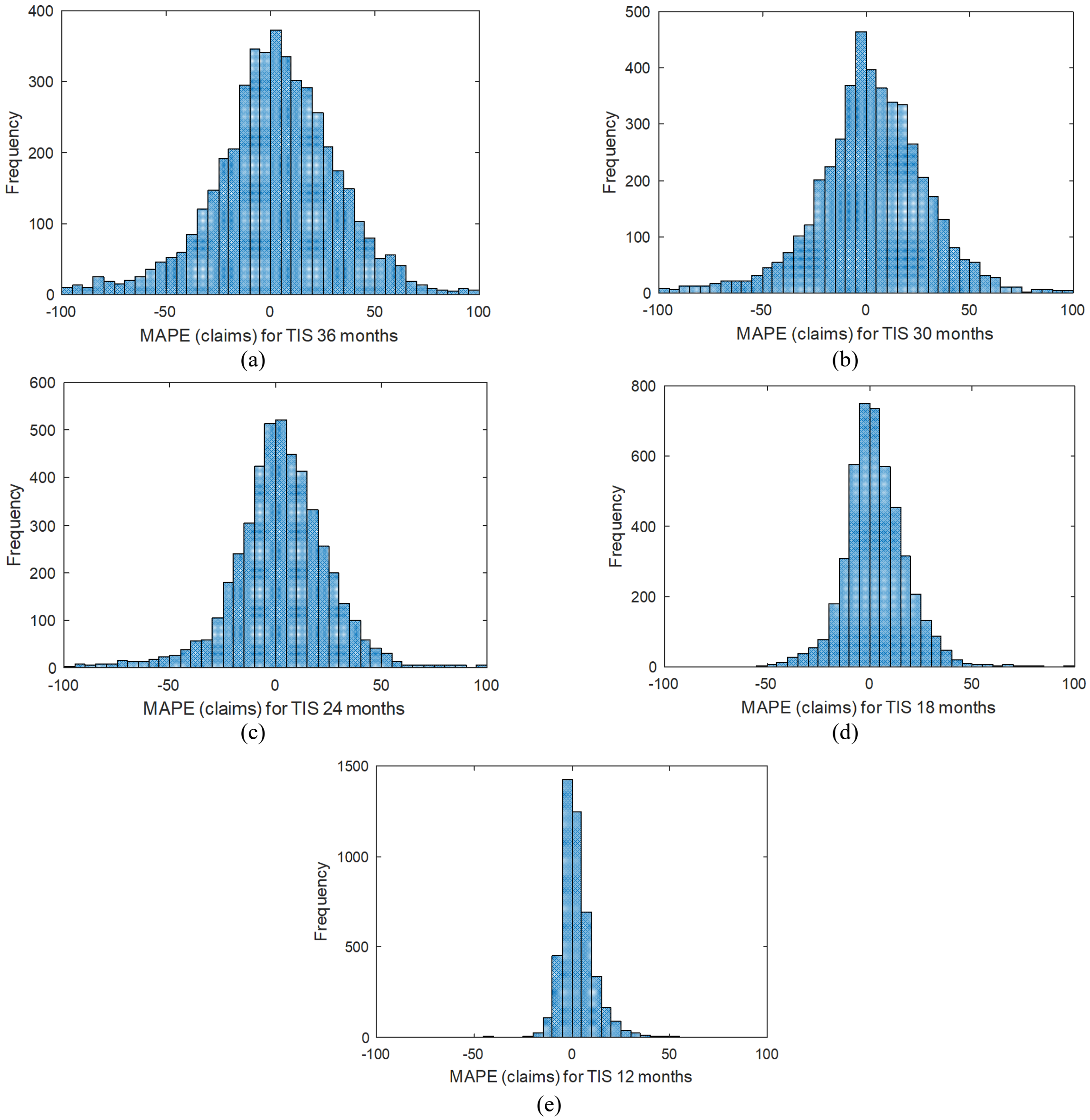
$$PE(i,t) = 100 \times \frac{z_{i,t} - \mu^z|y_{i,t}}{z_{i,t}} \tag{3-17}$$

Furthermore, Table 3-3 summarizes some selected quantiles of the error distributions for the forecasted TIS 12, 18, 24, 30, and 36 months. This table clearly shows that the error increases with TIS.

Table 3-3 Selected Quantiles for the Error Distributions

TIS (months)	90%	75%	Median	25%	10%
36	-37.8%	-15.6%	2.05%	20.4%	37.7%
30	-31.7%	-12.5%	2.31%	18.5%	34.0%
24	-23.3%	-9.6%	2.34%	15.4%	28.6%
18	-13.6%	-6.1%	2.01%	11.4%	21.8%
12	-5.81%	-2.69%	1.00%	6.38%	13.1%

This is expected because the uncertainty is accumulated in long-term forecasting. Table 3-3 also shows that the median percentage error is centered around 2%, which is very close to the target of 0%. Overall, Table 3-3 shows that the first and third quartiles for all the forecasted TIS are bounded between -15% and 20%. This is considered to be a reasonably good performance for long-term forecasting. It is important to note that the MAPE for TIS 12 and TIS 18 months is sufficiently small. It can thus be concluded that the CGMM is accurately shifting/updating the trend from data reported until AUG 31 towards the mature claims data. The MAPE for those shifts is reasonably small, which provides confidence that the CGMM successfully leveraged parts from previous model years to capture those shifts.



(a)

(b)

(c)

(d)

(e)

Figure 3-5 The distribution of PE for TIS (a) 36 months, (b) 30 months, (c) 24 months, (d) 18 months, and (e) 12 months.

Traditional models, such as NHPP, show poor performance for long-term forecasting [21], [163]. They tend to show an asymmetric error distribution biased to under-predicting the number of claims due to reporting delays and limited claims counts observed at TIS 16, 17, and 18 months on AUG 31 (the 18th TIS observation time). All observations in Table 3-3 show that the CGMM is unbiased and successfully forecasts the number of cumulative claims for the next 18 months (up to TIS 36 months) with an error of +/-37%. In the following section, we compare the CGMM to the standard warranty claims forecasting approach, the NHPP.

3.4.4 Comparison to Standard NHPP

A standard reliability model for the repairable system is the NHPP [164]. Below, we explain the necessity of developing approaches similar to the CGMM to avoid bias and thus improve over the performance shown by NHPP and its derivatives. Note that the classical estimation of the NHPP only leverages the warranty claims from the part being modeled. In contrast, the CGMM leverages claims from all other parts of all previous model years to capture non-trivial trends better.

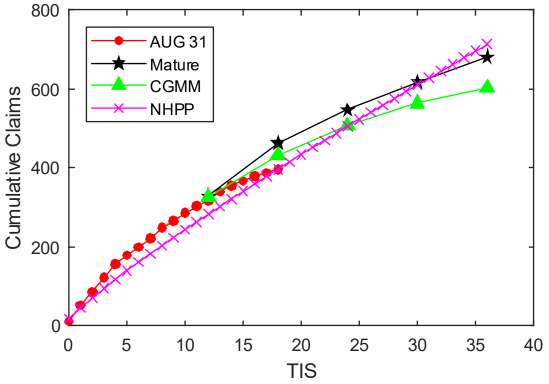


Figure 3-6 Cumulative number of claims for validation part 1 as a function of TIS (in months).

The first case occurs when the NHPP shows acceptable forecasting performance without capturing the actual trend of the warranty claims for a specific part from thousands of vehicles. An example of a validation part from the model year 2013 is shown in Figure 3-6, where the predicted claims using the NHPP are shown in purple crosses. This figure clearly shows that the NHPP fails to capture the non-linearity of the warranty claims, which is mainly attributed to the parametric nature of the NHPP.

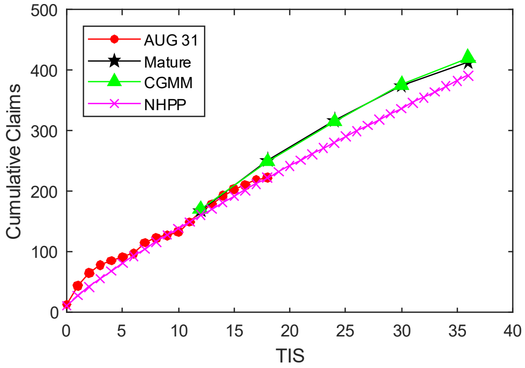


Figure 3-7 Cumulative number of claims for validation part 2 as a function of TIS (in months).

The second case occurs when the NHPP shows acceptable fitting performance before AUG 31 but fails to show accurate forecasting performance. An example of a validation part is shown in Figure 3-7. For this specific example, in Figure 3-7, the CGMM shows a much better performance than the NHPP because it leverages the historical knowledge that links between mature warranty claims and a partial subset of recorded claims before AUG 31. Furthermore, the CGMM does not force any time-dependent parametric assumptions and utilizes Bayesian schemes to update the distribution of possible trends, given the warranty claims recorded before AUG 31. This proves a need to leverage historical warranty trends to learn the dynamics and shifts in the warranty claims trajectory between the immature observations and the mature claim data.

The third case occurs when the NHPP captures the warranty claims' actual curvature but shows a bias. An example of a validation part is shown in Figure 3-8. This figure shows that the NHPP successfully captures the curvature of the warranty claims but shows a bias, mainly attributed to the limited number of claims at TIS 13, 14, 15, 16, 17, and 18 months. The figure also shows that the CGMM successfully learns such biases from historical trends and shows higher effectiveness in capturing the curvature and the shift between the observation time AUG 31 and the mature claim data.

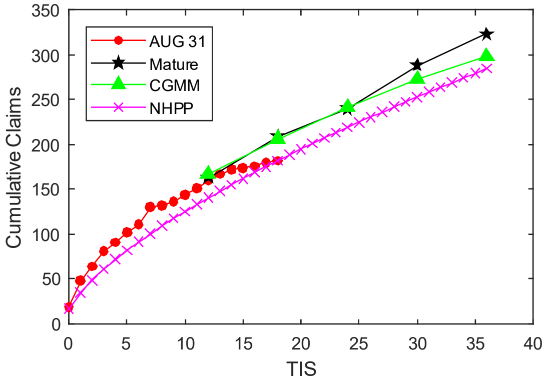


Figure 3-8 Cumulative number of claims for validation part 3 as a function of TIS (in months).

The fourth case occurs when the claims rate is low with a small number of claims at an earlier TIS and then starts to increase for later TIS. An example from validation is shown in Figure 3-9. The figure shows that the NHPP successfully captured the curvature of the warranty trend before AUG 31; however, it failed to adjust for the non-trivial dynamic change in the warranty trend. This figure shows that the CGMM successfully learned such jumps and dynamic behavior from parts of the previous model years.

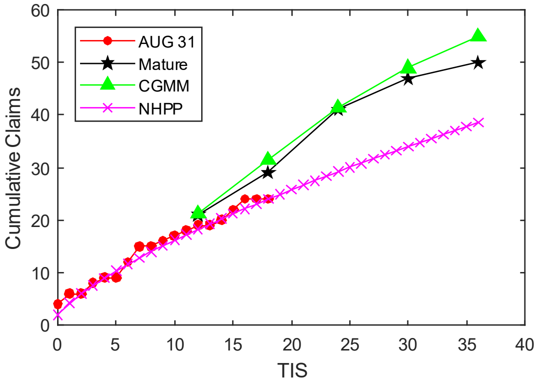


Figure 3-9 Cumulative number of claims for validation part 4 as a function of TIS (in months).

The last case occurs when the NHPP heavily overpredicts the number of claims at the forecasted TIS points, as shown for some validation part in Figure 3-10. This figure shows the NHPP successfully fitted the cumulative claims for the early months (TIS 10 months and before) and started overshooting thereafter. In doing so, the NHPP predicted almost 2500 more claims at TIS 36 than actual data. This is mainly attributed to the warranty data maturation problem. The figure shows that the CGMM successfully captured such a non-trivial trend. The above examples show the importance of leveraging the historical warranty trends. The CGMM is particularly effective here and hence outperforms other traditional models that only leverage warranty claims from the model year of interest.

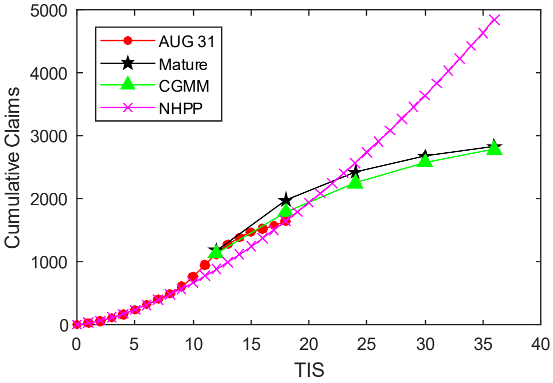


Figure 3-10 Cumulative number of claims for validation part 5.as a function of TIS (in months).

3.4.5 Comparison to Benchmarks in Warranty Forecasting

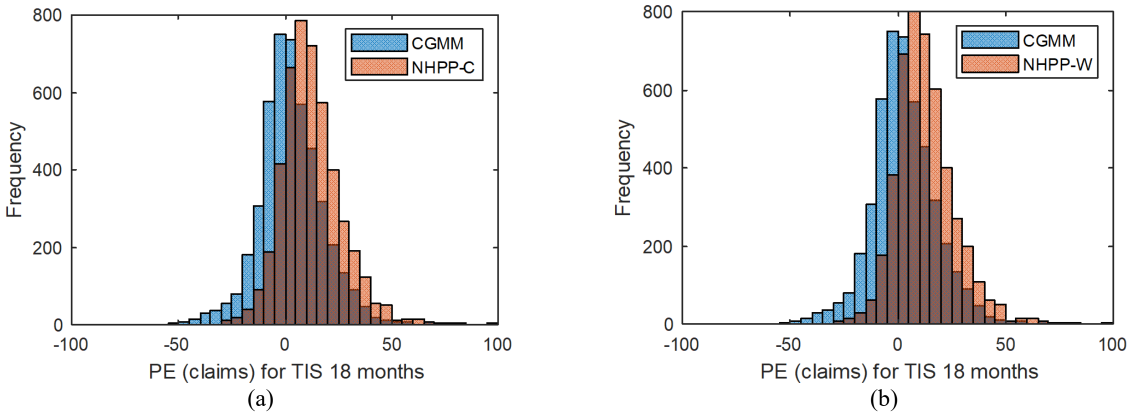
In this section, we compare CGMM to the other two benchmark models. The first, NHPP-W, is similar to the standard NHPP but enforces weights on the log-likelihood that follows a normalized geometric distribution [21]. The second, NHPP-C, is similar to the standard NHPP but enforces upper and lower bounds on the expected number of claims at the forecasted TIS points [163]. The intuition of NHPP-W lies in providing more weights to later TIS points to achieve higher forecasting accuracy. The intuition of the NHPP-C lies in providing upper and lower bounds to avoid extreme over/under-prediction when forecasting. Two metrics are considered for comparison- the mean absolute percentage error (MAPE) as given in eq. (3-18) and the root mean squared error (RMSE) as given in eq. (3-19) for TIS 18, 24, 30, and 36 months. The results in Table 3-4 clearly show that the CGMM outperforms the NHPP-W and NHPP-C for all the forecasted TIS points with lower MAPE and RMSE values. Next, we show the efficacy of the CGMM for warranty maturation compared to NHPP-W and NHPP-C by plotting the distribution of the percentage errors at TIS 18 months.

$$MAPE(i,t) = \frac{100}{4663} \times \sum_{i=1}^{4663} \left| \frac{z_{i,t} - \hat{z}_{i,t}}{z_{i,t}} \right| \tag{3-18}$$

$$RMSE(i,t) = \frac{100}{\sqrt{4663}} \times \sqrt{\sum_{i=1}^{4663} \left(\frac{z_{i,t} - \hat{z}_{i,t}}{z_{i,t}} \right)^2} \tag{3-19}$$

Table 3-4 Comparison with Traditional Warranty Forecasting Models

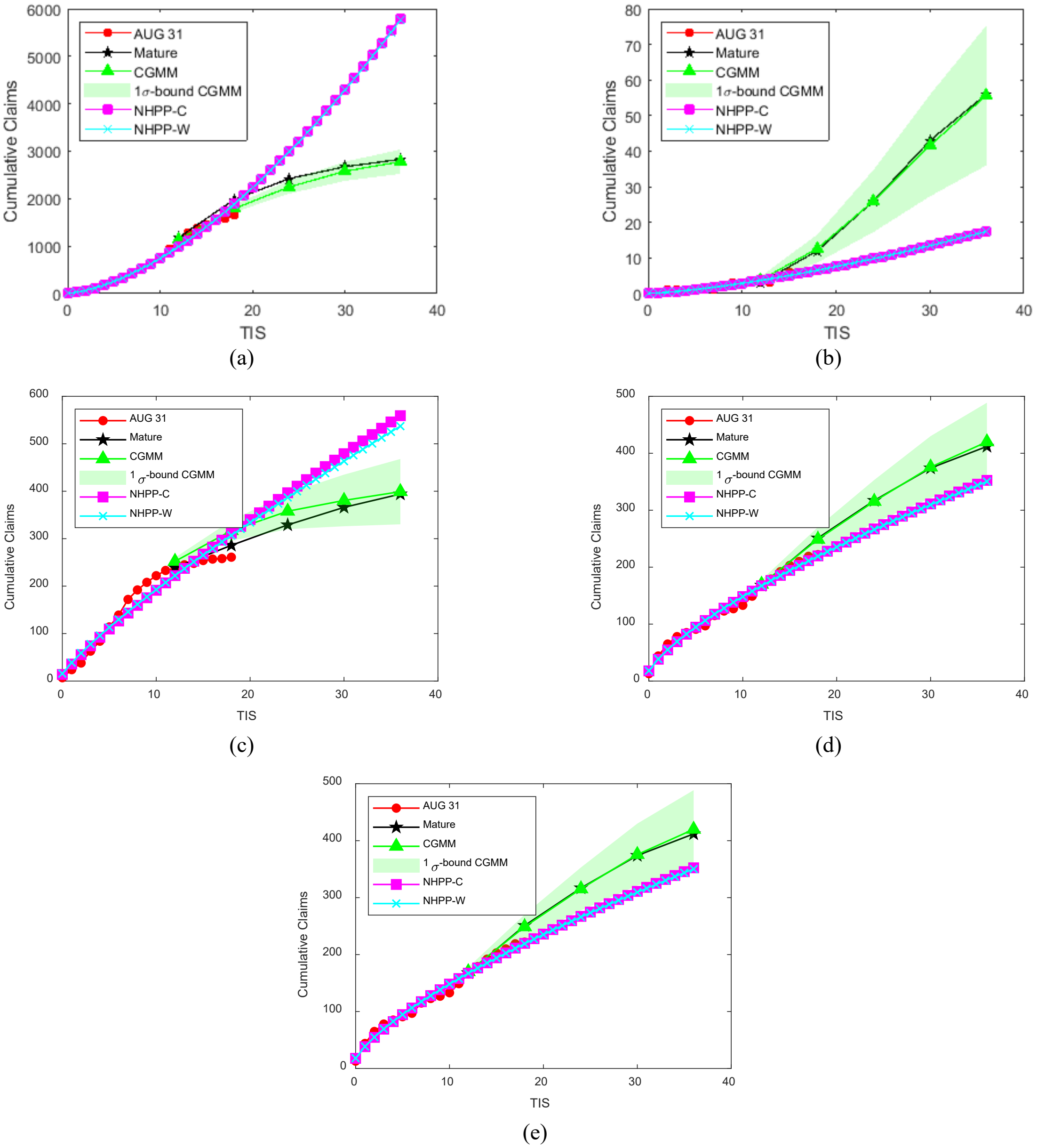
TIS (months)	METRIC	CGMM	NHPP-C	NHPP-W
36	MAPE	**25.60%**	32.20%	31.66%
	RMSE	**38.25%**	145.0%	145.9%
30	MAPE	**21.77%**	24.13%	23.83%
	RMSE	**31.50%**	43.49%	45.80%
24	MAPE	**17.24%**	18.83%	18.93%
	RMSE	**24.15%**	24.95%	36.10%
18	MAPE	**11.65%**	14.47%	15.43%
	RMSE	**16.63%**	18.93%	72.48%



(a)

(b)

Figure 3-11 The distribution for MAPE of CGMM and (a) NHPP-C, and (b) NHPP-W for TIS 18 months.



(a)

(b)

(c)

(d)

(e)

Figure 3-12 The cumulative number of claims for several validation parts as a function of the TIS (months).

Figure 3-11 shows that the MAPE for CGMM is centered near zero, and thus we can conclude that the forecasts by CGMM successfully corrected for maturity at TIS 18. On the other hand, NHPP-C and NHPP-W failed to address maturity, and this can be observed in the skewness of distribution (towards under-prediction). Such observations are expected since the NHPP-C and NHPP-W (unlike the CGMM) do not consider mapping the claims from the immature warranty level to the mature warranty level. This concludes the real-life case study that clearly shows the advantages of the proposed CGMM, which include (i) accurate predictions, (ii) capturing non-trivial trends, (iii) bias correction, and (iv) mapping between the immature and mature spaces. Figure 3-12 shows multiple additional comparisons between the benchmark methods and the proposed model-CGMM.

Figure 3-12 also shows, in shaded green, the uncertainty quantification using the one standard deviation bound for the CGMM. These figures show that (i) the CGMM is reliable and robust for multiple trends, (ii) the CGMM almost consistently outperforms the benchmark methods, (iii) the uncertainty increases as the forecast duration increases, and that is a natural consequence of the unit-to-unit manufacturing, and (iv) the one standard deviation bound almost always covers the actual mature data at the end of the warranty period. These observations provide further evidence of the robustness, precision, and accuracy of the proposed CGMM.

3.4.6 Comparison to Recurrent Neural Networks

Next, we compare the CGMM to a Recurrent Neural Network (RNN). RNNs are known to outperform feedforward neural networks and traditional machine learning models such as support vector regression, random forests, and decision trees. The inputs to the RNN are the TIS of interest, the number of warranty claims for the past four time-steps, and the output is the cumulative warranty claims at the TIS of interest. Table 3-5 summarizes the quantiles for the percentage errors from the trained RNN. Table 3-6 summarizes the MAPE for CGMM and RNN. Long-term forecasts using RNNs are significantly worse than the CGMM, and thus we only showcase the performance of the trained RNN at TIS 12 and 18.

When comparing Table 3-5 for RNN and Table 3-3 for CGMM, we see that the errors from the RNN are not centered. We can also see that this is more evident at TIS 18 months than TIS 12 months. Additionally, the tables show that the quantile ranges are wider (higher variance) for the RNN compared to the CGMM. This helps conclude that the CGMM achieves lower bias and variance than the RNN.

Table 3-5 Selected Quantiles for the Error Distributions of the RNN Predictions for TIS 12 and 18 Months

TIS (months)	**90%**	**75%**	**Median**	**25%**	**10%**
18	2.99%	7.47%	14.50%	23.43%	33.35%
12	-3.05%	0.92%	5.08%	10.56%	17.63%

Table 3-6 The MAPE and RMSE Values for the Predictions of the CGMM and RNN at TIS 12 and 18 Months

TIS (months)	**METRIC**	**CGMM**	**RNN**	**Percentage Improvement**
18	MAPE	**11.65%**	16.86%	30.90%
	RMSE	**16.63%**	20.95%	20.95%
12	MAPE	**6.36%**	8.33%	23.65%
	RMSE	**10.59%**	11.89%	10.93%

Table 3-6 summarizes the MAPE and RMSE of both the CGMM and RNN. The results in Table 3-6 clearly show that the CGMM outperforms the RNN for at TIS 12 and 18 months with an average percentage improvement of around 25% in MAPE and 15% in RMSE. Furthermore, Table 3-6 shows that the RNN performance deteriorates at TIS 18 compared to TIS 12, which is expected because the warranty data maturation effect is most evident at later TIS values. This shows further evidence about the efficacy of the proposed CGMM.

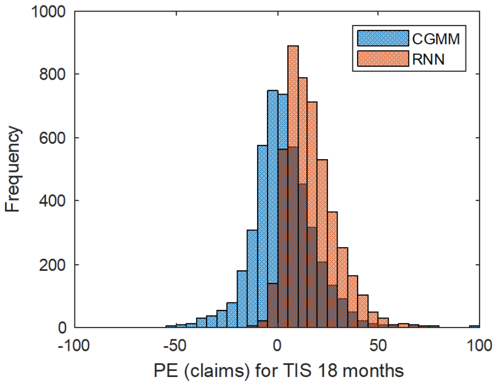


Figure 3-13 The distribution for the MAPE of the CGMM and RNN for TIS 18 months.

3.5 Conclusion

Forecasting warranty claims is challenging because of the unpredictable behavior of claims trends due to factors, such as reporting delays. Further, the inventory size is dynamic (i.e., more vehicles are produced and sold with time). This makes warranty forecasting quite profound. Therefore, there is a need to leverage warranty trends from similar historical products to mitigate the challenges mentioned above. A significant problem complicating warranty data forecasting is the data drift also referred to as the warranty data maturation phenomenon. This phenomenon results in the fact that claims trends for a nominally homogeneous population change as a function of the observation time. The reasons causing data maturation include but are not limited to: reporting delays, lot rot, warranty expiration, and seasonality.

In this work, we propose the CGMM for long-term (>12 months) warranty claims forecasting. To prove the effectiveness of the proposed CGMM, a real-life dataset consisting of more than 15,000 unique parts installed on different vehicle lines for different model years was used. The results show that the CGMM can forecast the number of claims for 18 months with a median error of 2% and a middle 80% coverage probability of +/-37%. Furthermore, it is shown to be superior to the commonly used benchmarked models like the NHPP. The CGMM encourages investigation of more ML models for warranty claims forecasting including, deep neural networks, deep non-parametric survival models, non-parametric NHPPs, and Gaussian processes.

The CGMM models the cross-correlation between different time-in-service points, enabling it to capture new trends that are not present in historical data. For example, the constant improvement in technology will carry its effect on the number of claims at all time-in-service points. Therefore, learning the effect of technology improvement on warranty claims at early time-in-service points enables predictions for the number of claims at later unobserved time-in-service points. However, if the cross-correlation between the time-in-service points for the new part of interest is significantly different from the training parts, the CGMM performance may start to degrade. For future studies, we plan to extend the CGMM to other non-normal distributions including discrete distributions. We also plan to integrate the CGMM with survival models to consider censoring based on incomplete warranty data.

Another major challenge in warranty claims forecasting for complex systems (including vehicles) is the high uncertainty at TIS points near the end-of-warranty [137]. This is expected because of the warranty expiration rush and accumulated stochasticity in the aging process of complex systems. With the advancements in internet-of-things and edge devices, many manufacturers including automakers are integrating edge devices for online sensing of the critical components of their products. For future studies, it is of significant value to utilize such online sensing data from edge devices to construct high-precision prognostics, which can then be utilized for more precise forecasting of warranty claims.

Chapter 4 BOSE: A Bayesian Architecture for Verification and Validation of Autonomous Vehicles

The advent of AI models has enabled the development of AVs [17]. AVs commonly utilize a combination of sensors, cameras, and a range of AI-driven subsystems to analyze data in real-time, making informed decisions and navigating through traffic without human intervention [18]. However, the input space for exhaustively testing complex systems such as AVs that involve the use of AI-driven ADSs is infinite. Therefore, scaling the ‘V-model’ defined under ISO 26262 to accurately identify and predict “anomalous behavior” is challenging. To overcome this issue, state-of-the-art scenario-based test approaches often limit their scope to a single Advanced Driver Assistance System (ADAS) feature. Alternatively, some approaches attempt to test an ADS using low-dimensional scenarios. Such approaches are, therefore, limited given complex multi-dimensional scenarios. To improve over these limitations, we propose, Bayesian Optimization for Scenario Exploration (BOSE). BOSE is an automated, purpose-driven, search-based framework that limits the sequential scenario-based simulations required to explore and understand complex, multi-dimensional scenarios. Furthermore, the Bayesian design structure of BOSE continues to explore every test scenario until convergence. Such a strategy combined with Latin Hypercube Sampling (LHS) helps BOSE expedite scenario exploration by efficiently learning complex interactions between scenario parameters. In our experiments with a custom Level V ADS, BOSE reduced scenario exploration effort by over 30% when compared to a state-of-the-art approach.

4.1 Introduction

National Highway Traffic Safety Administration (NHTSA) defines five (5) distinct levels of autonomy. Each incremental level of autonomy adds more driving duties to the vehicles' systems and provides additional benefits to the average driver [229], [230]. More details on levels of autonomy are shown in Table 4-1. Specifically, above Level III, limited driving and monitoring duties are given to the vehicle's driver. At this stage, the ADS must successfully achieve its driving task under every possible environment and traffic condition that falls within its Operational Design Domain (ODD) [28]. Empirically, researchers claim that a Level V ADS needs to be tested for more than 11 billion miles to have a 95% confidence that it is at least 20% safer than its human counterparts [22]. Thus, before it is deployed in the real world, any ADS' safety, robustness, and reliability must be sufficiently evaluated. In this work,

• **Safety** is defined as "the freedom from unacceptable risk of physical injury or damage to the health of vehicle occupants by the ADS."

• **Robustness** is defined as "the degree to which the ADS can function correctly in the presence of invalid inputs or stressful environmental conditions."

• **Reliability** is defined as "the probability that the ADS performs its required functions for the desired period without failure given an ODD with the desired confidence."

Table 4-1 Levels of Autonomy

Level	**Automation**	**Characteristics**
0	No Autonomy	Full control by user.
I	Driver Assistance	Full control but assisted by some features like AEB, ACC, etc.
II	Partial Automation	Assisted by automated acceleration, braking etc., but needs monitoring of environment by driver.
III	Conditional Automation	Driver is necessary but not required to monitor environment. Needs to take control in certain situations.
IV	High Automation	Driver is needed only in certain conditions.
V	Full Automation	Driver is not needed at all.

While traditional test approaches, as given under ISO 26262 [231], are still relevant to mechanical components, ensuring that an ADS possesses a certain level of safety, robustness, and reliability under the definitions of ISO 26262 is challenging. More specifically, due to the infinite-dimensional nature of ADSs, scaling the traditional 'V-model' becomes hard [85]. It is, therefore, impossible to manually identify and generate all test cases that the ADS needs to be evaluated across. The non-deterministic nature of several Deep Neural Networks (DNNs) and AI driven subsystems further complicate this problem. Existing test approaches are thus deemed insufficient for testing ADSs [28]. Recent crashes reported by the NHTSA[1] are a testament to the insufficiency of existing test approaches [89], [232].

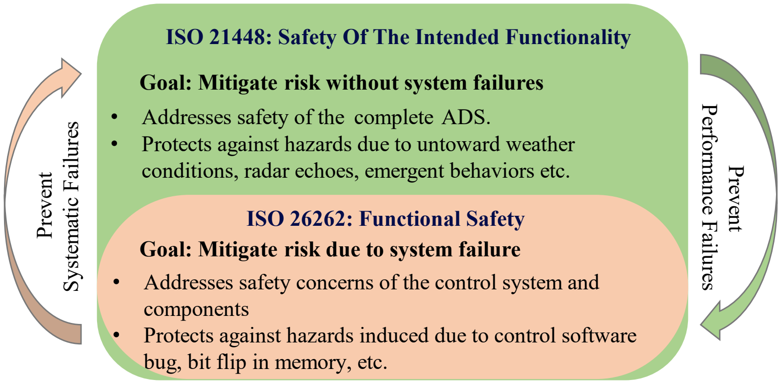
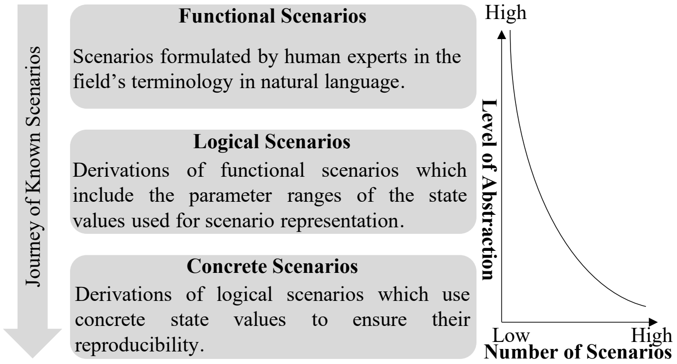

ISO 21448: Safety Of The Intended Functionality
Goal: Mitigate risk without system failures
• Addresses safety of the complete ADS.
• Protects against hazards due to untoward weather conditions, radar echoes, emergent behaviors etc.
ISO 26262: Functional Safety
Goal: Mitigate risk due to system failure
• Addresses safety concerns of the control system and components
• Protects against hazards induced due to control software bug, bit flip in memory, etc.
Prevent Systematic Failures
Prevent Performance Failures

Figure 4-1 Scope of ISO 21448 and ISO 262262 for safety of ADSs.

To address these challenges, researchers have recently proposed using scenario exploration methods under the umbrella of ISO/PAS 21448 [26], [59], [233]. Figure 4-1 provides a graphical summary of the scopes of ISO 26262 and ISO 21448 [231], [233]. Scenario exploration methods overcome the problem of generating relevant test scenarios by automatically identifying critical ones that help explore and understand the underlying functional scenario. Therefore, scenario exploration methods not only help identify complex interactions but also help identify any "anomalous behavior" henceforth referred to as "emergent behavior" within the ADS. However,

[1] https://www.nhtsa.gov/laws-regulations/standing-general-order-crash-reporting

while scenario exploration provides a rigorous testing approach, scaling it across complex, multi-dimensional functional scenarios is challenging [22], [78], [234]. To overcome this challenge, state-of-the-art scenario exploration methods often limit their scope to a single ADAS feature [78], [234]. Alternatively, some methods attempt to test the complete ADS, although using low-dimensional scenarios [22], [78]. However, such methods are often inadequate and often fail to sufficiently comprehend the interactions between the features of multi-dimensional scenarios. Figure 4-2 formalizes the definitions of functional, logical, and concrete scenarios used in this work.



Functional Scenarios

Scenarios formulated by human experts in the field's terminology in natural language.

Logical Scenarios

Derivations of functional scenarios which include the parameter ranges of the state values used for scenario representation.

Concrete Scenarios

Derivations of logical scenarios which use concrete state values to ensure their reproducibility.

Figure 4-2 Formalized scenario definitions used in this work.

To overcome these limitations and to provide a true understanding of the underlying functional scenario, we propose the Bayesian Optimization for Scenario Exploration (BOSE). BOSE is an automated, purpose-driven, search-based framework that limits the sequential scenario exploration effort required to understand a given multi-dimensional functional scenario. BOSE employs optimal Bayesian design strategies to accurately sample high-entropy concrete scenarios in the multi-dimensional design space. These concrete scenarios and their observed responses are then used to train a Bayesian DNN that develops robust priors over this multi-dimensional design space. Such a strategy combined with Laten Hypercube Sampling (LHS) in BOSE helps it identify critical scenarios of failure while developing a complete understanding of the underlying functional scenario. In our tests with a custom Level V ADS, BOSE reduced scenario simulation needs by over 30% compared to a state-of-the-art testing approach. In summary, BOSE exhibits the following advantages:

1. **Interpretability**: BOSE learns robust priors that may be leveraged for better scenario understanding across multi-dimensional test scenarios.
2. **Efficiency**: The entropy-based design of BOSE combined with LHS ensures fast convergence by only simulating those scenarios that maximize scenario understanding.
3. **Scalability**: The use of a Bayesian DNN within BOSE allows it to scale effortlessly across complex, multi-dimensional scenarios.
4. **Automation**: The simulation-based approach allows for automated test generation and scenario exploration in BOSE.

4.2 Related Work

4.2.1 Scenario Generation Approaches

Several scenario-generation approaches have been proposed in the literature. In their work, Ding *et al.* [235] categorize all scenario generation approaches into the following three categories, namely, (i) Data-driven methods, (ii) Adversarial methods, and (iii) Knowledge-based methods.

Data-driven methods for ADSs often rely on sampling scenarios from existing scenario libraries to improve exposure to rare, high-risk scenarios (e.g. scenario replay, clustering, and random generation) [236], [237]. To increase diversity, some researchers have begun using randomized generation of constrained variants of collected scenarios [237]. Additionally, other researchers have used Bayesian methods such as Importance Sampling (IS) to generate scenarios [87]. Although, these techniques are useful in generating risky scenarios, they may struggle when it comes to multi-dimensional scenarios. In recent years, deep learning models have also been used to generate scenarios as they are able to handle high-dimensional scenarios more effectively [238].

The use of Adversarial methods has gained popularity in generating risky scenarios for ADSs [239]–[243]. Adversarial methods can be classified into two types, (i) static methods, and (ii) dynamic methods. Static methods typically target sensor modules that require single-frame data as inputs, while dynamic methods target control and driving policies that need smooth, dynamic movements within the generated data [244]. Methods that manipulate the initial conditions of the adversaries to generate scenarios are particularly effective for dynamic adversarial scenario generation because of their lower computational costs [239], [242]. Many methods have been proposed that attempt to control adversarial objects, often set up as Reinforcement Learning (RL) problems [240], [241]. However, it should be noted that the methods are limited by their ability to replicate sensor data and do not cover the entirety of test spaces for dynamic systems [22].

While data-driven and adversarial methods generate scenarios with little consideration of constraints such as traffic regulations, knowledge-based methods integrate domain-knowledge to generate high-risk scenarios efficiently [245]–[250]. Knowledge-based methods can be further classified as rule-based methods and guided methods. Rule-based methods typically attempt to replicate real-world scenarios [245], [249] and have been used in ADS evaluation platforms [246], [248], [250]. However, these methods lack diversity, and researchers have traditionally augmented them using other exploration methods [251]. This approach not only encourages diversity between the generated scenarios but also encourages high interpretability of the underlying functional scenario. In a recent work, Kluck *et al.* [234] proposed a genetic algorithm-based exploration method that reduces simulation needs for scenario exploration. However, they did not test a complete ADS and limited their experiments to testing a single ADAS feature. Gangopadhyay *et al.* [22] proposed another sequential learning approach using Bayesian optimization and a Level IV ADS, but only evaluated it using a single low-dimensional scenario. Therefore, to overcome over these limitations, in this work we propose the BOSE framework.

4.2.2 Simulation Platforms

Table 4-2 Comparison of Common Simulation Platforms used for Scenario Generation

Variable **Simulator**	**Open Source**	**Realistic Perception**	**Back-end**	**Realistic Maps**	**ROS Integration**	**Scenario Scripting**	**Scripting Language**
CARLA	✓	✓	UE4	✓	✓	✓	Python / C++
CarSim	✗	✗	-	✓	✓	✓	MATLAB / Simulink
SMARTS	✓	✗	-	✗	✗	✓	Python
TORCS	✓	✓	-	✓	✗	✗	C++
L2R	✓	✓	UE4	✓	✗	✓	Python
AutoDrive	✓	✓	Unity	✗	✓	✓	Python / C++
Webots	✓	✓	ODE	✓	✓	✓	Python / C++
GTA	✗	✓	RAGE	✗	✗	✗	-
Autonovi-sim	✗	✓	PhysX	✗	✗	✓	-
LGVSL	✓	✓	Unity	✓	✓	✓	Python

In order to develop a practical framework for synthetic scenario exploration, various stakeholders have developed driving simulators. Some of them include SMARTS [247], TORCS [192], CarSim [252], L2R [253], CARLA [184], and many more [190], [191], [193], [254]–[258]. Unfortunately, simulators such as SMARTS and TORCS lack the realism and complexity needed for modern driving. Further, standard simulators such as CarSim and L2R lack the interaction needed between a vehicle, its sensors, and its environment to replicate real-world conditions. Finally, high-fidelity simulators such as GTA-V [191] and Autonovi-sim [190] are not open source, limiting the developmental resources needed for advanced research. Thus, to overcome these issues, developers along with academia have been developing simulators such as L2R [253], CARLA, LGVSL [259], Webots [255], and AutoDrive [254] that are not only open source but also offer high fidelity and customizations for realistic simulations. Especially, CARLA, with its close integration with software packages like RoadRunner [260] and VectorZero [261] to replicate real-world road maps using tools like OpenStreetMap [262], has made its use more prolific. Furthermore, its close integration with tools such as ROS [263] and custom scenario generation capabilities make it an ideal choice for scenario generation. Table 4-2 Comparison of Common Simulation Platforms used for Scenario Generation summarizes the aforementioned simulation platforms and their important caveats.

4.3 Bayesian Optimization for Scenario Exploration (BOSE)

4.3.1 The Non-sequential Case

Consider a design problem that is non-sequential. Let $\boldsymbol{\theta} \in \boldsymbol{\Theta}$ be model parameters with prior $p(\boldsymbol{\theta})$ and a likelihood model $p(y|\boldsymbol{\theta}, \boldsymbol{d})$ where, $\boldsymbol{d} \in \boldsymbol{D}$ represents a multi-dimensional concrete scenario, and $y \in Y$ is the possible outcome of the concrete scenario. In our case, $Y \in \{0,1\}$ since the outcome of every simulated concrete scenario can be either 'Crash' $(y = 1)$ or 'Normal' $(y = 0)$. Next, we assume that we are free to choose $\boldsymbol{d}$, but the subsequent outcome will be probabilistic and distributed according to,

$$p(y|\boldsymbol{d}) = \int p(y, \boldsymbol{\theta}\, |\boldsymbol{d}) d\boldsymbol{\theta} = \int p(y|\boldsymbol{\theta}, \boldsymbol{d}) p(\boldsymbol{\theta}|\boldsymbol{d}) d\boldsymbol{\theta} \tag{4-1}$$

Since $\boldsymbol{\theta}$ is independent of $\boldsymbol{d}$, we have $p(\boldsymbol{\theta}|\boldsymbol{d}) = p(\boldsymbol{\theta})$. Thus, eq. (4-1) can be rewritten as,

$$p(y|\boldsymbol{d}) = \int p(y|\boldsymbol{\theta}, \boldsymbol{d}) p(\boldsymbol{\theta}) d\boldsymbol{\theta} \tag{4-2}$$

Let $U(y, \boldsymbol{d})$ be some utility function representing the utility of choosing some concrete scenario $\boldsymbol{d}$ and observing response y. Thus, our goal is to maximize the information about $\boldsymbol{\theta}$. This can be done using the Shannon information [264] between the prior and the posterior,

$$U(y, \boldsymbol{d}) = \underbrace{\int p(\boldsymbol{\theta}|y, \boldsymbol{d}) \log\big(p(\boldsymbol{\theta}|y, \boldsymbol{d})\big)\, d\boldsymbol{\theta}}_{\text{Entropy of Posterior}} - \underbrace{\int p(\boldsymbol{\theta}) \log\big(p(\boldsymbol{\theta})\big) d\boldsymbol{\theta}}_{\text{Entropy of Prior}} \tag{4-3}$$

However, since the outcome y is still uncertain, we define the expected utility $\bar{U}(\boldsymbol{d})$ with respect to $p(y|\boldsymbol{d})$, such that,

$$\bar{U}(\boldsymbol{d}) = \int U(y, \boldsymbol{d}) p(y|\boldsymbol{d}) dy \tag{4-4}$$

$$= \int \int p(y, \boldsymbol{\theta}|\boldsymbol{d}) \log\big(p(\boldsymbol{\theta}|y, \boldsymbol{d})\big)\, d\boldsymbol{\theta} dy - \int p(\boldsymbol{\theta}) \log\big(p(\boldsymbol{\theta})\big) d\boldsymbol{\theta} \tag{4-5}$$

$$= \int \int p(y, \boldsymbol{\theta}|\boldsymbol{d}) \log\left(\frac{p(\boldsymbol{\theta}|y, \boldsymbol{d})}{p(\boldsymbol{\theta})}\right) d\boldsymbol{\theta} dy\,. \tag{4-6}$$

Since our goal is to maximize the mutual information, under the Bayesian adaptive strategy, the next best concrete scenario that can be simulated is given by,

$$\boldsymbol{d}^* = \underset{\boldsymbol{d} \in \boldsymbol{D}}{\operatorname{argmax}} \bar{U}(\boldsymbol{d}). \tag{4-7}$$

$\boldsymbol{d}^*$ can thus be interpreted as the next best concrete scenario that most reduces uncertainty over model parameters $\boldsymbol{\theta}$ over design space $\boldsymbol{D}$. However, in practice, the likelihood model approximates the ADS under test. While we would like the system to be completely deterministic, several stochastic processes are involved in the ADS' decision-making, making it almost impossible to construct a perfect distribution over $\boldsymbol{\theta}$. As a result, $p(\boldsymbol{\theta}|y, \boldsymbol{d})$ is considered intractable and is rarely known in a closed form. Thus, it becomes crucial to approximate $p(\boldsymbol{\theta}|y, \boldsymbol{d})$. Although a full-factorial design that runs an exhaustive-grid search over $\boldsymbol{D}$ will help provide an estimate of

$p(\boldsymbol{\theta}|y,\boldsymbol{d})$, it is often impractical [265]. Such a design will need infinite concrete scenario simulations when given a multi-dimensional functional scenario. Therefore, using the Bayes rule,

$$p(\boldsymbol{\theta}|y,\boldsymbol{d}) = \frac{p(\boldsymbol{\theta})p(y|\boldsymbol{\theta},\boldsymbol{d})}{p(y|\boldsymbol{d})} \tag{4-8}$$

Thus, we can rewrite eq. (4-6) such that,

$$\bar{U}(\boldsymbol{d}) \; = \int\int p(y,\boldsymbol{\theta}|\boldsymbol{d}) \log\left(\frac{p(y|\boldsymbol{\theta},\boldsymbol{d})}{p(y|\boldsymbol{d})}\right) d\boldsymbol{\theta} dy \tag{4-9}$$

$$= \int\int p(y,\boldsymbol{\theta}|\boldsymbol{d}) \log\big(p(y|\boldsymbol{\theta},\boldsymbol{d})\big)\, d\boldsymbol{\theta} dy - \int p(y|\boldsymbol{d}) \log\big(p(y|\boldsymbol{d})\big) dy \tag{4-10}$$

Equation (4-10) can now be solved using Monte-Carlo simulations under the Bernoulli approximation of $Y \in \{0,1\}$ such that,

$$\bar{U}(\boldsymbol{d}) = \hat{Z}(\boldsymbol{d}) - \hat{P}\log\big(\hat{P}\big) - \big(1-\hat{P}\big)\log\big(1-\hat{P}\big) + \epsilon \tag{4-11}$$

Where ϵ is the approximation error and,

$$\begin{aligned} \hat{Z}(\boldsymbol{d}) = \frac{1}{N}\sum_{n=1}^{N} p(y=1|\boldsymbol{\theta}_n,\boldsymbol{d}) \log\big(p(y=1|\boldsymbol{\theta}_n,\boldsymbol{d})\big) + \\ \frac{1}{N}\sum_{n=1}^{N}(1-\,p(y=1|\boldsymbol{\theta}_n,\boldsymbol{d}))\log(1-p(y=1|\boldsymbol{\theta}_n,\boldsymbol{d})) \end{aligned} \tag{4-12}$$

$$\hat{P} = \frac{1}{N}\sum_{n=1}^{N} p(y=1|\boldsymbol{\theta}_n,\boldsymbol{d}) \approx p(y=1|\boldsymbol{d}) \tag{4-13}$$

Therefore, as $N \to \infty$, $\epsilon \to 0$.

4.3.2 The Sequential Case

In practice, scenario exploration is a sequential experimental procedure, and hence $\boldsymbol{d}_t^*$ is sampled based on previous $t-1$ scenario simulations. As such, we replace $p(\boldsymbol{\theta})$ with

$p(\boldsymbol{\theta}|\boldsymbol{d}_{1:t-1}, y_{1:t-1})$ where $\boldsymbol{d}_{1:t-1}$ and $y_{1:t-1}$ are simulated concrete scenarios and their observed responses. Therefore, we can rewrite eq. (4-10) such that,

$$\bar{U}_t(\boldsymbol{d}) = \int\int p(\boldsymbol{\theta}|\boldsymbol{d}_{1:t-1}, y_{1:t-1}) p(y_t|\boldsymbol{\theta}, \boldsymbol{d}_t) \log\big(p(y_t|\boldsymbol{\theta}, \boldsymbol{d}_t)\big)\, d\boldsymbol{\theta} dy_t - \int p(y_t|y_{1:t-1}, \boldsymbol{d}_{1:t}) \log\big(p(y_t|y_{1:t-1}\boldsymbol{d}_{1:t})\big) dy_t \qquad (4\text{-}14)$$

where,

$$p(y_t|y_{1:t-1}, \boldsymbol{d}_{1:t}) = \int p(\boldsymbol{\theta}|\boldsymbol{d}_{1:t-1}, y_{1:t-1}) p(y_t|\boldsymbol{\theta}, \boldsymbol{d}_t)\, d\boldsymbol{\theta}. \qquad (4\text{-}15)$$

Note that in the sequential case, the expectations are taken with respect to $p(\boldsymbol{\theta}|\boldsymbol{d}_{1:t-1}, y_{1:t-1})$ instead of $p(\boldsymbol{\theta})$. Here, it is impossible to sample from or evaluate $p(\boldsymbol{\theta}|\boldsymbol{d}_{1:t-1}, y_{1:t-1})$. Hence, we need *Bayesian inference* to estimate it.

4.3.3 Sequential Design using Bayesian Deep Neural Networks

In Deep Neural Networks (DNNs) functions $\hat{\boldsymbol{f}}$ that are most likely to fit our experiments are defined using the weights of the DNN. Let, $\boldsymbol{\omega} = (\boldsymbol{\theta}_i)_{i=1}^{L}$ represent our sufficient statistics for a DNN with L layers. Consequently, we are interested in understanding the posterior over the weights given our observables $y_{1:t-1}$, $\boldsymbol{d}_{1:t-1}$: $p(\boldsymbol{\omega}|\boldsymbol{d}_{1:t-1}, y_{1:t-1})$. Since this is intractable for a DNN, we use the variational inference procedure to approximate it. Let $q(\boldsymbol{\theta}_i)$ be the approximate variational distribution for every layer i of the DNN such that,

$$\boldsymbol{\theta}_i = \boldsymbol{M}_i \mathrm{diag}\left(\left[z_{i,j}\right]_{j=1}^{K_i}\right) \qquad (4\text{-}16)$$

$$z_{i,j} \sim Bernoulli(p_i)\ \forall\ i = 1, \dots, L \text{ and } j = 1, \dots, K_{i-1} \qquad (4\text{-}17)$$

Here, $z_{i,j}$ are Bernoulli distributed random variables with some probabilities p_i, and $\boldsymbol{M}_i$ are variational parameters to be optimized. The diag(·) operator maps vectors to diagonal matrices. The approximate predictive distribution $q(\boldsymbol{\theta}_i)$ can thus be calculated by minimizing the Kullback-

Liebler divergence, $\mathrm{KL}\big(q(\boldsymbol{\omega})\ ||\ p(\boldsymbol{\omega}|\boldsymbol{d}_{1:t-1}, y_{1:t-1})\big)$. This is equivalent to maximizing the log evidence *lower bound* and is given by,

$$\hat{\mathcal{L}}_{VI} := \sum_{i=1}^{N} \mathrm{E}\left(y_i, \hat{f}(d_i, \widehat{\boldsymbol{\omega}}_i)\right) - \mathrm{KL}(q(\boldsymbol{\omega})||p(\boldsymbol{\omega})) \tag{4-18}$$

where, $\widehat{\boldsymbol{\omega}}_i \sim q(\boldsymbol{\omega})$ and $E(\cdot,\cdot)$ is expected value of the binary classification loss. Note that sampling from $q(\boldsymbol{\theta}_i)$ is identical to performing dropout on layer i in a DNN whose weights are $(\boldsymbol{M}_i)_{i=1}^{L}$. The binary variable $z_{i,j} = 0$ corresponds to unit j in layer $i - 1$ being dropped out as an input to the i^{th} layer. The effective DNN with dropout thus forms a Bayesian DNN resulting in the same model parameters that best explain the previous $t - 1$ experiments. More details about Bayesian DNNs can be seen in [90], [266], [267].

Since our goal is to maximize the mutual information, using Bayesian inference, the next best concrete scenario that can be simulated is given by,

$$\boldsymbol{d}_t^* = \underset{\boldsymbol{d}\in\boldsymbol{D}}{\operatorname{argmax}}\, \overline{U}_t(\boldsymbol{d}) \tag{4-19}$$

We now see that the estimation procedure in Section 4.3.2 only needs samples of $\widehat{\boldsymbol{\omega}}_n$ as input. We can therefore estimate $\overline{U}_t(\boldsymbol{d})$ such that,

$$\begin{gathered}\overline{U}_t(\boldsymbol{d}) = \\ \hat{Z}_t(\boldsymbol{d}) - \hat{P}_t \log(\hat{P}_t) - (1 - \hat{P}_t)\log(1 - \hat{P}_t) + \epsilon\end{gathered} \tag{4-20}$$

where,

$$\hat{P}_t = p(y_t = 1|y_{1:t-1}, \boldsymbol{d}_{1:t}) \approx \int p(y_t = 1|\boldsymbol{\omega}, \boldsymbol{d})\, q(\boldsymbol{\omega}) d\boldsymbol{\omega} \tag{4-21}$$

$$\hat{P}_t \approx \frac{1}{N}\sum_{n=1}^{N} p(y_t = 1|\widehat{\boldsymbol{\omega}}_n, \boldsymbol{d}) \tag{4-22}$$

and,

$$\widehat{Z}_t(\boldsymbol{d}) = \frac{1}{N}\sum_{n=1}^{N} p(y_t|\widehat{\boldsymbol{\omega}}_n, \boldsymbol{d}) \log\big(p(y_t|\widehat{\boldsymbol{\omega}}_n, \boldsymbol{d})\big) + \big(1 - p(y_t|\widehat{\boldsymbol{\omega}}_n, \boldsymbol{d})\big) \log\big(1 - p(y_t|\widehat{\boldsymbol{\omega}}_n, \boldsymbol{d})\big) \tag{4-23}$$

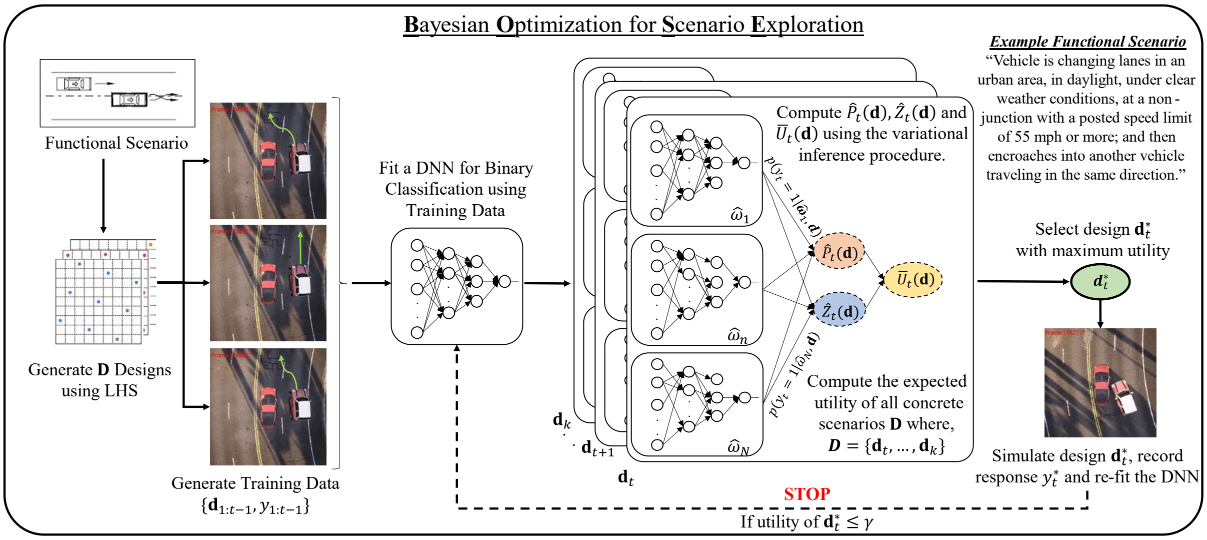

Bayesian Optimization for Scenario Exploration
Functional Scenario
Generate D Designs using LHS
Generate Training Data $\{\mathbf{d}_{1:t-1}, y_{1:t-1}\}$
Fit a DNN for Binary Classification using Training Data
Compute $\hat{P}_t(\mathbf{d}), \hat{Z}_t(\mathbf{d})$ and $\bar{U}_t(\mathbf{d})$ using the variational inference procedure.
$p(y_t = 1|\hat{\omega}_1, \mathbf{d})$
$p(y_t = 1|\hat{\omega}_N, \mathbf{d})$
$\hat{\omega}_1$
$\hat{\omega}_n$
$\hat{\omega}_N$
$\hat{P}_t(\mathbf{d})$
$\hat{Z}_t(\mathbf{d})$
$\bar{U}_t(\mathbf{d})$
Compute the expected utility of all concrete scenarios D where, $D = \{\mathbf{d}_t, \ldots, \mathbf{d}_k\}$
$\mathbf{d}_k$
$\mathbf{d}_{t+1}$
$\mathbf{d}_t$
Example Functional Scenario
"Vehicle is changing lanes in an urban area, in daylight, under clear weather conditions, at a non-junction with a posted speed limit of 55 mph or more; and then encroaches into another vehicle traveling in the same direction."
Select design $\mathbf{d}_t^*$ with maximum utility
$\mathbf{d}_t^*$
Simulate design $\mathbf{d}_t^*$, record response y_t^* and re-fit the DNN
STOP
If utility of $\mathbf{d}_t^* \leq \gamma$

Figure 4-3 Overview of the proposed BOSE framework in autonomous vehicles.

Table 4-3 Algorithm I: BOSE

```
Require: Record initial $t-1$ observations for a given functional scenario; Maximum evaluations $T^U$; $k$ concrete scenarios using LHS; posterior samples $\{\widehat{\boldsymbol{\omega}}_n\}_{n=1}^{N}$; stopping criteria $\gamma = 0.1$.
Generate: Concrete scenarios using LHS such that
$\boldsymbol{D} = \{\boldsymbol{d}_1, \ldots, \boldsymbol{d}_t, \ldots, \boldsymbol{d}_k\}$.
Simulate: Initial $t-1$ concrete scenarios to generate training data $\{\boldsymbol{d}_{1:t-1}, y_{1:t-1}\}$.
while $t \leq T^U$ do
    Fit a Bayesian DNN for binary classification using $\{\boldsymbol{d}_{1:t-1}, y_{1:t-1}\}$.
    Set $\boldsymbol{D} = \{\boldsymbol{d}_t, \ldots, \boldsymbol{d}_k\}$.
    Compute expected utility $\bar{U}_t(\boldsymbol{d})$ of all concrete scenarios $\boldsymbol{D}$.
    Select concrete scenario $\boldsymbol{d}_t^*$ with maximum utility using eq. (4-19).
    if the utility of $\boldsymbol{d}_t^* \leq \gamma$ then
        Break
    end if
    Simulate concrete scenario $\boldsymbol{d}_t^*$ and record response $y_t^*$.
    $t = t + 1$
end while
```

Table 4-3 effectively summarizes the overall working of the proposed BOSE framework using a real-life ADS and a NHTSA-defined functional scenario. A concrete scenario is only tagged as a 'Crash' ($y = 1$) if the ego vehicle loses control (crosses lane lines), experiences lateral acceleration above 0.3g, or makes contact with any other actor within the simulation. Since our goal is to maximize the mutual information, we follow the sequential design strategy given in

Section 4.3.3 and find the next best concrete scenario $\boldsymbol{d}_t^*$ with utility $\overline{U}_t(\boldsymbol{d}^*)$. In our evaluations with BOSE, although several architectures for Bayesian DNN were tried, we only provide details for the best-performing architecture. We obtained the best results when the Bayesian DNN is designed using two layers with four and one neuron(s) each. The Bayesian DNN is then trained for 20 epochs using the ADAM optimizer with a dropout rate of 50% per layer while trying to minimize the binary classification loss at every iteration of BOSE. We set $N = 500$ in all our experiments. The initial training data $\{\boldsymbol{d}_{1:t-1}, y_{1:t-1}\}$ is generated by sequentially simulating and observing $t - 1$ concrete scenarios in an autonomous driving simulator, CARLA [184]. As can be seen in Algorithm I, γ when set to lower values, provides a better understanding of the underlying functional scenario at the expense of simulation resources. In our experiments, γ, when set to 0.1, achieved the best trade-off by accurately predicting outcomes given a concrete scenario $\boldsymbol{d}$.

4.4 Case Study Using A Custom ADS

In this section, we evaluate and benchmark the proposed algorithm using a custom Level V ADS and two complex, multi-dimensional functional scenarios. In this work, we evaluate all methods by defining several concrete scenarios from a given functional scenario using the ontological process defined by Menzel *et al.* [245]. More specifically, a selected functional scenario is decomposed into p critical design parameters, such that $\boldsymbol{d} \in \mathbb{R}^p$. The defined concrete scenarios $\boldsymbol{d}$ are then simulated in CARLA and the response y is recorded.

4.4.1 Low-Dimensional Scenario

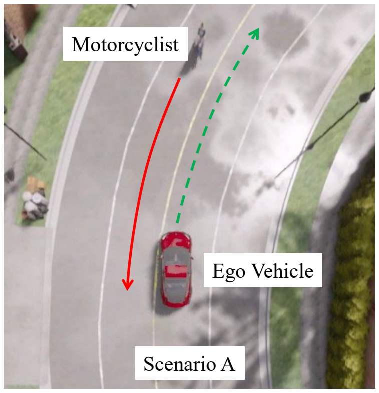


Figure 4-4 Underlying scenario typology for Scenario A. The 'red' solid line shows the expected path of the motorcyclist while the dotted 'green' line shows the expected path followed by ego vehicle under 'Normal' circumstances.

Functional Scenario Description: Ego vehicle attempts to negotiate a curve in an urban area, in daylight, under wet weather conditions, and then loses control leading to a crash with an oncoming motorcyclist. We refer to this scenario as Scenario A in this work and is explained visually using Figure 4-4.

Table 4-4 Parameter Ranges for Scenario A

Parameter	Range	Unit	Characteristic
Max ego vehicle speed	[30-40]	KPH	Influences vehicle control.
ADS proximity threshold	[8-30]	Meters	Influences sensor outputs
ADS Minimum Time-to-Collision (ADS minTTC)	[2-6]	Seconds	Influences braking behavior
ADS safe braking distance	[2-5]	Meters	

Each concrete scenario for functional Scenario A is defined using four critical parameters in our experiments. These are given in Table 4-4. These parameters and their ranges are of high interest to us and are extracted from naturalistic driving studies such as [195], [200], [202], [212]. While the 'Max Ego Speed' variable controls the maximum speed the ego vehicle can achieve, a combination of the remaining three variables controls the ego vehicle policy. Therefore, a combination of the four given variables sufficiently characterize the behavior of the ego vehicle given functional Scenario A. Furthermore, to avoid confounding effects, the motorcyclist and the ego vehicle always start from the same location on the map and the motorcyclist follows the same speed profile when handling the curve in all our simulations. Finally, based on our computing budget, we set $k = 165$ and $t = 50$ (~30% of the design space $\boldsymbol{D}$) for experiments with Scenario A. The goal, therefore, is to completely explore the given scenario by identifying a decision boundary that can sufficiently separate 'Crashes' ($y = 1$) from 'Normal' ($y = 0$) scenarios, given a scenario space $\boldsymbol{D}$.

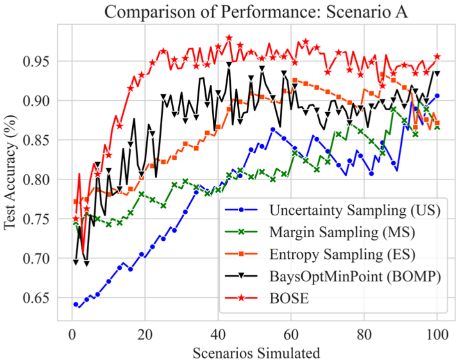


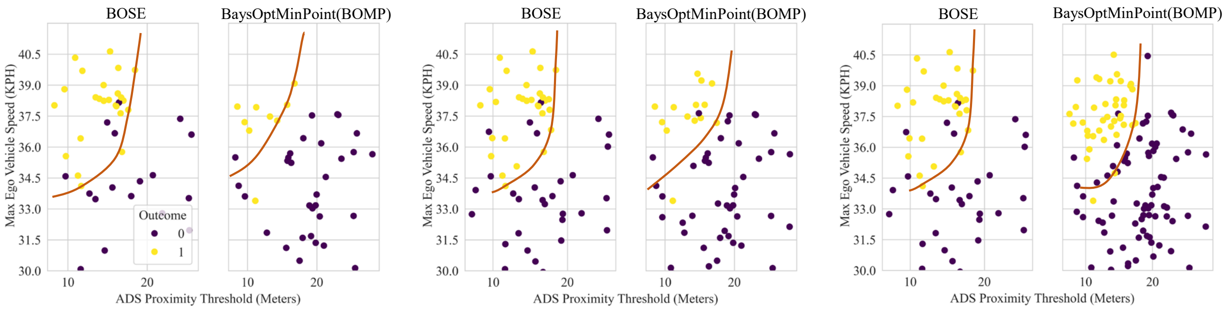
Figure 4-5 Performance comparison of BOSE with other benchmarked methods for functional Scenario A.

In our experiments, we evaluate and benchmark the proposed method across other well-known active learning methods such as Uncertainty Sampling (US), Margin Sampling (MS), and Entropy Sampling (ES) [268]. All of these methods are implemented using their default values in the modAL toolbox [269] and use the same DNN architecture and training strategy as given in Section 4.3. Next, we create a test set of 100 random concrete scenarios for Scenario A and evaluate the mentioned traditional active learning methods along with the BaysOptMinPoint (BOMP) method [22]. Sufficient care is taken to exclude these 100 test scenarios from those given in $\boldsymbol{D}$. For a fair comparison, we set a budget of 100 concrete scenario simulations with $T^U = 100$. In our experiments with the test set, BOSE is shown to clearly outperform the other benchmark methods. As can be seen in Table 4-5, although the traditional active learning methods perform fairly well, BOMP and BOSE clearly outperform them reaching test accuracies over 90% within the first 50 concrete scenario simulations. This is primarily due to the advantages of the Bayesian strategies leveraged by these methods.

Table 4-5 Comparison of Performance for Scenario A

Model	Maximum Test Accuracy (%)	Scenarios Simulated to reach 90% Test Accuracy	Time per Iteration (Secs)	Total Run Time (Secs)
Uncertainty Sampling (US)	90.8	100	**0.3**	1530.0
Margin Sampling (MS)	90.1	96	**0.3**	1468.8
Entropy Sampling (ES)	93.6	45	**0.3**	688.5
BaysOptMinPoint (BOMP)	94.8	23	0.6	372.6
BOSE	**97.1**	**16**	1.6	**265.6**

While any scenario exploration method needs to be accurate, it is our high interest to analyze whether it is scalable and robust. Therefore, we compare and report the average performance of all tested methods averaged over five runs for functional Scenario A in Table 4-5. As can be seen in this table and figure, BOSE only needs about 16 concrete scenario simulations to achieve a test accuracy of 90% thus reducing the need for simulations by over 30% over the closest benchmarked method. While BOSE is marginally slower than BOMP on an iteration basis, it is about 107 secs faster than the BaysOptMinPoint method when total run time is considered. This is primarily due to the overhead incurred in running additional simulations since the simulator needs about 15 secs to simulate every concrete scenario for Scenario A.



(a) (b) (c)

Figure 4-6 Progression of decision boundaries after, (a) 40 simulations, (b) 60 simulations, and (c) after convergence for BOSE and the BaysOptMinPoint method. Here, ‘yellow’ dots represent ‘Crash’ scenarios where y=1, and purple dots represent ‘Normal’ scenario.

To better understand the decision-making process of BOSE given scenario A, we re-run the top two methods, i.e., the BOMP method and BOSE with $T^U = 500$ and only stop when convergence is achieved where $\bar{U}_t(\boldsymbol{d}^*) \leq \gamma$. Here, we find a high correlation between the outcomes of the simulated concrete scenarios and two design parameters, namely, (i) Max Ego Vehicle Speed and (ii) ADS Proximity Threshold. As can be seen in Figure 4-6, due to its ability to define robust priors, BOSE is quick to identify an interaction between these two variables and establish a decision boundary. As a result, while it takes 60 simulations for BOSE to converge, it takes about 116 simulations for BOMP to converge. Here, in contrast to BOSE, BOMP starts from random initial states and spends much of its initial effort in understanding the scenario space. The LHS-driven initializations within BOSE help it overcome this drawback by performing better exploratory experiments during its initial phase. As can be seen in Figure 4-8, while BOSE performs well-targeted experiments that help it establish a clear decision boundary, much of the initial effort by the benchmarked method is spent in randomly exploring the space. Often, scenarios are more complicated and involve two or more participants. Therefore, in the following section, we evaluate the proposed method on a complex multi-dimensional scenario characterized by several design parameters and multiple actors.

4.4.2 High-Dimensional Scenario

Functional Scenario Description: Ego vehicle attempts to negotiate a roundabout in an urban area under daylight and then crashes with one of the two vehicles violating a roundabout yield. We refer to this scenario as Scenario B in this work and is explained visually using Figure 4-7.

In our experiments, Scenario B can be decomposed into several concrete scenarios using the twelve parameters given in Table 4-6. The parameters and their ranges are of high interest to us and are extracted from naturalistic driving studies such as [195], [200], [202], [212]. While the

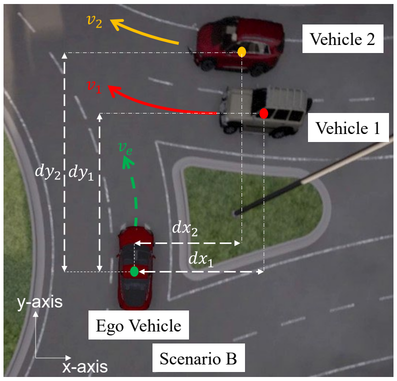


Figure 4-7 Underlying scenario typology for Scenario B. The 'red' solid line shows the expected path of both vehicle violating roundabout yield, while the dotted 'green' line shows the expected path followed by ego vehicle under 'Normal' circumstances.

Table 4-6 Parameter Ranges for Scenario B

Parameter	Range	Unit	Characteristic
Ego Vehicle			
Max vehicle velocity (v_e)	[20-50]	KPH	Influences vehicle control.
ADS proximity threshold	[8-30]	Meters	Influences sensor outputs
ADS Minimum Time-to-Collision (ADS minTTC)	[2-6]	Seconds	Influences braking behavior
ADS safe braking distance (b_e)	[2-5]	Meters	
Vehicle 1			
Max Vehicle velocity (v_1)	[14-42]	KPH	Influences the probability and the severity of crash near the round about
Initial distance of vehicle from ego vehicle in y-direction (dy_1)	[24-32]	Meters	
Initial distance of vehicle from ego vehicle in y-direction (dx_1)	[21-38]		
ADS safe braking distance (b_1)	[2-5]	Meters	Influences braking behavior
Vehicle 2			
Max Vehicle velocity (v_2)	[25-40]	KPH	Influences the probability and the severity of crash near the round about
Initial distance of vehicle from ego vehicle in y-direction (dy_2)	[32-50]	Meters	
Initial distance of vehicle from ego vehicle in y-direction (dx_2)	[26-38]	Meters	
ADS safe braking distance (b_2)	[2-5]	Meters	Influences braking behavior

ego vehicle parameters influence the behavior and response of the ego vehicle, a complex combination of the remaining eight parameters influence the behavior of the two adversary vehicles, namely, 'Vehicle 1' and 'Vehicle 2.' Variables such as $dx_1, dy_1, dx_2,$ and dy_2 can then be used to change the locations of 'Vehicle 1' and 'Vehicle 2', respectively. Note that 'Vehicle 1' and 'Vehicle 2' will always attempt to violate the roundabout yield but may perform precautionary braking. Thus, a scenario is tagged as a 'Crash' or 'Normal' using the same criteria in Section 4.3.

Finally, based on our computing budget, we set $k = 350$ and $t = 105$ (~30% of the design space $\boldsymbol{D}$) for experiments with Scenario B.

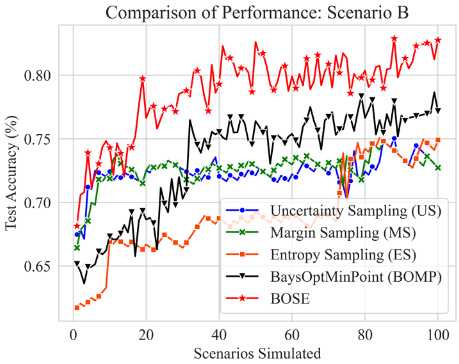


Figure 4-8 Performance comparison of BOSE with other benchmarked methods for Scenario B.

In our experiments, we benchmark the proposed method across other well-known active learning methods using a test set of 100 random concrete scenarios for Scenario B. Sufficient care is taken to exclude these 100 test scenarios from those given in $\boldsymbol{D}$. In our experiments with this test set, BOSE clearly outperforms the other benchmark methods. As can be seen in Figure 4-8, the high dimensionality of the underlying functional scenario overwhelms the traditional active learning methods. They, thus, barely cross the 75% test accuracy barrier when given a budget of 100 concrete scenario simulations. On the other hand, the BOMP method performs slightly better reaching a maximum accuracy of 78% but only after 79 concrete scenario simulations. While no method is able to cross the 90% test accuracy barrier, BOSE clearly outperforms all benchmarked methods achieving a maximum test accuracy of 83.2%. As can be seen in Table 4-7 and Figure 4-8, BOSE only needs 19 concrete scenario simulations to achieve a test accuracy of 75% thus reducing the need for simulations by over 43% over the closest benchmarked method. However, in comparison to Scenario A, the improvement in test accuracy is much slower for BOSE and all benchmark methods. This may be primarily due to the complex interactions that exist in Scenario B. While BOSE is slower on an iteration basis, it is about 243 secs faster than the BOMP method when total run time is considered (averaged over 5 runs). This is primarily due to the overhead incurred in running additional simulations since the simulator needs about 20 secs to simulate every concrete scenario for Scenario B.

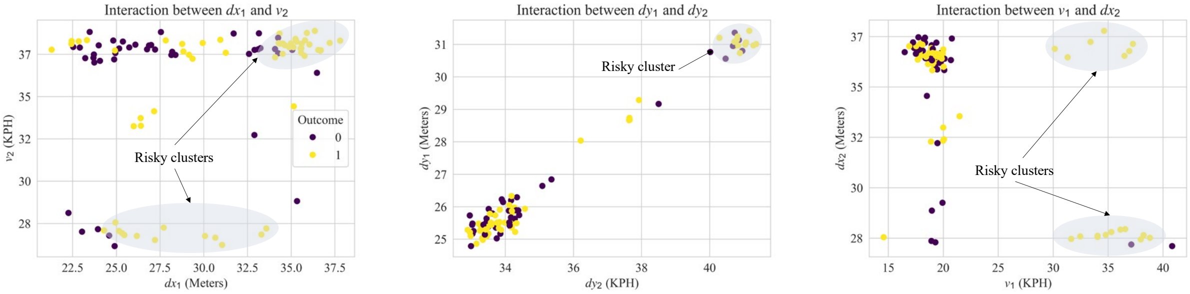
Table 4-7 Comparison of Performance for Scenario B

Model	Maximum Test Accuracy (%)	Scenarios Simulated to reach 75% Test Accuracy	Time per Iteration (Secs)	Total Run Time (Secs)
Uncertainty Sampling (US)	75.1	86	**20.7**	1780.2
Margin Sampling (MS)	75.0	84	**20.7**	1738.8
Entropy Sampling (ES)	75.3	84	**20.7**	1738.8
BaysOptMinPoint (BOMP)	78.6	32	**21.2**	678.4
BOSE	**83.2**	**19**	22.9	**435.1**

To better understand the decision-making process of BOSE given scenario B, we re-run the top two methods, i.e., the BOMP method and BOSE with $T^U = 500$ and only stop when convergence is achieved where $\bar{U}_t(\boldsymbol{d}^*) \leq \gamma$. While BOSE can achieve convergence after 101 scenarios, BOMP requires over 200 scenarios. While the LHS-driven initializations help in improving the exploratory performance, the Bayesian DNN within BOSE helps in quickly identifying the multi-dimensional clusters in the underlying scenario space. It is thus able to develop a robust prior over the design space $\boldsymbol{D}$. This helps BOSE quickly explore and search the scenario space.

As mentioned earlier, a complex interaction exists between the twelve design variables mentioned in Table 4-6. Thus, due to the complexity of the underlying scenario, it may be challenging to visualize a clean decision boundary in two dimensions. Figure 4-9 can be used to visualize some of these complex interactions identified by BOSE that exist in the underlying scenario space. Therefore, we have a sufficient belief to hypothesize that a decision boundary in higher dimensions may exist that the scenario exploration methods should be able to identify.

Figure 4-9 can be used to visualize some examples of concrete scenarios, their predicted, and observed outcomes.



(a) (b) (c)

Figure 4-9 Risky clusters with high probability of crashes identified by BOSE in (a) dx_1 and v_2, (b) v_1 and dx_2, and (c) dy_1 and dy_2. Here, 'yellow' dots represent 'crash' scenarios where y=1, and purple dots represent 'normal' scenarios where y=0.

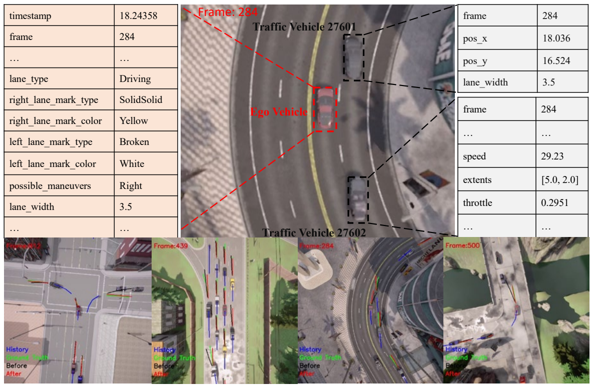
4.5 Conclusion

Traditional test approaches are often deemed insufficient when testing modern systems that involve a myriad of AI-driven subsystems. One example of such systems are the ADSs in AVs. Therefore, modeling and predicting anomalous behavior becomes challenging. To overcome this challenge, traditional methods use scenario-based tests. However, given the infinite-dimensional nature of the ADS, scaling the traditional 'V-model' to identify and predict emergent behavior is challenging. Therefore, to improve over this limitation, in this work, we propose the Bayesian Optimization for Scenario Exploration (BOSE) framework. BOSE is an automated, purpose-driven, search-based framework that uses Bayesian theories and LHS-driven initializations to explore the scenario space efficiently and identify complex interactions in an ADS. In tests with a custom Level V ADS, BOSE performed fast and accurate scenario exploration, reducing simulation needs by more than 30% over state-of-the-art methods. For future work, we plan to evaluate this work using more complex scenarios while attempting to improve its inference speed.

Chapter 5 Collaborative Modeling using Infrastructure Information for Improved Predictive Analytics in Autonomous Vehicles

V2V and V2I communications are vital components of AVs. Such technologies enable these vehicles to communicate with each other (V2V) and with infrastructure (V2I), creating a network of interconnected devices that can provide valuable information of the driving environment. Advancements in V2I and V2V technologies offer a path to limiting the uncertainties in an AV's systems. However, these systems do not use AI models that leverage such information since publicly available datasets do not include explicit annotations for infrastructure and pooled vehicle data. To overcome these challenges, in this work, we first introduce VTrackIt, the first comprehensive synthetic dataset, which explicitly includes annotations for intelligent infrastructure and pooled vehicle information. Next, to demonstrate the effectiveness of collaborative modeling, we introduce InfraGAN. InfraGAN is a novel deep learning model that uses the information provided in the VTrackIt dataset for accurate trajectory predictions. Our experiments with InfraGAN showcase the power of VTrackIt and its data. With VTrackIt and InfraGAN, this work lays the foundation for the development of informed applications that support higher levels of connected, safer, and reliable AVs.

5.1 Introduction



timestamp	18.24358
frame	284
…	…
lane_type	Driving
right_lane_mark_type	SolidSolid
right_lane_mark_color	Yellow
left_lane_mark_type	Broken
left_lane_mark_color	White
possible_maneuvers	Right
lane_width	3.5
…	…

frame	284
pos_x	18.036
pos_y	16.524
lane_width	3.5

frame	284
…	…
speed	29.23
extents	[5.0, 2.0]
throttle	0.2951
…	…

Figure 5-1 An overview of the VTrackIt dataset with infrastructure and pooled vehicle information. Observed trajectories are labeled as ‘History’ and future trajectories are labeled as ‘Ground Truth’. Predicted trajectories using state-of-the-art benchmark (without infrastructure and pooled vehicle information) are labeled as ‘Before.’ Predicted trajectories using infrastructure and pooled vehicle information in VTrackIt are labeled as ‘After.’

Advancements in electronic and communication technologies will soon help realize near real-time Vehicle-to-Infrastructure (V2I) and Vehicle-to-Vehicle (V2V) data transfer [165]. These technologies provide an unprecedented opportunity to limit the uncertainties in an AV’s systems by relaying accurate and reliable data, even in adverse situations [30], [166]. Examples of V2V data include speed, heading, and pedal positions. Examples of V2I data include red light status, speed limits, and lane data [167]. Vehicle-to-Everything (V2X) encompasses both V2V and V2I data [32], [33], [168]. Therefore, it is evident that V2X data will be necessary for higher levels of safe, resilient, robust, and reliable autonomous driving. Multiple research efforts have explored the potential of infrastructure and pooled vehicle information to develop individual AV applications [32], [33], [168]–[171]. A detailed review by Jeong *et al.* [172] summarizes many of these research efforts. Unfortunately, the broad availability of public datasets with explicit annotations for infrastructure and pooled vehicle information is limited to large-scale industrial efforts in the form of proprietary collected data [69]. This limits the progress within the self-driving community to develop reliable models for AV applications that support higher levels of autonomous driving. To address this research challenge, we introduce a synthetically generated self-driving dataset named VTrackIt [89]. The name VTrackIt reflects tracking infrastructure and surrounding vehicle information in real-time. VTrackIt aims to attract the attention of the larger self-driving community to integrate infrastructure and pooled vehicle information in their AV applications for safer and more reliable autonomous driving. Moreover, VTrackIt also contains out-of-distribution data to expose the developed models to rare, high-risk edge cases. The VTrackIt[2] dataset is publicly available for use upon request [89].

VTrackIt is inspired by existing datasets like Level5 [69], NuScenes [173], Argoverse [174], ApolloScape [175], Waymo [176], DAIR-V2X [177], AIODrive [178], and V2X-Sim [31]. However, Argoverse, NuScenes, ApolloScape, AIODrive, V2X-Sim, and DAIR-V2X do not provide explicit tags for relevant infrastructure information like lane line annotations, lane widths, speed limits, stop signs, and many more. In contrast, it is assumed that such information can be sensed using perception sensors in an AV. However, extracting such information from collected perception data is complemented by severe uncertainties due to the non-deterministic nature of DL models involved [179]. One dataset that seems to record relevant infrastructure information is the Level5 dataset; however, it fails to provide data from a pool of surrounding vehicles like speed, lateral and longitudinal accelerations, and pedal positions, among others. It is worth noting that although some datasets offer semantic maps, they are biased to scenarios within their given Operational Design Domain (ODD) [69], [174], [176]. Furthermore, they lack out-of-distribution data and thus introduce an inherent training bias in the developed DL models [178]. VTrackIt provides a more comprehensive vision of integrating infrastructure and pooled vehicle information. In our work, we validate these advantages of VTrackIt by developing a Generative Adversarial Network (GAN) called InfraGAN for trajectory predictions. Our experiments with

[2] https://vtrackit.irda.club/

InfraGAN show that accuracy and high-risk edge cases can be reduced by training DL models using data from the VTrackIt dataset. The contributions of this work are multifold:

1. A large-scale, synthetic, self-driving dataset with infrastructure and pooled vehicle information.
2. Inclusion of high-risk out-of-distribution data, that complements and expands beyond real-world datasets.
3. A video database with high-resolution Birds-Eye-View (BEV) and 360° views of the ego vehicle for better scenario understanding and visualizations.
4. The first GAN (InfraGAN) that integrates infrastructure and pooled vehicle information for trajectory predictions.

We acknowledge that synthetically generated datasets will not replace real-world datasets. We also acknowledge that presently it is hard to integrate or access such information in real-time. Therefore, VTrackIt and other similar synthetic datasets should be used to explore the benefits that infrastructure and pooled vehicle information may provide before being integrated into real-world AV applications.

5.2 Existing Datasets and Related Work

5.2.1 Real-world Driving Datasets

NGSIM [180] is one of the earliest naturalistic self-driving datasets used to develop several Machine Learning (ML) solutions. This dataset provides limited infrastructure information such as lane ids, lane section id, and lane direction. Provided surrounding vehicle information includes vehicle extents, class, speed, and accelerations. This data is captured at 10 Frames-Per-Second (FPS). However, NGSIM is limited by rigid location constraints using a small section on the southbound US-101 and Lankershim Boulevard in Los Angeles, CA, eastbound I-80 in

Emeryville, CA, and Peachtree Street in Atlanta, Georgia. Huang *et al.* [175] proposed the ApolloScape dataset that logs over 2 hours of driving, including 103 scenarios with complex vehicle and pedestrian traffic flows captured at 2 FPS. However, ApolloScape does not include annotations for relevant infrastructure information. Chang *et al.* [174] proposed Argoverse, the first large-scale self-driving dataset with detailed semantic maps for the cities of Pittsburgh and Miami captured at 10 FPS. Although the Argoverse dataset logs about 320 hours of driving across 324k scenarios using a semantic map encoded with lane centers, it fails to explicitly annotate crucial infrastructure information such as lane line type, color, among others. Instead, it is assumed that the provided semantic maps implicitly carry such information. The Argoverse dataset is supplemented by rasterized maps identifying drivable areas and corresponding ground height information. In a major limitation, this dataset represents traffic vehicles using centroids, and no other information is provided apart from their locations. The Argoverse dataset also fails to offer detailed surrounding vehicle information. Houston *et al.* [69] proposed the Level5 dataset that logged 1,118 hours of driving across 170k scenes with some surrounding vehicle information captured at 10 FPS. This dataset provides semantic maps with limited infrastructure information. A significant limitation of this dataset is its bias toward a single heavily trafficked route under rigid location constraints. Holger *et al.* [173] released the NuScenes dataset that logged over 15 hours of driving across 1000 scenarios captured at 2 FPS. The NuScenes dataset includes annotations for vehicle category, locations, extents, and yaw for a pool of surrounding vehicles. However, only limited infrastructure information is provided. Another heavily used dataset for developing DL solutions for AVs is the KITTI dataset [181] which logged 22 minutes of driving captured at 10 FPS. Osinski *et al.* [182] proposed the OpenDD dataset with 501 scenarios recorded at 30 FPS; however, again, with limited infrastructure information. IntentNet [183] is a dataset used to develop DL solutions for AV applications but is not publicly available. More recently, datasets such as the DAIR-V2X [177] have also been released. While such datasets support cooperative autonomous driving using V2I and V2V information, they lack explicit annotation for several important V2I and V2V features. In summary, although important, real-world datasets need to be complemented by synthetic datasets to include out-of-distribution data that maybe hard to collect due safety or regulatory reasons.

5.2.2 Synthetic Driving Datasets

To expose the DL algorithms in AV applications to such risky scenarios, some work has also been done to collect synthetic data using state-of-art simulators such as CARLA [184]. For example, Weng *et al.* [178] released the AIODrive dataset. Although this dataset includes data from different synchronized sensors, it lacks infrastructure information, lacks diversity in road conditions. Xu *et al.* [185] introduced the OPV2V dataset with 73 interesting scenarios; however, this dataset does not investigate use of infrastructure information. KITTI-CARLA [186] is another publicly available synthetic dataset that records information identical to the KITTI dataset over seven simulated scenarios at 10 FPS. Some other synthetic datasets such as V2X-Sim [187], CODD [188], and Copper [189] have also been made publicly available using simulation-based environments in recent years. However, most of these datasets are purpose-driven and only focus on LiDAR and/or perception-driven tasks. Moreover, they also lack explicit annotations for surrounding vehicle and infrastructure information. Table 5-1 presents an overview of all significant datasets used to develop several modern AV applications. While multiple efforts collect limited information from surrounding vehicles, there is a clear gap of annotating and leveraging infrastructure information for most of the discussed datasets. Next, we introduce the VTrackIt dataset.

Table 5-1 Comparison of Major Driving Datasets for Autonomous Vehicle Applications

Variable **Dataset**	**Scenarios**	**Maps**	**Synthetic**	**Frequency (FPS)**	**BEV**	**V2V Info. Tags**	**V2I Info. Tags**	**Diverse Road Conditions**	**Low Light Conditions**
Level5	170k; 25s each	Single route in Palo Alto, CA	-	10	✓ (Aerial + Semantic)	✓	✓ (Partial)	-	-
NuScenes	1000; 20s each	Boston, Singapore	-	2	✓ (Semantic)	✓	-	✓	✓
Argoverse	324k; 6s each	Pittsburgh, Pa /Miami, FL	-	10	✓ (Semantic)	-	✓ (Partial)	✓	✓
ApolloScape	103; 60s each	China	-	2	-	✓	-	-	✓
DAIR-V2X	350; 60s each	City, highway and Intersections typologies	-	20	-	✓ (Partial)	✓ (Partial)	✓	-
AIODrive	100; 100s each	8 Virtual maps with varying typologies	✓	10	-	✓	-	✓	✓
NGSIM	-	Sections of US-101, and I-80 highways	-	10	-	✓	✓ (Partial)	-	-
KITTI	50	Karlsruhe	-	10	✓	✓	-	-	-
OpenDD	501; 5-15 min each	7 Roundabouts	-	30	✓ (Aerial + Semantic)	✓	-	-	✓
V2X-Sim	100; 20s each	3 virtual maps with varying typologies	✓	20	✓ (Aerial + Semantic)	✓ (Partial)	✓ (Partial)	-	-
VTrackIt (Proposed)	600; 30s each	6 virtual maps with varying typologies	✓	20	✓ (Aerial)	✓	✓	✓	✓

5.3 The VTrackIt Dataset

The VTrackIt dataset consists of 600 scenarios (360 training, 120 validation, and 120 test), each recorded for a maximum of 30 seconds (the duration of some out-of-distribution scenarios where the ego vehicle had a crash may be less than 30 seconds.) Uniquely, VTrackIt tracks and provides annotations for infrastructure information. Further, it also provides annotations for information collected from a pool of vehicles surrounding the ego vehicle within a 50-meter radius. All scenarios in VTrackIt are generated using the CARLA simulator 0.9.13 [184]. Although several state-of-art simulators such as Nvidia DriveSim[3], Autonovi-sim [190], and GTA-V [191] exist, these simulators are not open source and may limit the future extension of this dataset. Although simulators such as TORCS [192] and AirSim [193] are publicly available, they offer limited sensor suites and lack the photorealistic rendering capabilities needed to train modern perception-based DL algorithms. Due to its seamless integration with Unreal Engine[4], CARLA offers several advantages over other mentioned simulators, such as (i) enhanced realism, (ii) sophisticated vehicle dynamics, (iii) map customizations, and (iv) realistic traffic customizations.

The VTrackIt dataset only includes information from RGB sensors at this time since the information extracted from other sensors such as LiDAR and Radar are readily available in the V2I and V2V tags with no uncertainty. The VTrackIt dataset is generated using 'DirectSim,' a custom synthetic scenario generation platform that will be released soon and is considered to be out-of-scope in this discussion.

5.3.1 Sensing Package, Lane Annotations, and Aerial BEV

Table 5-2 Sensing Package

Sensor	**Sensor Position Code**	**Location (in m)**			**Sensor Details**
		X	**Y**	**Z**	
RGB Camera (Front)	a	0.90	0.0	1.44	Forward facing stereo camera with 120° field-of-view.
RGB Camera (Right)	b	0.90	1.20	1.44	
RGB Camera (Left)	c	0.90	-1.20	1.44	
RGB Camera (Rear)	d	-0.90	-1.20	1.44	
IMU	e	0.0	0.0	0.0	Inertial measurement sensor to record ego vehicle information.
GNSS	e	0.0	0.0	0.0	Global navigation sensor used to record ego vehicle pose information.

[3] https://developer.nvidia.com/drive/simulation

[4] https://www.unrealengine.com/

To generate synthetic data for the VTrackIt dataset, we equip the ego vehicle (Tesla Model 3) with four highly synchronized RGB cameras, one IMU, and one GNSS sensor. Specifications of the sensing package are given in Table 5-2. Figure 5-2 visualizes the sensor positions and their orientations from the vehicle center as defined in CARLA. While the four RGB cameras with 120° Field-of-View (FoV) provide a 360° view around the ego vehicle, VTrackIt also provides an aerial BEV of the ego vehicle. All scenarios are captured and tagged at 20 FPS.

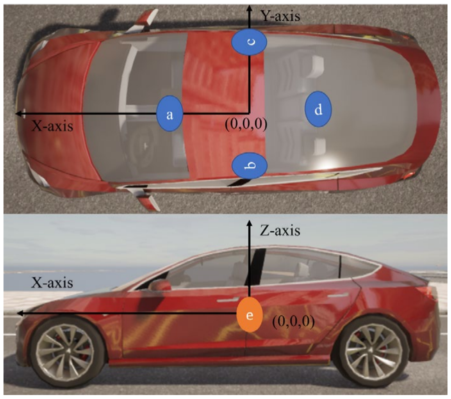


Figure 5-2 Co-ordinate system for data recorded and locations of sensors with their respective sensor codes in the VTrackIt dataset.

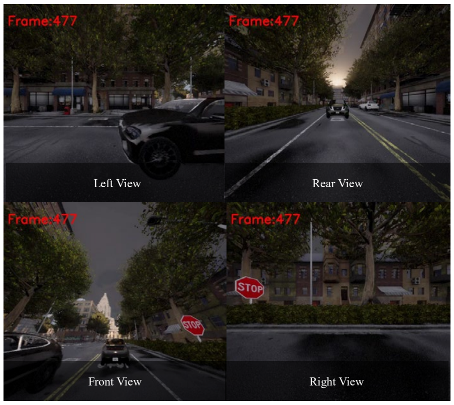


Figure 5-3 360-degree perception coverage around the ego vehicle using four RGB cameras.

Data from the surrounding vehicles can thus be used to train DL solutions for AVs in addition to the ego vehicle data. Therefore, in addition to the provided perception information, VTrackIt also includes annotations for surrounding vehicle information, including their locations (x, y, and z), attributes, color, extents (length, width, and height), heading, steer, throttle, and brake pedal positions, along with the relative position from the center of the ego vehicle to the center of the tracked traffic vehicle. Furthermore, VTrackIt also annotates several infrastructure information variables, including lane line color and lane line type on both sides, possible maneuvers (lane restrictions), lane width, vehicle deviation from the centerline, and red-light status. Such data is provided for the ego vehicle and a pool of surrounding vehicles. Table 5-3 provides detailed information on all annotated variables, data types, ranges, and measurement units. A brief description of each variable can also be seen in Table 5-3. Figure 5-3 shows a sample frame in a random scenario captured by all four RGB cameras giving the ego vehicle a 360° view of its surroundings. Note that each variable is annotated with a corresponding frame number that helps synchronize perception-based sensor information with other variables in the VTrackIt dataset.

In our experiments with 'VTrackIt,' we equip the ego vehicle with four highly synchronized RGB cameras, one IMU, and one GNSS sensor. Specifications of all attached sensors are given in Table 5-2. Figure 5-2 visualizes the sensor positions and their orientations from the vehicle center as defined in CARLA. While the four RGB cameras provide a 360° view around the vehicle, we also provide an aerial BEV of the ego vehicle. All sequences are recorded and tagged at 20 FPS along with corresponding IMU and GNSS measurements. In addition to RGB images, we also provide advanced vehicle information for all tracked vehicles, including their locations, attributes, color, extents, heading, steer, throttle, brake pedal positions, along with the relative position from the center of the ego vehicle. Recorded lane information includes lane line color and lane line type on both sides, lane type, possible maneuvers, lane width, vehicle deviation from the centerline, and red-light status. Similar lane information is also provided for all tracked vehicles surrounding the ego vehicle. Data from these vehicles can thus be used to train ML algorithms in addition to the ego vehicle data.

Table 5-3 Details of Variables Recorded at Every Frame in the VTrackIt Dataset

Recorded Property	**Data Type**	**Description**	**Range / Possible Values**	**Unit of Measurement**
timestamp	Float	Time stamp of the measurement	[0,inf) Increments by 0.05	Seconds
frame	Integer	Frame number of measurement	[0,inf) Increments by 1	-
actor_id	Integer	Unique id given to every actor in simulation	[0,inf)	-
actor_type	String	Identifier given to distinguish ego vehicle from traffic vehicles	Ego / Traffic	-
attr	String	Classification of vehicle in CARLA standards		-
color	Tuple	RGB values of given actor	(0-255,0-255,0-255)	-
pos_x	Float	Global location of given actor along X-axis in cartesian co-ordinate system	[-inf,inf]	Meters
pos_y	Float	Global location of actor along Y-axis in cartesian co-ordinate system	[-inf,inf]	Meters
pos_z	Float	Global location of actor along Z-axis in cartesian co-ordinate system	[-inf,inf]	Meters
heading	Float	Global heading of actor relative to true North	(0,360]	Degrees
extents	List	Actor length and width measured end-to-end	[0,inf]	Meters
speed	Float	Actor speed	-	KMPH
acceleration	List	Actor acceleration in X and Y-axis	-	m/s^2
throttle	Float	Throttle pedal position for a given actor	(0,1)	-
steer	Float	Steer angle for a given actor	(-1,1)	-
brake	Float	Brake pedal position a given actor	(0,1)	-
red_light	Binary	Unique identifier that is set to '1' if the vehicle is directly affected by a red light.	0/1	-
rel_angle	Float	Relative angle of a traffic vehicle measured from the center of ego vehicle; for example, vehicle in front is 90°	(0,360]	Degrees
rel_x	Float	Relative position of a traffic vehicle measured from the center of ego vehicle along its y-axis;	(0,50)	Meters
rel_y	Float	Relative position of a traffic vehicle measured from the center of ego vehicle along its x-axis	(0,50)	Meters
lane_type	String	Lane type affecting actor location	Driving / Junction / Shoulder	-
right_lane_mark_type	String	Right lane marking type affecting given actor location	Solid / Broken / SolidSolid / NONE	-
right_lane_mark_color	String	Right lane marking color affecting given actor location	White / Yellow	-
left_lane_mark_type	String	Left lane marking type affecting given actor location	Solid / Broken / SolidSolid / NONE	-
left_lane_mark_color	String	Left lane marking color affecting given actor location	White / Yellow	-
possible_manuvers	String	Permissible lane changes for a given actor based on its location	Left / Right / Both / None	-
lane_width	Float	Width of driving lane based on given actor location	(0,inf]	Meters
off_center	Float	Deviations along lane center lines recorded in along given actor's X-axis.	(0,inf]	Meters

5.3.2 Diversity in Weather and Typologies

A significant limitation of existing synthetic datasets is their lack of ability to replicate real-world driving conditions. Thus, it is of high interest to, (i) simulate scenarios that depict close to real-life road typologies, and (ii) simulate scenarios that adequately consider the interaction of vehicles with their surrounding environment. Furthermore, a synthetic dataset such as VTrackIt must also include scenarios under varying operating conditions, including those induced by adverse weather. We thus randomize the weather in different maps in VTrackIt. Typically, we define and randomly sample weather from various weather conditions ranging from noon to sunset, clear to foggy, and from dry to wet. We also modify the road friction to closely mimic real-life conditions based on values given by Hall *et al.* [194] for wet weather scenarios. Figure 5-4 visualizes some of these weather conditions. Table 5-4 details various CARLA maps in VTrackIt.

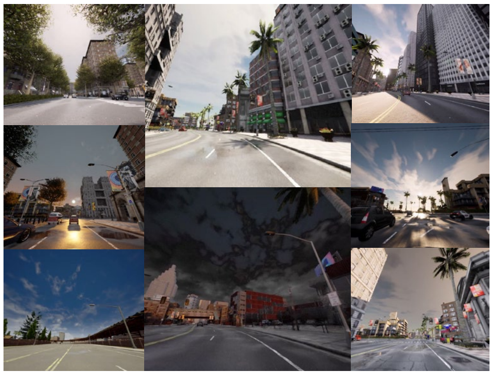


Figure 5-4 Diverse set of weather and typologies included in the VTrackIt dataset.

Table 5-4 Map and Road Features

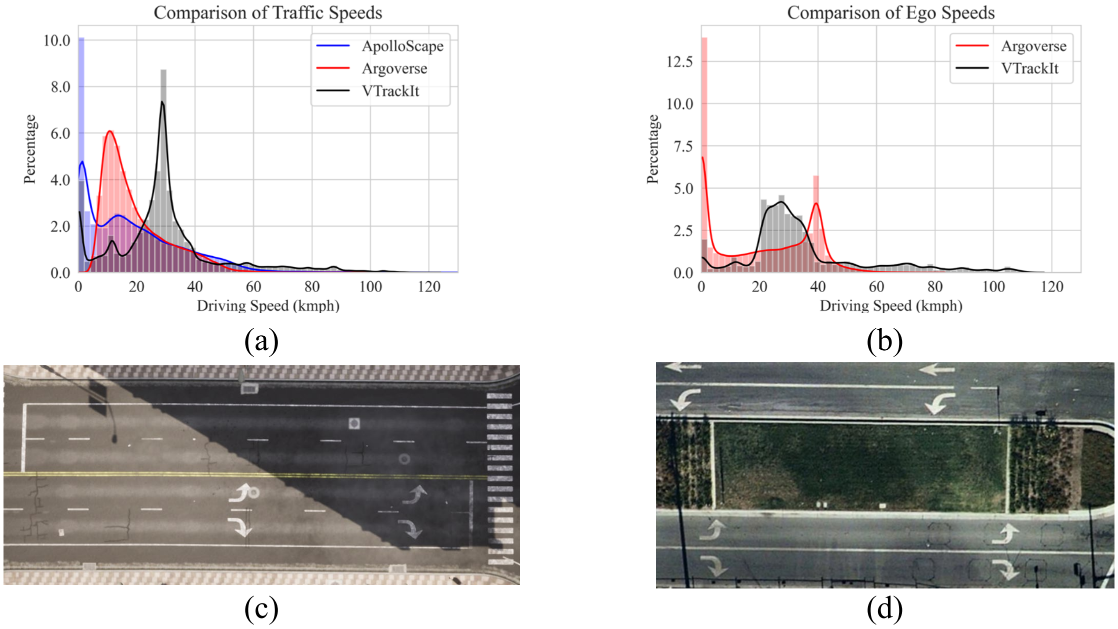
Map Name	**Location**	**Salient Road Features**
Town01	Low density urban (20-45 kmph)	3-way intersections, single-lane roads, traffic lights, and stop signs.
Town03	Urban-Highway (20-90kmph)	5-lane intersections, roundabouts, uneven grades, tunnels, highways, traffic lights, and stop signs.
Town04	Highway (20-90kmph)	Lane merges, highway ramps, 4-way junctions.
Town05	Hybrid (20-90 kmph)	Multiple highway and urban lanes, 4-way junctions.
Town06	Highways (20-90 kmph)	Long highways, ramps, Michigan-left, round abouts, lane merges.
Town10HD	Urban (20-45 kmph)	High fidelity road textures, parking lanes, stop signs, traffic lights, pedestrian crossings, and junctions.

5.3.3 Traffic and Ego Vehicle Customizations

As mentioned earlier, interactions with the surrounding environment and traffic regulations play a significant role in developing DL solutions for AVs. Traffic is undoubtedly one aspect of it. However, no two vehicles are the same in real life because of the complex interaction between factors such as driver behavior and intrinsic differences in the vehicle, such as tire wear, vehicle age, and many more. Unfortunately, most synthetically generated datasets fail to intelligently model such vehicle attributes and assume similar attributes for all traffic actors. In their work, Weng *et al.* [178] briefly discuss and try to address this issue. In our case, we adopt a fuzzing strategy and randomize every traffic actor in every scenario by varying its (i) vehicle type, (ii) vehicle color, (iii) minimum following distance, (iv) maximum speed over/under speed limits, (v) probability of ignoring other vehicles, and (vi) probability of ignoring traffic regulations, as defined in the CARLA documentation [184]. All of these variable values are sampled randomly from a uniform distribution for every variable, for every actor, in every scenario, based on real-life values extracted from naturalistic driving studies such as [195]–[199]. Furthermore, a few selected vehicles in every scenario are uniquely modified using out-of-distribution values to represent overly aggressive and cautious drivers. Such drivers are known to pose additional risks to the ego vehicle that may result in edge cases. Finally, to augment the realism of wet weather scenarios with low friction conditions, we modify the road-surface friction values based on [194]. Further, we also modify the speed limits in that scenario based on naturalistic driving studies [200]–[202].

In our experiments, we use the same ego vehicle model in all scenarios to remain consistent with other real-life datasets. The simulator parameters for every scenario are sampled from predefined statistical distributions. Sampled parameters include safe distance, maximum speed over/under speed limits, probability of ignoring other vehicles, and probability of ignoring traffic regulations.

5.3.4 Comparisons to Real-world Datasets



(a)

(b)

(c)

(d)

Figure 5-5 Comparison of VTrackIt dataset using – (a) traffic speed distributions from the ApolloScape, and Argoverse datasets (for non-stationary vehicles), (b) ego speed distribution from the Argoverse dataset (for non-stationary vehicles), and (c) BEV in the VTrackIt dataset, and (d) BEV in Level 5 dataset.

In this work, we are specifically interested in replicating real-world traffic speed distributions and expanding beyond its right tail to encourage out-of-distribution events that may lead to high-risk edge cases. Such scenarios are purposely excluded in real-world driving for obvious safety reasons; however, this introduces a bias towards safe scenarios. As shown in Figure 5-5 (a-b), while the ApolloScape and Argoverse datasets contain many low-speed vehicles, the VTrackIt dataset expands the right-tail of the distribution by spawning more medium-high speed vehicles. The VTrackIt dataset thus capitalizes on the power of synthetic data generation tools to generate a wider distribution of traffic and ego vehicles. Such vehicles are more likely to result in edge cases in a real-world scenario but not in a scenario released by companies promoting AV solutions. Finally, Figure 5-5 (c-d) shows the realism of the BEV provided by VTrackIt compared to the BEV provided by the Level5 dataset. Table 5-5 highlights some salient features of the VTrackIt dataset compared to other state-of-the-art self-driving datasets.

Table 5-5 Diversity of VTrackIt in Comparison with Commonly Used Self-driving Datasets

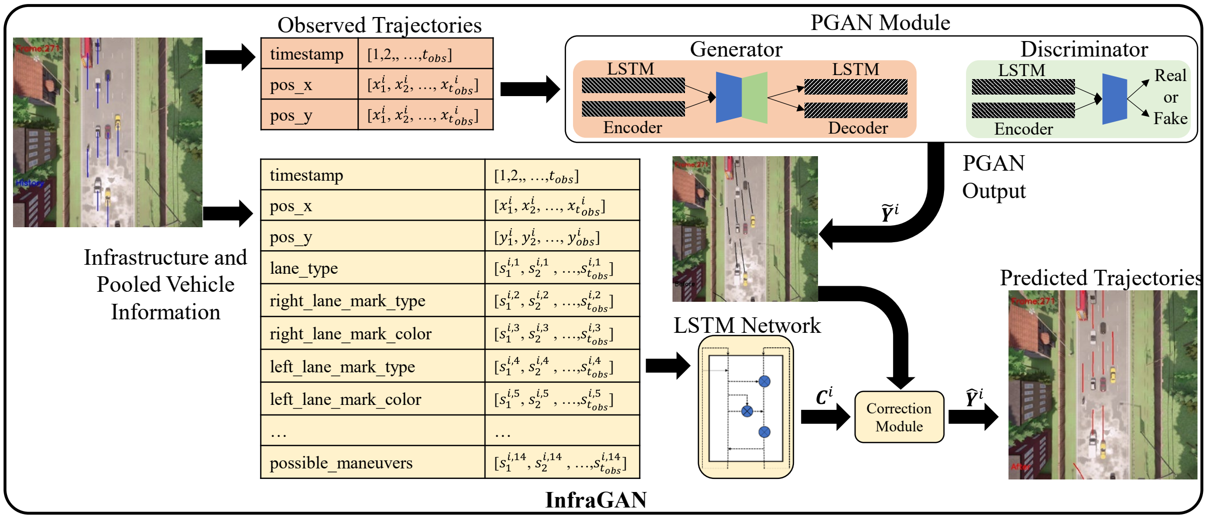
Dataset	**Crashes**	**Varying Road Surface Conditions**	**Dynamic Speed Limits**	**Highway Driving**	**Traffic Rule Violations**	**Average Lane Width (m)**	**Lane Center Annotations**	**360° Cameras w/ BEV**
ApolloScape	-	✓	-	-	-	-	-	-

Argoverse	-	✓	-	-	-	3.84(Miami);
						3.97(Pittsburgh)
							✓	-

AIODrive	✓	-	-	✓	✓	3.5	-	✓ (No BEV)
VTrackIT (Proposed)	✓	✓	✓	✓	✓	3.5	✓ (lane deviations are given by 'off_center')	✓ 360° and BEV

5.4 Case Study for Trajectory Predictions

Although the VTrackIt dataset can be used for a wide range of self-driving applications, we focus on trajectory prediction tasks in this section. Trajectory prediction is commonly employed in AVs to identify vehicles that present a higher risk to the ego vehicle given its planned trajectory [203]. In their work, Lefèvre *et al.* [203] categorize all trajectory prediction models into three broad categories, namely, (i) Physics-based models [204], [205], (ii) Maneuver-based models [206]–[210] and, (iii) Interaction-aware models [58], [60], [63], [64], [66], [68], [183], [211]–[218]. Today, Graph-based interaction-aware models leveraging semantic maps are considered state-of-the-art [60], [214], [219], [220]. Recently, many DL models that leverage BEV representations to capture complex interactions with other road agents have been proposed [20], [221]–[225]. Having said that, such models often ignore traffic rules' effect due to limited access to infrastructure and surrounding vehicle information. Although many trajectory prediction approaches have been proposed [34], [35], [65], [72]–[74], in this work, we propose the first model (InfraGAN) that leverages infrastructure and pooled vehicle information for accurate trajectory predictions. As shown in Figure 5-6, InfraGAN is a two-part network that consists of an interaction-aware prediction module: PGAN (Pooled-GAN) and an LSTM network with a Correction Module (CM). Details of the InfraGAN and its individual components are provided in the following sections.



timestamp	$[1,2,, \dots, t_{obs}]$
pos_x	$[x_1^i, x_2^i, \dots, x_{t_{obs}}^{\ i}]$
pos_y	$[x_1^i, x_2^i, \dots, x_{t_{obs}}^{\ i}]$

timestamp	$[1,2,, \dots, t_{obs}]$
pos_x	$[x_1^i, x_2^i, \dots, x_{t_{obs}}^{\ i}]$
pos_y	$[y_1^i, y_2^i, \dots, y_{obs}^i]$
lane_type	$[s_1^{i,1}, s_2^{i,1}, \dots, s_{t_{obs}}^{i,1}]$
right_lane_mark_type	$[s_1^{i,2}, s_2^{i,2}, \dots, s_{t_{obs}}^{i,2}]$
right_lane_mark_color	$[s_1^{i,3}, s_2^{i,3}, \dots, s_{t_{obs}}^{i,3}]$
left_lane_mark_type	$[s_1^{i,4}, s_2^{i,4}, \dots, s_{t_{obs}}^{i,4}]$
left_lane_mark_color	$[s_1^{i,5}, s_2^{i,5}, \dots, s_{t_{obs}}^{i,5}]$
…	…
possible_maneuvers	$[s_1^{i,14}, s_2^{i,14}, \dots, s_{t_{obs}}^{i,14}]$

Figure 5-6 Overview of the InfraGAN baseline for trajectory prediction using infrastructure and pooled vehicle information in the VTrackIt dataset. The variables $s_t^{i,1}, \dots, s_t^{i,14}$ represent the 14 annotated information variables emboldened in

Table 5-3.

5.4.1 InfraGAN: PGAN Module

The PGAN module is similar to common GAN-based DL models that consist of two competing LSTM-autoencoders (Generator and Discriminator) trained to achieve a zero-sum game where the Discriminator attempts to distinguish between generated and actual samples [20]. The Generator (G) outputs an array of predicted trajectories $(\widetilde{\boldsymbol{Y}})$ for several pooled vehicles $G(z, \boldsymbol{X})$, given a random seed z and an array of observed trajectories $\boldsymbol{X}$. The Discriminator (D) then classifies these predicted trajectories by returning $D(\widetilde{\boldsymbol{Y}}|\boldsymbol{X})$ where $D(\widetilde{\boldsymbol{Y}}|\boldsymbol{X})$ represents the probability that the predicted trajectories are real (from training data rather than the Generator). The objective of PGAN is thus to train a robust generator that outputs potential samples of predicted trajectories by solving the min-max problem in eq. (5-1).

$$\min_{G}\max_{D}\ E_{\widetilde{\boldsymbol{Y}}|\boldsymbol{X}}\big[\log D\big(\widetilde{\boldsymbol{Y}}|\boldsymbol{X}\big)\big] + E_{Z}[\log(1 - D(G(z,\boldsymbol{X})|\boldsymbol{X})] \tag{5-1}$$

Here, $\boldsymbol{X} = [\boldsymbol{X}^1, \dots, \boldsymbol{X}^n]$ and $\boldsymbol{Y} = [\boldsymbol{Y}^1, \dots, \boldsymbol{Y}^n]$, where n is the number of pooled vehicles, including the ego vehicle. Let $\boldsymbol{X}^i = \{(x_t^i, y_t^i)\}_{t \le t_{\mathrm{obs}}}$ summarize the observed trajectory of vehicle i, and $\boldsymbol{Y}^i = \{(x_t^i, y_t^i)\}_{t_{\mathrm{obs}} < t \le t_{\mathrm{obs}} + T}$ summarize the actual trajectory of vehicle i, where x_t^i and y_t^i represent the x and y co-ordinates of vehicle i. T is the number of prediction timesteps, and z is sampled from $\mathcal{N}(0,1)$.

When given an input $\boldsymbol{X}^i$, G outputs a predicted trajectory $\widetilde{\boldsymbol{Y}}^i$. The Discriminator then uses this predicted trajectory $\widetilde{\boldsymbol{Y}}^i$ or the actual trajectory $\boldsymbol{Y}^i$ as input and classifies it as ‘Real’ or ‘Fake.’ To standardize the input data, we encode the position of a vehicle i at time t such that,

$$e_t^i = \phi(x_t^i, y_t^i; W_{ee}) \tag{5-2}$$

Here, $\phi(.)$ represents a single layer, shallow neural network with ‘ReLU’ non-linearity, and W_{ee} are the weights of the shallow neural network. The LSTM-encoder then uses e_t^i as an input and produces encodings h_t^{ei} for every vehicle i at time t such that,

$$h_t^{ei} = \mathrm{LSTM_e}(h_{t-1}^{ei}, e_t^i; W_{encoder}) \tag{5-3}$$

Here, $W_{encoder}$ represents the weights of the LSTM encoder ($\mathrm{LSTM_e}$) that are shared between all input vehicles at time t based on suggestions given by Alahi *et al.* [222]. The trajectory predictions can thus be obtained such that,

$$(\tilde{x}_t^i, \tilde{y}_t^i) = \gamma(h_t^{di}) \tag{5-4}$$
$$h_t^{di} = \mathrm{LSTM_d}(h_{t-1}^{di}, e_t^i; W_{decoder}) \tag{5-5}$$
$$e_t^i = \phi(x_{t-1}^i, y_{t-1}^i; W_{de}) \tag{5-6}$$

Here, $W_{decoder}$ represents the weights of the LSTM-decoder ($\mathrm{LSTM_d}$), W_{de} represents embedding weights, and γ is an MLP. The probability of a trajectory being real/fake is obtained by applying an MLP on the encoder's final hidden state. Note that $\tilde{x}_t^i$ and $\tilde{y}_t^i$ represent the predicted x and y co-ordinates for some vehicle i at time t. While LSTM-encoder consists of 16 cells, the LSTM-decoder consists of 32 cells.

5.4.2 InfraGAN: Correction Module (CM)

To leverage infrastructure and surrounding vehicle information given in the VTrackIt dataset, the spatial co-ordinates and 14 additional variables emboldened in Table 5-3 are given as input to an LSTM network with ten cells and a CM. The LSTM network combined with the CM reasons and corrects the PGAN predicted trajectories using this additional infrastructure and surrounding vehicle information, if necessary. For example, the CM tends to correct trajectories that violate lane regulations or trajectories that may lead to crashes. $\widehat{\boldsymbol{Y}}^i$ represents the trajectories predicted using InfraGAN and are mathematically given by the CM such that,

$$\widehat{\boldsymbol{Y}}^i = \widetilde{\boldsymbol{Y}}^i + \tanh\left(\boldsymbol{C}^i\right) * \widetilde{\boldsymbol{Y}}^i \qquad (5\text{-}7)$$

Here, $\tanh\left(\boldsymbol{C}^i\right)$ is the correction factor for the PGAN predictions using the LSTM network that utilizes the infrastructure and surrounding vehicle information provided in the VTrackIt dataset.

The PGAN is first trained for 200 epochs to produce acceptable trajectories using both the adversarial loss in eq. (5-1) and the mean squared error between the predicted and actual trajectories. Then, the entire InfraGAN is trained to correct for PGAN predictions using the 'Adam' optimizer for additional 200 epochs with an adaptive learning rate. The PGAN is expected to adjust its weights to work in tandem with the LSTM network to achieve reliably corrected trajectories that minimize the loss function of the InfraGAN given in eq. (5-8).

$$\mathcal{L}_{\text{InfraGAN}} = \sum_{i} \sum_{k} \left\{ \left\| \widehat{\boldsymbol{Y}}^i - \boldsymbol{Y}^i \right\|_2^2 + \frac{1}{\left\| \boldsymbol{C}^i \right\|_1} \right\} \tag{5-8}$$

In all our experiments, we configure the InfraGAN to generate five likely trajectories ($k = 5$) for every input vehicle i by randomly sampling z from a standard normal distribution.

5.4.3 Trajectory Prediction Benchmarks

Most trajectory prediction models and AV modules operate at 2-2.5 FPS [20], [58], [67], [214], [226]–[228]. Thus, we down sample the VTrackIt dataset to 2.5 FPS in this case study. The scenarios in the VTrackIt 'train' and 'val' sets are then used to train and validate the PGAN and the InfraGAN models. More importantly, to avoid divergence while training the DL model, we exclude scenarios where the ego vehicle had a collision but may still have scenarios where surrounding actors collide. In all our experiments, we observe vehicle trajectory for eight timesteps (or 3.2 secs) and predict trajectory for the following eight timesteps unless mentioned otherwise. We perform evaluations during testing using the three most used criteria [174], [175], [178]. The Minimum Average Displacement Error (minADE) is the average displacement error for the best-predicted trajectory over all scenarios. The Minimum Final Displacement Error (minFDE) is the Final Displacement Error for the best-predicted trajectory averaged over all scenarios. The Miss Rate is the percentage of all predicted trajectories with the final displacement error above 2.0 meters over all scenarios.

While the goal of this work is not to compare baseline models, it is critical to understand how the additional infrastructure and pooled vehicle information affects the trajectory prediction algorithms. Moreover, emphasis is given on analyzing the right tails of error distributions. Right tails of error distributions represent infrequent cases with high errors and are typically referred to as ‘edge cases.’ Therefore, in our work, to provide a fair comparison to future models, we provide benchmarks using both: the PGAN and the InfraGAN models. Note that the results reported using the PGAN model do not use any infrastructure and pooled vehicle information apart from the ego and tracked vehicle’s x and y co-ordinates. As intuition suggests, the InfraGAN significantly improves over the PGAN by accurately predicting trajectories using the infrastructure and pooled vehicle information in the VTrackIt dataset. This performance gain can be seen in Figure 5-7.

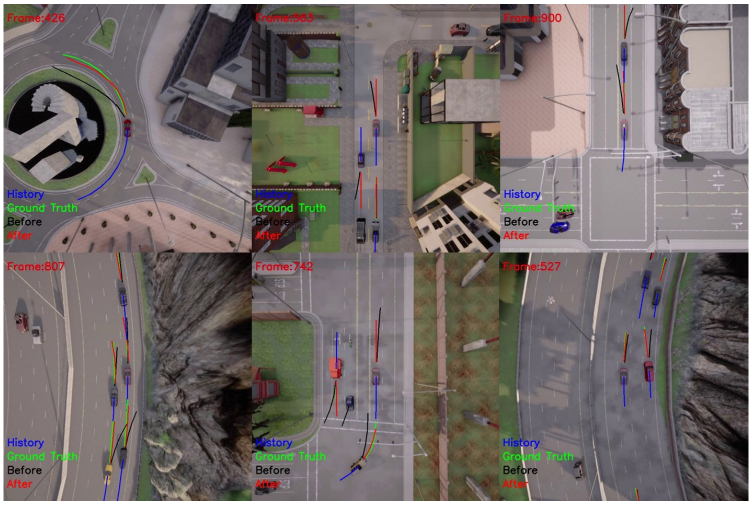


Figure 5-7 Performance comparison with and without infrastructure and pooled vehicle information.

Table 5-6 can be used to verify this performance improvement using the InfraGAN over the PGAN when predicting k most likely trajectories using the specified metrics. As is observed in similar GAN-based approaches [20], the PGAN performs significantly worse when put to the test using smaller values for k. Further, we also report individualized metrics for every considered map in the test set using Table 5-7. Table 5-8 can be used to analyze the performance of both considered models over varying prediction horizons. Figure 5-8 and Figure 5-9 show a clear improvement in the error distributions across ADE and FDE for all predicted trajectories using the PGAN and InfraGAN models. From these figures, we can clearly see a shorter right tail in the error distributions using InfraGAN. Thus, we can conclude that the infrastructure and pooled vehicle information provided by VTrackIt can support the development of newer trajectory prediction models that experience significantly fewer edge cases.

Table 5-6 Trajectory Prediction Benchmarks Using Top k Predictions

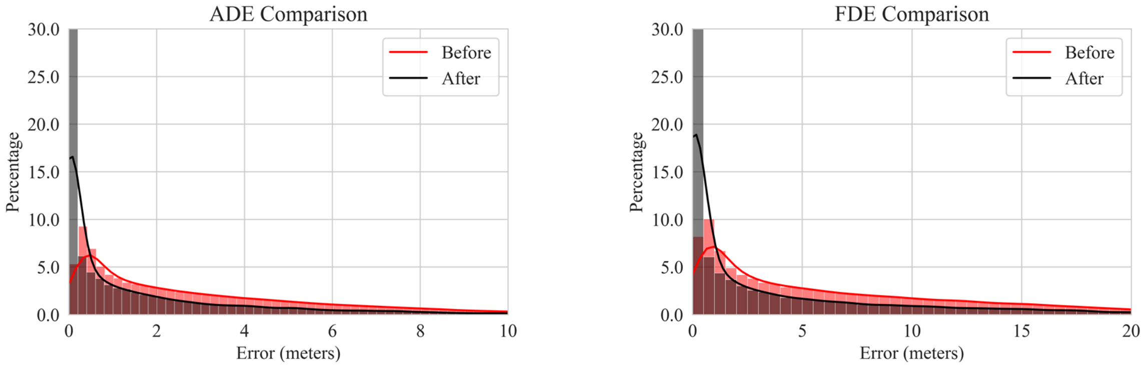
Metric	**K=1**	**K=3**	**K=5**	**Average**
Without Infrastructure and Pooled Vehicle Information (PGAN)				
minADE	3.19	2.11	1.87	2.39
minFDE	7.15	4.55	3.97	5.22
Miss Rate	0.74	0.60	0.56	0.63
With Infrastructure and Pooled Vehicle Information (InfraGAN)				
minADE	1.56	1.56	1.55	1.55
minFDE	3.58	3.57	3.56	3.56
Miss Rate	0.43	0.42	0.42	0.42

Table 5-7 Comparison of Benchmark Models on the VTrackIt Test Set Per Map

Metric	**Town01**	**Town03**	**Town04**	**Town05**	**Town06**	**Town10**	**Overall**
Without Infrastructure and Pooled Vehicle Information (PGAN)							
minADE	1.50	1.93	2.74	**1.13**	2.45	1.62	1.87
minFDE	3.20	4.14	5.61	**2.43**	5.15	3.52	3.97
Miss Rate	0.45	0.58	0.70	**0.34**	0.70	0.51	0.56
With Infrastructure and Pooled Vehicle Information (InfraGAN)							
minADE	1.29	1.69	2.07	**0.91**	2.33	1.20	1.55
minFDE	2.93	3.89	4.75	**2.15**	5.34	2.75	3.56
Miss Rate	0.32	0.44	0.51	**0.25**	0.59	0.35	0.42

Table 5-8 Comparison of Benchmark Models on the VTrackIt Test Set Over Varying Prediction Horizons

Metric	**Pred (2s)** **5 Time steps**	**Pred (2.4s)** **6 Time steps**	**Pred (3.2s)** **8 Time steps**	**Average**
Without Infrastructure and Pooled Vehicle Information (PGAN)				
minADE	0.63	0.95	1.87	1.17
minFDE	1.93	2.53	3.97	2.81
Miss rate	0.36	0.44	0.56	0.54
With Infrastructure and Pooled Vehicle Information (InfraGAN)				
minADE	0.49	0.75	1.55	0.93
minFDE	1.56	2.14	3.56	2.42
Miss rate	0.28	0.34	0.42	0.34



(a) (b)

Figure 5-8 Comparison of error distributions with and without infrastructure and pooled vehicle information using the VTrackIt test set over – (a) ADE, and (b) FDE. Errors for trajectories predicted using PGAN and are labeled as 'Before'. Errors for trajectories predicted using InfraGAN and are labeled as 'After.'Ego Trajectory Prediction Baseline

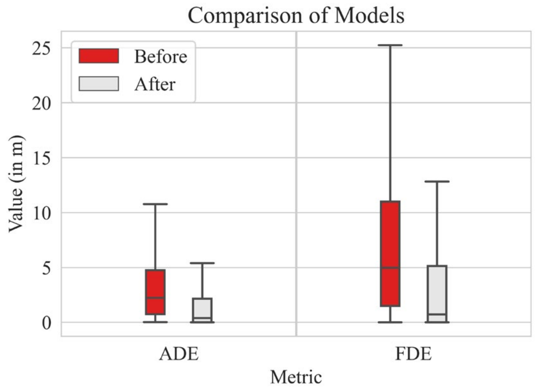


Figure 5-9 Comparison of baseline models using the VTrackIt test set. Errors for trajectories predicted using PGAN and are labeled as 'Before'. Errors for trajectories predicted using InfraGAN and are labeled as 'After.'

5.4.4 Ego Trajectory Prediction Benchmarks

Table 5-9 Ego Trajectory Prediction Benchmarks Using Top k Predictions on the VTRACKIT Test Set

Metric	$\boldsymbol{k = 1}$	$\boldsymbol{k = 3}$	$\boldsymbol{k = 5}$	**Average**
Without Infrastructure and Pooled Vehicle Information (PGAN)				
minADE	3.87	2.28	1.89	2.68
minFDE	8.63	4.81	3.88	5.77
Miss Rate	0.75	0.57	0.51	0.61
With Infrastructure and Pooled Vehicle Information (InfraGAN)				
minADE	1.80	1.79	1.78	1.79
minFDE	4.21	4.18	4.17	4.18
Miss Rate	0.40	0.40	0.40	0.40

Datasets such as VTrackIt may also be used to build robust motion planning algorithms. As a related task, we use the VTrackIt dataset and utilize recorded data for all tracked vehicles around the ego vehicle while only predicting the trajectory for the ego vehicle. Table 5-9 can be used to compare the results of the baseline models when predicting k most likely trajectories using mentioned metrics. Using this table, we observe a similar pattern where the InfraGAN outperforms the PGAN. This performance gain can also be attributed to the additional infrastructure and vehicle information provided in the VTrackIt dataset.

5.5 Conclusion

In this study, we present VTrackIt [89], a novel, publicly available synthetic dataset specifically designed for advancing AV research. VTrackIt stands out as one of the largest datasets of its kind, explicitly annotated with infrastructure and pooled vehicle information. This dataset aims to apprise researchers of the benefits such data may provide in building advanced algorithms for AV applications. The VTrackIt dataset has been generated to closely emulate real-world driving conditions, and includes out-of-distribution scenarios, which may be hard to collect in real-world conditions. The efficacy of data provided by the VTrackIt dataset is validated using a compelling case study on trajectory predictions. For this, we develop a novel deep learning model named InfraGAN that leverages the data provided in the VTrackIt dataset. The results of our experiments using InfraGAN show that the use of infrastructure and surrounding vehicle information vastly helps improve trajectory predictions, with fewer edge cases. For future work, we plan to enhance the VTrackIt dataset by adding more sensor data and challenging scenarios, as well as exploring other use cases that highlight the importance of the information in the dataset.

Chapter 6 Summary of Original Contributions and Future Work

6.1 Summary of Original Contributions

This work largely contributes to the field of predictive analytics. More specifically, this dissertation proposes custom AI architectures and frameworks that leverage Bayesian statistics and DL methods for improved predictive analytics. The major contributions of this work are:

1. Uncorrelated Sparse Autoencoders with Long Short-term Memory for Long-term Memory (USAL) for State-of-Charge (SOC) Estimations in Battery Cells.

This work: (i) proposes a multi-task training strategy to learn efficient encodings given limited training data, (ii) penalizes the learned encodings for high correlations to efficiently transform measured inputs into a space of informative features, and (iii) identifies and maps data-driven trends to support long-term estimations. The proposed method, USAL, is designed to be a data-driven SOC estimation method that is, (i) capable of alerting the user to a faulty cell when integrated into a real-life Battery Management System (BMS) and (ii) identifying the relative quality of a battery cell from only a few initial charge-discharge cycles. The experimental results show that the proposed USAL outperforms state-of-the-art while performing (i) long-term SOC estimations, (ii) short-term SOC estimations, and (iii) SOC estimations in dynamic cycles.

2. Conditional Gaussian Mixture Model (CGMM) for Warranty Claims Forecasting

This work proposes CGMM, a custom AI architecture for long-term warranty claims forecasting. The proposed CGMM (i) attempts to provide a solution to the data drift in warranty data by modeling the joint distribution of mature and immature warranty claims as a Gaussian Mixture Model (GMM), (ii) avoids forcing a time-dependent parametric function for claim's forecasting, (iii) uses conditional Bayesian inference to consider part similarity, and (iv) provides an epistemic uncertainty quantification measure. The effectiveness of the proposed CGMM was proven using a real-life dataset consisting of more than 15,000 unique parts installed on different vehicle lines for different model years. The results show that the CGMM can outperform commonly used benchmarked models like the NHPP, NHPP-W, NHPP-C and RNNs.

3. VTrackIt: A Synthetic Self-Driving Dataset with Infrastructure and Pooled Vehicle Information.

This work introduces InfraGAN, a novel deep learning architecture that performs collaborative modeling using infrastructure and pooled vehicle information for accurate trajectory predictions. The results of our experiments using InfraGAN show that the use of infrastructure and surrounding vehicle information vastly helps improve trajectory predictions with fewer edge cases. However, the data required to train such models is not publicly available. Therefore, in this work, we also publicly release VTrackIt, a synthetic dataset explicitly annotated with intelligent infrastructure and pooled vehicle information. VTrackIt is the largest dataset of its kind that includes (i) explicit tags for infrastructure and pooled vehicle information, (ii) scenarios generated to replicate real-world driving data and expand its tails to generate out-of-distribution data, (iii) a video database with high-resolution Birds-Eye-View (BEV) and 360° views of the ego vehicle for better scenario understanding and visualizations.

4. Bayesian Optimization for Scenario Exploration (BOSE) in Autonomous Vehicles (AVs)

This work proposes BOSE, an automated, search-based framework for scenario exploration in AVs. The proposed framework (i) learns robust priors that help identify critical scenarios of failure while developing a complete understanding of the underlying functional scenario, (ii) ensures fast convergence by sampling high-entropy scenarios limiting sequential scenario exploration effort, and (iii) employs a Bayesian DNN with LHS to scale effortlessly across complex, multi-dimensional scenario spaces. The proposed framework is thus shown to be interpretable, efficient, and scalable given low as well as high-dimensional scenarios. In our experiments, BOSE was shown to reduce sequential scenario exploration experiments by over 30% when compared to the state-of-the-art. BOSE is, therefore, a robust yet accurate scenario exploration method that helps in the verification and validation of modern AVs by accurately modeling and predicting anomalous events in an AI-driven system.

6.2 Future Work

Although this work focusses on the development of custom AI models and data-driven frameworks that solve some challenges in the area of predictive analytics, there are some improvements that can be made in future studies.

1. *Extension of Proposed Custom AI Models Across Multiple Domains*

Although the proposed custom AI architectures in this dissertation focus on automotive applications, they are built to provide solutions to the six mentioned challenges in predictive analytics. Therefore, it is of high interest to extend these architectures across multiple domains and verify and validate their performance. In one example, while USAL's combination of training losses is effective for the given task, it still faces limitations in achieving completely orthogonal encodings in the sparse autoencoder. Therefore, it is of our high interest to develop theoretically sound AI architectures that enforce complete orthogonality for accurate long-term estimations.

2. *Development of Bayesian Models for Verification and Validation of Autonomous Vehicles*

The verification and validation of ADSs can be a complex undertaking due to the "black-box" nature of the many sub-systems used in ADSs. While the BOSE approach has demonstrated the ability to explore and comprehend "known" low and high-dimensional scenarios with a Level V ADS, it is essential to extend this work to address "unknown" scenarios. Many real-world ADSs encounter unforeseen situations that can result in anomalous events or emergent behavior. Therefore, it may be necessary to develop a Bayesian-style framework that can predict the response of the entire system without conducting simulations in the near future. This approach would provide a more comprehensive understanding of the system's behavior and allow for adequate verification and validation of ADSs in the presence of "unknown" scenarios.

3. AI-Driven Predictive Analytics for Improved Reliability of Charging Stations

The increasing adoption of EVs has resulted in a growing demand for electric charging infrastructure, leading to the development of thousands of charging stations worldwide. However, maintaining the reliability of these chargers is a significant challenge, as they are subjected to various environmental and usage-related stresses, including extreme temperatures, voltage fluctuations, and physical damage. These issues can result in equipment failures, reducing the availability of charging stations and negatively impacting the EV user experience. Using Bayesian statistics and DL methods, it may be possible to tackle this challenge by enabling predictive analytics for the prediction of charging station performance. By analyzing data from sensors, performing collaborative modeling, and adjusting for data drift, AI-powered predictive analytics may be able to identify potential faults and predict when maintenance is required. This will help develop proactive maintenance practices and minimize unplanned downtime. Thus, predictive analytics using Bayesian statistics and DL methods may have the potential to improve the overall reliability of electric charging infrastructure and increase user confidence in the use of EVs.

Appendices

Appendix A Expectation-Maximization (EM) in CGMM

The EM algorithm is one common approach used to approximate the maximum likelihood estimate for the GMM. In the expectation step, the parameter values $\boldsymbol{\theta}^0$ at the current iteration are used to find the posterior distribution of the latent variable given by $P(m|\boldsymbol{X}, \boldsymbol{\theta}^0)$ [161], [162]. Then, the expected value of the complete data log-likelihood $Q(\boldsymbol{\theta}, \boldsymbol{\theta}^0)$ can be expressed in the following form,

$$Q(\boldsymbol{\theta}, \boldsymbol{\theta}^0) = E_{m|\boldsymbol{X},\boldsymbol{\theta}^0}\left[\log\left(f_{\boldsymbol{X},m}(\boldsymbol{X}', \boldsymbol{m}'|\boldsymbol{\Theta})\right)\right] \tag{A-1}$$

where,

$$f_{\boldsymbol{X},m}(\boldsymbol{X}', \boldsymbol{m}'|\boldsymbol{\Theta}) = \prod_{i=1}^{n}\prod_{k=1}^{K} \alpha_k^{I(m_i=k)} \mathcal{N}(\boldsymbol{x}_i'; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)^{I(m_i=k)} \tag{A-2}$$

Here, $I(.)$ is an indicator function. Thus, the expected value of the complete data log-likelihood can be written as,

$$\log\left(f_{\boldsymbol{X},m}(\boldsymbol{X}', \boldsymbol{m}'|\boldsymbol{\Theta})\right) = \sum_{i=1}^{n}\sum_{k=1}^{K} I(m_i = k)\left\{\log(\alpha_k) + \log\left(\mathcal{N}(\boldsymbol{x}_i'; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)\right)\right\} \tag{A-3}$$

In the maximization step, we find the maximum likelihood estimator $\widehat{\boldsymbol{\theta}}$ by solving the following optimization problem,

$$\widehat{\boldsymbol{\theta}} = \mathrm{argmax}_{\boldsymbol{\theta}} Q(\boldsymbol{\theta}, \boldsymbol{\theta}^0) \tag{A-4}$$

With further simplifications of the optimization problem in eq. (A-4) and the log-likelihood in eq. (A-3), the parameters of the GMM are calculated using the iterative EM algorithm:

Initialization step: For fair initialization, the chance of product i belonging to k^{th} cluster is given by,

$$P(m_i = k|\boldsymbol{x}_i') = q_{ik} \tag{A-5}$$

Here,

$$q_{ik} = \frac{f_{\boldsymbol{X}}(\boldsymbol{x}_i'|\boldsymbol{\mu_k}, \boldsymbol{\Sigma_k})}{\sum_{k=1}^{K} f_{\boldsymbol{X}}(\boldsymbol{x}_i'|\boldsymbol{\mu_k}, \boldsymbol{\Sigma_k})} \tag{A-6}$$

where, $\boldsymbol{\mu_k}$ is the initial mean and $\boldsymbol{\Sigma_k}$ is the covariance matrix of the k^{th} cluster given by the K-means algorithm [160]–[162].

$$\alpha_k = \frac{N_k}{N} \tag{A-7}$$

where, N_k is the number of points in k^{th} cluster given by the K-means algorithm.

Iterative expectation step:

$$q_{ik} = \frac{\alpha_k * f_{\boldsymbol{X}}(\boldsymbol{x}_i'|\boldsymbol{\mu}_i = \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_i = \boldsymbol{\Sigma}_k)}{\sum_{r=1}^{K} \alpha_r * f_{\boldsymbol{X}}(\boldsymbol{x}_i'|\boldsymbol{\mu}_i = \boldsymbol{\mu}_r, \boldsymbol{\Sigma}_i = \boldsymbol{\Sigma}_r)} \tag{A-8}$$

Iterative maximization step for α_k:

$$\alpha_k = \frac{\sum_{i=1}^{N} q_{ik}}{\sum_{k=1}^{K}\sum_{i=1}^{N} q_{ik}} \tag{A-9}$$

Iterative maximization step for $\boldsymbol{\mu}_k|\alpha_k$:

$$\boldsymbol{\mu}_k = \frac{\sum_{i=1}^{N} q_{ik}\boldsymbol{x}_i'}{\sum_{i=1}^{N} q_{ik}} \tag{A-10}$$

Iterative maximization step for $\boldsymbol{\Sigma}_k|\alpha_k, \boldsymbol{\mu}_k$:

$$\boldsymbol{\Sigma}_k = \frac{\sum_{i=1}^{N} q_{ik}(\boldsymbol{x}_i' - \boldsymbol{\mu}_k)(\boldsymbol{x}_i' - \boldsymbol{\mu}_k)'}{\sum_{i=1}^{N} q_{ik}} \tag{A-11}$$

Given that the joint distribution of $\begin{bmatrix}\boldsymbol{y}_i\\ \boldsymbol{z}_i\end{bmatrix}$ is expressed as a multivariate Gaussian distribution.

Let,

$$\boldsymbol{x}_i = \begin{bmatrix}\boldsymbol{y}_i\\ \boldsymbol{z}_i\end{bmatrix} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) \tag{A-12}$$

Here, $\boldsymbol{y_i}$ is the vector representing the number of claims for the observed TIS points, and $\boldsymbol{z_i}$ is the vector representing the number of claims for the TIS points to be forecasted. According to the theory of multivariate Gaussian Distribution, the joint Probability Density Function (PDF) for $\boldsymbol{x_i}$ is given by,

$$f_{\boldsymbol{Y},\boldsymbol{Z}}(\boldsymbol{y}_i,\boldsymbol{z}_i) = \frac{1}{(2\pi)^{\mathrm{T}/2|\boldsymbol{\Sigma}|^{1/2}}} \exp\left[-\frac{1}{2}Q(\boldsymbol{y}_i,\boldsymbol{z}_i)\right] \tag{A-13}$$

where $Q(\boldsymbol{y_i},\boldsymbol{z_i})$ is defined as,

$$Q(\boldsymbol{y}_i,\boldsymbol{z}_i) = (\boldsymbol{x}_i - \boldsymbol{\mu_i})'\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}_i - \boldsymbol{\mu_i}) \tag{A-14}$$

$$\begin{gathered} Q(\boldsymbol{y}_i,\boldsymbol{z}_i) = \left(\boldsymbol{y}_i - \boldsymbol{\mu}_{\boldsymbol{y}_i}\right)'\boldsymbol{\Sigma}^{\boldsymbol{y},\boldsymbol{y}}\left(\boldsymbol{y}_i - \boldsymbol{\mu}_{\boldsymbol{y}_i}\right) + 2\left(\boldsymbol{y}_i - \boldsymbol{\mu}_{\boldsymbol{y}_i}\right)'\boldsymbol{\Sigma}^{\boldsymbol{y},\boldsymbol{z}}\left(\boldsymbol{z}_i - \boldsymbol{\mu}_{\boldsymbol{z}_i}\right) + \\ \left(\boldsymbol{z}_i - \boldsymbol{\mu}_{\boldsymbol{z}_i}\right)'\boldsymbol{\Sigma}^{\boldsymbol{z},\boldsymbol{z}}\left(\boldsymbol{z}_i - \boldsymbol{\mu}_{\boldsymbol{z}_i}\right) \end{gathered} \tag{A-15}$$

Here, it is assumed that,

$$\boldsymbol{\Sigma}^{-1} = \begin{bmatrix} \boldsymbol{\Sigma}_{\boldsymbol{y},\boldsymbol{y}} & \boldsymbol{\Sigma}_{\boldsymbol{y},\boldsymbol{z}} \\ \boldsymbol{\Sigma}_{\boldsymbol{z},\boldsymbol{y}} & \boldsymbol{\Sigma}_{\boldsymbol{z},\boldsymbol{z}} \end{bmatrix}^{-1} = \begin{bmatrix} \boldsymbol{\Sigma}^{\boldsymbol{y},\boldsymbol{y}} & \boldsymbol{\Sigma}^{\boldsymbol{y},\boldsymbol{z}} \\ \boldsymbol{\Sigma}^{\boldsymbol{z},\boldsymbol{y}} & \boldsymbol{\Sigma}^{\boldsymbol{z},\boldsymbol{z}} \end{bmatrix} \tag{A-16}$$

Let,

$$\boldsymbol{\Sigma}^{\boldsymbol{y},\boldsymbol{y}} = \left(\boldsymbol{\Sigma}_{\boldsymbol{y},\boldsymbol{y}} - \boldsymbol{\Sigma}_{\boldsymbol{y},\boldsymbol{z}}\boldsymbol{\Sigma}_{\boldsymbol{z},\boldsymbol{z}}^{-1}\boldsymbol{\Sigma}_{\boldsymbol{y},\boldsymbol{z}}'\right)^{-1} \tag{A-17}$$

$$\boldsymbol{\Sigma}^{\boldsymbol{z},\boldsymbol{z}} = \left(\boldsymbol{\Sigma}_{\boldsymbol{z},\boldsymbol{z}} - \boldsymbol{\Sigma}_{\boldsymbol{y},\boldsymbol{z}}'\boldsymbol{\Sigma}_{\boldsymbol{y},\boldsymbol{y}}^{-1}\boldsymbol{\Sigma}_{\boldsymbol{y},\boldsymbol{z}}\right)^{-1} \tag{A-18}$$

$$\boldsymbol{\Sigma}^{\boldsymbol{y},\boldsymbol{z}} = (\boldsymbol{\Sigma}^{\boldsymbol{z},\boldsymbol{y}})' \ = -\boldsymbol{\Sigma}_{\boldsymbol{y},\boldsymbol{y}}^{-1}\boldsymbol{\Sigma}_{\boldsymbol{y},\boldsymbol{z}}\left(\boldsymbol{\Sigma}_{\boldsymbol{z},\boldsymbol{z}} - \boldsymbol{\Sigma}_{\boldsymbol{y},\boldsymbol{z}}'\boldsymbol{\Sigma}_{\boldsymbol{y},\boldsymbol{y}}^{-1}\boldsymbol{\Sigma}_{\boldsymbol{y},\boldsymbol{z}}\right)^{-1} \tag{A-19}$$

On substituting eq. (A-16)-(A-19) in eq. (A-15), we get,

$$\begin{gathered} Q(\boldsymbol{y}_i,\boldsymbol{z}_i) = \ \left(\boldsymbol{y}_i - \boldsymbol{\mu}_{\boldsymbol{y}}\right)'\boldsymbol{\Sigma}_{\boldsymbol{y},\boldsymbol{y}}^{-1}\left(\boldsymbol{y}_i - \boldsymbol{\mu}_{\boldsymbol{y}}\right) + \left[(\boldsymbol{z}_i - \boldsymbol{\mu}_{\boldsymbol{z}}) - \boldsymbol{\Sigma}_{\boldsymbol{y},\boldsymbol{z}}'\boldsymbol{\Sigma}_{\boldsymbol{y},\boldsymbol{y}}^{-1}\left(\boldsymbol{y}_i - \boldsymbol{\mu}_{\boldsymbol{y}}\right)\right]' \\ \left(\boldsymbol{\Sigma}_{\boldsymbol{z},\boldsymbol{z}} - \boldsymbol{\Sigma}_{\boldsymbol{y},\boldsymbol{z}}'\boldsymbol{\Sigma}_{\boldsymbol{y},\boldsymbol{y}}^{-1}\boldsymbol{\Sigma}_{\boldsymbol{y},\boldsymbol{z}}\right)^{-1}\left[(\boldsymbol{z}_i - \boldsymbol{\mu}_{\boldsymbol{z}}) - \boldsymbol{\Sigma}_{\boldsymbol{y},\boldsymbol{z}}'\boldsymbol{\Sigma}_{\boldsymbol{y},\boldsymbol{y}}^{-1}\left(\boldsymbol{z}_i - \boldsymbol{\mu}_{\boldsymbol{y}}\right)\right] \end{gathered} \tag{A-20}$$

Let,

$$\boldsymbol{b}_i \triangleq \ \boldsymbol{\mu}_{\boldsymbol{z}} + \ \boldsymbol{\Sigma}_{\boldsymbol{y},\boldsymbol{z}}'\boldsymbol{\Sigma}_{\boldsymbol{y},\boldsymbol{y}}^{-1}\left(\boldsymbol{y}_i - \boldsymbol{\mu}_{\boldsymbol{y}}\right) \tag{A-21}$$

$$\boldsymbol{A} \triangleq \ \boldsymbol{\Sigma}_{\boldsymbol{z},\boldsymbol{z}} - \boldsymbol{\Sigma}_{\boldsymbol{y},\boldsymbol{z}}'\boldsymbol{\Sigma}_{\boldsymbol{y},\boldsymbol{y}}^{-1}\boldsymbol{\Sigma}_{\boldsymbol{y},\boldsymbol{z}} \tag{A-22}$$

$$Q_{\boldsymbol{y}_i}(\boldsymbol{y}_i) \triangleq \left(\boldsymbol{y}_i - \boldsymbol{\mu}_{\boldsymbol{y}}\right)'\boldsymbol{\Sigma}_{\boldsymbol{y},\boldsymbol{y}}^{-1}\left(\boldsymbol{y}_i - \boldsymbol{\mu}_{\boldsymbol{y}}\right) \tag{A-23}$$

$$\begin{gathered} Q_{\boldsymbol{z}_i}(\boldsymbol{y}_i,\boldsymbol{z}_i) \triangleq \left[(\boldsymbol{z}_i - \boldsymbol{\mu}_{\boldsymbol{z}}) - \boldsymbol{\Sigma}_{\boldsymbol{y},\boldsymbol{z}}'\boldsymbol{\Sigma}_{\boldsymbol{y},\boldsymbol{y}}^{-1}\left(\boldsymbol{y}_i - \boldsymbol{\mu}_{\boldsymbol{y}}\right)\right]'\left(\boldsymbol{\Sigma}_{\boldsymbol{z},\boldsymbol{z}} - \boldsymbol{\Sigma}_{\boldsymbol{y},\boldsymbol{z}}'\boldsymbol{\Sigma}_{\boldsymbol{y},\boldsymbol{y}}^{-1}\boldsymbol{\Sigma}_{\boldsymbol{y},\boldsymbol{z}}\right)^{-1} \\ \left[(\boldsymbol{z}_i - \boldsymbol{\mu}_{\boldsymbol{z}}) - \boldsymbol{\Sigma}_{\boldsymbol{y},\boldsymbol{z}}'\boldsymbol{\Sigma}_{\boldsymbol{y},\boldsymbol{y}}^{-1}\left(\boldsymbol{y}_i - \boldsymbol{\mu}_{\boldsymbol{y}}\right)\right] \end{gathered} \tag{A-24}$$

Thus,

$$Q_{\boldsymbol{z}_i}(\boldsymbol{y}_i, \boldsymbol{z}_i) = (\boldsymbol{z}_i - \boldsymbol{b}_i)'\boldsymbol{A}^{-1}(\boldsymbol{z}_i - \boldsymbol{b}_i) \tag{A-25}$$

$$Q(\boldsymbol{y}_i, \boldsymbol{z}_i) = \ Q_{\boldsymbol{y}_i}(\boldsymbol{y}_i) + \ Q_{\boldsymbol{z}_i}(\boldsymbol{y}_i, \boldsymbol{z}_i) \tag{A-26}$$

Using eq. (A-25) and eq. (A-26) the joint distribution function can now be written as,

$$f_{\boldsymbol{Z}|\boldsymbol{Y}}(\boldsymbol{z}_i|\boldsymbol{y}_i) = N\left(\boldsymbol{y}_i; \boldsymbol{\mu}_y, \boldsymbol{\Sigma}_{y,y}\right)\mathcal{N}(\boldsymbol{z}_i; \boldsymbol{b}_i, \boldsymbol{A}) \tag{A-27}$$

The conditional distribution of $\boldsymbol{z}_i$ given $\boldsymbol{y}_i$ is thus given by,

$$f_{\boldsymbol{Z}|\boldsymbol{Y}}(\boldsymbol{z}_i|\boldsymbol{y}_i) = \frac{f_{\boldsymbol{Y},\boldsymbol{Z}}(\boldsymbol{y}_i, \boldsymbol{z}_i)}{f_{\boldsymbol{Y}}(\boldsymbol{y}_i)} = \mathcal{N}(\boldsymbol{z}_i|\boldsymbol{y}_i; \boldsymbol{b}_i, \boldsymbol{A}) \tag{A-28}$$

Extending eq. (A-28) to the GMM with *K* clusters where $\begin{bmatrix}\boldsymbol{y}_i \\ \boldsymbol{z}_i\end{bmatrix}$ is expressed as a mixture of multivariate Gaussian distribution based on the GMM we get,

$$f_{\boldsymbol{Z}|\boldsymbol{Y}}(\boldsymbol{z}_i|\boldsymbol{y}_i) = \sum_{k=1}^{K}(\alpha_k|\boldsymbol{y}_i)\,\mathcal{N}\left(\boldsymbol{z}_i|\boldsymbol{y}_i; \boldsymbol{b}_{i,k}, \boldsymbol{A}_k\right) \tag{A-29}$$

where,

$$\alpha_k|\boldsymbol{y}_i = \frac{\alpha_k * f_{\boldsymbol{Y}}(\boldsymbol{y}_i|\boldsymbol{\mu}_k^{\boldsymbol{y}}, \boldsymbol{\Sigma}_k^{\boldsymbol{y}})}{\sum_{r=1}^{K}\left\{\alpha_r * f_{\boldsymbol{Y}}(\boldsymbol{y}_i|\boldsymbol{\mu}_r^{\boldsymbol{y}}, \boldsymbol{\Sigma}_r^{\boldsymbol{y}})\right\}} \tag{A-30}$$

$$\boldsymbol{b}_{i,k} = \ \boldsymbol{\mu}_k^{\boldsymbol{z}}|\boldsymbol{y}_i = \boldsymbol{\mu}_k^{\boldsymbol{z}} + \left(\boldsymbol{\Sigma}_k^{\boldsymbol{y},\boldsymbol{z}}\right)'\left(\boldsymbol{\Sigma}_k^{\boldsymbol{y}}\right)^{-1}\left(\boldsymbol{y}_i - \boldsymbol{\mu}_k^{\boldsymbol{y}}\right) \tag{A-31}$$

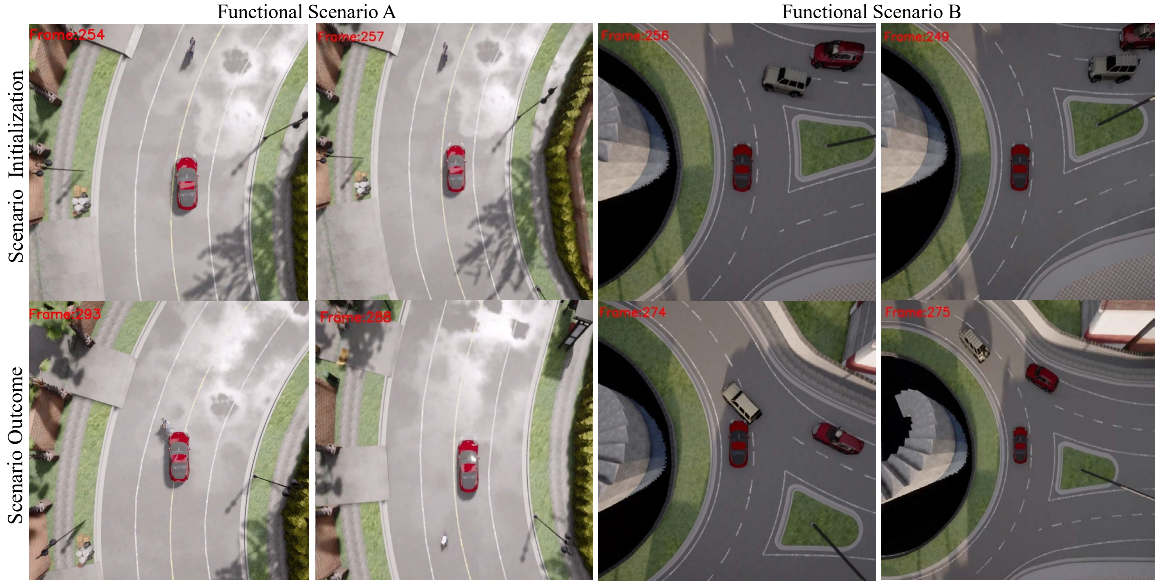
$$\boldsymbol{A}_k = \boldsymbol{\Sigma}_k^{\boldsymbol{z}}|\boldsymbol{y}_i = \boldsymbol{\Sigma}_k^{\boldsymbol{z}} - \left(\boldsymbol{\Sigma}_k^{\boldsymbol{y},\boldsymbol{z}}\right)'\left(\boldsymbol{\Sigma}_k^{\boldsymbol{y}}\right)^{-1}\left(\boldsymbol{\Sigma}_k^{\boldsymbol{y},\boldsymbol{z}}\right) \tag{A-32}$$

Appendix B Experimental Evaluation of BOSE

In our experiments, BOSE is able to rightly identify and tag risky concrete scenarios with a high probability of ‘crash’ $p(y = 1)$. Figure B-1 shows examples of concrete scenarios and their predicted outcomes for functional Scenario A and Scenario B.

In Figure B-1 (a), while the ego vehicle is traveling at 36 KPH, the ADS Proximity Threshold is set to 12 meters. The Bayesian DNN within BOSE is able to identify this risky combination and tag this scenario with a high probability of ‘crash’ $(p(y = 1) = 94\%)$. On the other hand, in Figure B-1 (b), while the ego vehicle is traveling at 31 KPH, the ADS Proximity Threshold is set to 18 meters. Thus, the ADS has sufficient time to identify the oncoming motorcyclist and correct any erroneous behavior. BOSE rightly identifies this interaction and tags this scenario as ‘normal’ with a low probability of ‘crash’ $(p(y = 1) = 16\%)$. Figure 4-6 can be used to visualize the decision boundaries established by BOSE given functional Scenario A. In Figure B-1 (c), while the ego vehicle is traveling at 44 KPH, v_1 and v_2 are set to 35 and 36 KPH, respectively. Thus, the ADS has very little time to respond to the yield violations performed by the other two vehicles. Again, BOSE is able to identify this risky combination and tag this scenario with a high probability of ‘crash’ $(p(y = 1) = 88\%)$. On the other hand, in Figure B-1 (d) while the ego vehicle is traveling at 31 KPH, v_1, and v_2 are set to 18 and 22 KPH respectively. Thus, the ADS has sufficient time to identify the yield violations and apply emergency braking, thus avoiding a ‘crash’. Although complicated by the remaining factors, BOSE rightly identifies this interaction and tags this scenario as ‘normal’ with a low probability of ‘crash’ $(p(y = 1) =$

11%). Figure 4-9 can be used to visualize the risky clusters established by BOSE given functional Scenario B.



(a)
True Outcome: Crash
Predicted Crash Probability: 94%

(b)
True Outcome: Normal
Predicted Crash Probability: 16%

(c)
True Outcome: Crash
Predicted Crash Probability: 88%

(d)
True Outcome: Normal
Predicted Crash Probability: 11%

Figure B-1 Sample concrete scenarios and their predicted crash probability by BOSE for functional Scenario A and Scenario B from their respective test sets. Here, BOSE accurately predicts the scenario outcome tagging the true positives with a high probability crash.

Bibliography

[1] W. W. Eckerson, “Predictive analytics,” *Extending the Value of Your Data Warehousing Investment. TDWI Best Practices Report*, vol. 1, pp. 1–36, 2007.

[2] D. T. Larose, *Data mining and predictive analytics*. John Wiley & Sons, 2015.

[3] V. Kumar and M. L. Garg, “Predictive analytics: a review of trends and techniques,” *International Journal of Computer Applications*, vol. 182, no. 1, pp. 31–37, 2018.

[4] J. Fitz-Enz and I. I. John Mattox, *Predictive analytics for human resources*. John Wiley & Sons, 2014.

[5] N. Mishra and S. Silakari, “Predictive analytics: a survey, trends, applications, oppurtunities & challenges,” *International Journal of Computer Science and Information Technologies*, vol. 3, no. 3, pp. 4434–4438, 2012.

[6] H. V. Haghi, S. Lotfifard, and Z. Qu, “Multivariate predictive analytics of wind power data for robust control of energy storage,” *IEEE Transactions on Industrial Informatics*, vol. 12, no. 4, pp. 1350–1360, 2016.

[7] R. B. Parikh, Z. Obermeyer, and A. S. Navathe, “Regulation of predictive analytics in medicine,” *Science*, vol. 363, no. 6429, pp. 810–812, 2019.

[8] P. Ongsulee, V. Chotchaung, E. Bamrungsi, and T. Rodcheewit, “Big data, predictive analytics and machine learning,” in *2018 16th international conference on ICT and knowledge engineering (ICT&KE)*, 2018, pp. 1–6.

[9] C. Bravo *et al.*, “State of the art of artificial intelligence and predictive analytics in the E&P industry: a technology survey,” *Spe Journal*, vol. 19, no. 04, pp. 547–563, 2014.

[10] T. H. Davenport, “From analytics to artificial intelligence,” *Journal of Business Analytics*, vol. 1, no. 2, pp. 73–80, 2018.

[11] A. Schweyer, “Predictive analytics and artificial intelligence in people management,” *Incentive Research Foundation*, pp. 1–18, 2018.

[12] M. Wollowski *et al.*, “A survey of current practice and teaching of AI,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2016, vol. 30, no. 1.

[13] A. He *et al.*, “A survey of artificial intelligence for cognitive radios,” *IEEE transactions on*

vehicular technology, vol. 59, no. 4, pp. 1578–1592, 2010.

[14] J. Rowe and D. Partridge, “Creativity: A survey of AI approaches,” *Artificial Intelligence Review*, vol. 7, pp. 43–70, 1993.

[15] Y. Roh, G. Heo, and S. E. Whang, “A survey on data collection for machine learning: a big data-ai integration perspective,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 33, no. 4, pp. 1328–1347, 2019.

[16] Z. Allam and Z. A. Dhunny, “On big data, artificial intelligence and smart cities,” *Cities*, vol. 89, pp. 80–91, 2019.

[17] Y. Zhuang, F. Wu, C. Chen, and Y. Pan, “Challenges and opportunities: from big data to knowledge in AI 2.0,” *Frontiers of Information Technology & Electronic Engineering*, vol. 18, pp. 3–14, 2017.

[18] M. Savargaonkar, A. Chehade, and A. A. Hussein, “A Novel Neural Network with Gaussian Process Feedback for Modeling the State-of-Charge of Battery Cells,” *IEEE Transactions on Industry Applications*, 2022.

[19] E. Chemali, P. J. Kollmeyer, M. Preindl, and A. Emadi, “State-of-charge estimation of Li-ion batteries using deep neural networks: A machine learning approach,” *Journal of Power Sources*, vol. 400, pp. 242–255, 2018.

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

[21] A. Akbarov and S. Wu, “Warranty claim forecasting based on weighted maximum likelihood estimation,” *Quality and Reliability Engineering International*, vol. 28, no. 6, pp. 663–669, 2012.

[22] B. Gangopadhyay, S. Khastgir, S. Dey, P. Dasgupta, G. Montana, and P. Jennings, “Identification of test cases for automated driving systems using bayesian optimization,” in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, 2019, pp. 1961–1967.

[23] T. Fields, G. Hsieh, and J. Chenou, “Mitigating drift in time series data with noise augmentation,” in *2019 International Conference on Computational Science and Computational Intelligence (CSCI)*, 2019, pp. 227–230.

[24] D. E. A. Kleyner, “Warranty Data Maturity Patterns: Knowing When Your Data is Ready,” in *Applied Reliability & Durability Conference, Indianapolis, IN.*, 2019.

[25] N. Ståhl, G. Falkman, A. Karlsson, and G. Mathiason, “Evaluation of uncertainty quantification in deep learning,” in *Information Processing and Management of Uncertainty in Knowledge-Based Systems: 18th International Conference, IPMU 2020, Lisbon, Portugal, June 15–19, 2020, Proceedings, Part I 18*, 2020, pp. 556–568.

[26] A. K. Saberi, J. Hegge, T. Fruehling, and J. F. Groote, “Beyond sotif: Black swans and formal methods,” *SYSCON 2020 - 14th Annual IEEE International Systems Conference, Proceedings*, pp. 1–5, 2020.

[27] K. Chen, K. Chen, Q. Wang, Z. He, J. Hu, and J. He, “Short-Term Load Forecasting with Deep Residual Networks,” *IEEE Transactions on Smart Grid*, vol. 10, no. 4, pp. 3943–3952, 2019.

[28] Z. Saigol and A. Peters, “Verifying automated driving systems in simulation: framework and challenges,” *25 th ITS World Congress*, no. September, pp. 17–21, 2018.

[29] C. H. Lauro, R. B. D. Pereira, L. C. Brandão, and J. P. Davim, “Design of Experiments—Statistical and artificial intelligence analysis for the improvement of machining processes: A review,” *Design of experiments in production engineering*, pp. 89–107, 2015.

[30] S. Darbha, S. Konduri, and P. R. Pagilla, “Benefits of V2V communication for autonomous and connected vehicles,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 5, pp. 1954–1963, 2018.

[31] Y. Li *et al.*, “V2X-Sim: Multi-agent collaborative perception dataset and benchmark for autonomous driving,” *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 10914–10921, 2022.

[32] R. Weber, J. Misener, and V. Park, “C-V2X-A Communication Technology for Cooperative, Connected and Automated Mobility,” in *Mobile Communication-Technologies and Applications; 24. ITG-Symposium*, 2019, pp. 1–6.

[33] C. Jung, D. Lee, S. Lee, and D. H. Shim, “V2X-communication-aided autonomous driving: system design and experimental validation,” *Sensors*, vol. 20, no. 10, p. 2903, 2020.

[34] L. Rossi, M. Paolanti, R. Pierdicca, and E. Frontoni, “Human trajectory prediction and generation using LSTM models and GANs,” *Pattern Recognition*, vol. 120, p. 108136, 2021.

[35] L. Rossi, A. Ajmar, M. Paolanti, and R. Pierdicca, “Vehicle trajectory prediction and generation using LSTM models and GANs,” *Plos one*, vol. 16, no. 7, p. e0253868, 2021.

[36] S. Vasavi, K. Aswarth, T. S. D. Pavan, and A. A. Gokhale, “Predictive analytics as a service for vehicle health monitoring using edge computing and AK-NN algorithm,” *Materials Today: Proceedings*, vol. 46, pp. 8645–8654, 2021.

[37] A. A. Chehade, *Data-driven Approaches for Condition Monitoring and Predictive Analytics*. The University of Wisconsin-Madison, 2017.

[38] C. Merrefield, “Transportation in the Age of Artificial Intelligence and Predictive Analytics,” John A. Volpe National Transportation Systems Center (US), 2019.

[39] S. Grigorescu, B. Trasnea, T. Cocias, and G. Macesanu, “A survey of deep learning techniques for autonomous driving,” *Journal of Field Robotics*, vol. 37, no. 3, pp. 362–386, 2020.

[40] M. Savargaonkar, A. Chehade, Z. Shi, and A. A. Hussein, "A Cycle-based Recurrent Neural Network for State-of-Charge Estimation of Li-ion Battery Cells," in *2020 IEEE Transportation Electrification Conference & Expo (ITEC)*, 2020, pp. 584–587.

[41] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[42] M. Savargaonkar, I. Oyewole, and A. Chehade, "Sparse Autoencoded Long Short-Term Memory Network for State-of-Charge Estimations," in *2021 IEEE Transportation Electrification Conference & Expo (ITEC)*, 2021, pp. 474–478.

[43] C. Vidal, P. Kollmeyer, E. Chemali, and A. Emadi, "Li-ion Battery State of Charge Estimation Using Long Short-Term Memory Recurrent Neural Network with Transfer Learning," in *ITEC 2019 - 2019 IEEE Transportation Electrification Conference and Expo*, 2019.

[44] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on knowledge and data engineering*, vol. 22, no. 10, pp. 1345–1359, 2009.

[45] M. Danko, J. Adamec, M. Taraba, and P. Drgona, "Overview of batteries State of Charge estimation methods," *Transportation Research Procedia*, vol. 40, pp. 186–192, 2019.

[46] A. A. Chehade and A. A. Hussein, "A Multi-Output Convolved Gaussian Process Model for Capacity Estimation of Electric Vehicle Li-ion Battery Cells," 2019, pp. 1–4.

[47] W. Xie, L. Shen, and Y. Zhong, "Two-dimensional aggregate warranty demand forecasting under sales uncertainty," *IISE Transactions*, vol. 49, no. 5, pp. 553–565, 2017.

[48] J. D. Kalbfleisch, J. F. Lawless, and J. A. Robinson, "Methods for the analysis and prediction of warranty claims," *Technometrics*, vol. 33, no. 3, pp. 273–285, 1991.

[49] H. G. Teo, "Data mining in automotive warranty analysis," 2010.

[50] S. Wu, "Warranty data analysis: A review," *Quality and Reliability Engineering International*, vol. 28, no. 8, pp. 795–805, 2012.

[51] J. A. Sanguesa, V. Torres-Sanz, P. Garrido, F. J. Martinez, and J. M. Marquez-Barja, "A review on electric vehicles: Technologies and challenges," *Smart Cities*, vol. 4, no. 1, pp. 372–404, 2021.

[52] A. Chehade, M. Savargaonkar, and V. Krivtsov, "Conditional Gaussian mixture model for warranty claims forecasting," *Reliability Engineering & System Safety*, vol. 218, p. 108180, 2022.

[53] V. V Krivtsov, "Field Data Analysis & Statistical Warranty Forecasting," *IEEE Catalog No CFP11RAM-CDR*, 2011.

[54] M. Modarres, M. P. Kaminskiy, and V. Krivtsov, *Reliability engineering and risk analysis: a practical guide*, 3rd ed. CRC press, 2016.

[55] R. Lattarulo, J. Pérez, and M. Dendaluce, “A complete framework for developing and testing automated driving controllers,” *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 258–263, 2017.

[56] A. Broggi, A. Zelinsky, Ü. Özgüner, and C. Laugier, “Intelligent vehicles,” in *Springer Handbook of Robotics*, Springer, 2016, pp. 1627–1656.

[57] S. W. Loke, “Cooperative automated vehicles: A review of opportunities and challenges in socially intelligent vehicles beyond networking,” *IEEE Transactions on Intelligent Vehicles*, vol. 4, no. 4, pp. 509–518, 2019.

[58] R. Chandra, U. Bhattacharya, A. Bera, and Di. Manocha, “Traphic: Trajectory prediction in dense and heterogeneous traffic using weighted interactions,” *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2019-June, pp. 8475–8484, 2019.

[59] M. Borg *et al.*, “Safely Entering the Deep: A Review of Verification and Validation for Machine Learning and a Challenge Elicitation in the Automotive Industry,” *arXiv*, 2018.

[60] F. Zheng *et al.*, “Unlimited Neighborhood Interaction for Heterogeneous Trajectory Prediction,” pp. 13168–13177, 2021.

[61] H. Saleem, F. Riaz, L. Mostarda, M. A. Niazi, A. Rafiq, and S. Saeed, “Steering Angle Prediction Techniques for Autonomous Ground Vehicles: A Review,” *IEEE Access*, vol. 9, pp. 78567–78585, 2021.

[62] Q. Zhang, M. Zhang, T. Chen, Z. Sun, Y. Ma, and B. Yu, “Recent advances in convolutional neural network acceleration,” *Neurocomputing*, vol. 323, pp. 37–51, 2019.

[63] A. Wang, X. Huang, A. Jasour, and B. Williams, “Fast Risk Assessment for Autonomous Vehicles Using Learned Models of Agent Futures,” 2020.

[64] L. Wang, T. Wu, H. Fu, L. Xiao, Z. Wang, and B. Dai, “Prediction for Autonomous Driving,” vol. 6, no. 3, pp. 1–8, 2021.

[65] J. Amirian, J.-B. Hayet, and J. Pettré, “Social ways: Learning multi-modal distributions of pedestrian trajectories with gans,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2019, p. 0.

[66] R. P. Wildes, “Where are you heading ? Dynamic Trajectory Prediction with Expert Goal Examples,” pp. 7629–7638.

[67] A. Sadeghian, V. Kosaraju, A. Sadeghian, N. Hirose, H. Rezatofighi, and S. Savarese, “Sophie: An attentive gan for predicting paths compliant to social and physical constraints,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 1349–1358.

[68] J. Gu, C. Sun, and H. Zhao, “DenseTNT: End-to-end Trajectory Prediction from Dense Goal Sets,” pp. 15303–15312, 2021.

[69] J. Houston *et al.*, “One Thousand and One Hours: Self-driving Motion Prediction Dataset,” no. CoRL 2020, pp. 1–10, 2020.

[70] N. Gao *et al.*, “Generative adversarial networks for spatio-temporal data: A survey,” *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 13, no. 2, pp. 1–25, 2022.

[71] D. Saxena and J. Cao, “Generative adversarial networks (GANs) challenges, solutions, and future directions,” *ACM Computing Surveys (CSUR)*, vol. 54, no. 3, pp. 1–42, 2021.

[72] D. Roy, T. Ishizaka, C. K. Mohan, and A. Fukuda, “Vehicle trajectory prediction at intersections using interaction based generative adversarial networks,” in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, 2019, pp. 2318–2323.

[73] Y. Yuan, X. Weng, Y. Ou, and K. M. Kitani, “Agentformer: Agent-aware transformers for socio-temporal multi-agent forecasting,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 9813–9823.

[74] S. Eiffert, K. Li, M. Shan, S. Worrall, S. Sukkarieh, and E. Nebot, “Probabilistic crowd GAN: Multimodal pedestrian trajectory prediction using a graph vehicle-pedestrian attention network,” *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 5026–5033, 2020.

[75] H. Hexmoor, J. Lammens, G. Caicedo, and S. C. Shapiro, *Behaviour based AI, cognitive processes, and emergent behaviors in autonomous agents*, vol. 1. WIT Press, 1970.

[76] Z. Li, C. H. Sim, and M. Y. H. Low, “A survey of emergent behavior and its impacts in agent-based systems,” in *2006 4th IEEE international conference on industrial informatics*, 2006, pp. 1295–1300.

[77] L. Steels, “The artificial life roots of artificial intelligence,” *Artificial life*, vol. 1, no. 1_2, pp. 75–110, 1993.

[78] N. Rajabli, F. Flammini, R. Nardone, and V. Vittorini, “Software verification and validation of safe autonomous cars: A systematic literature review,” *IEEE Access*, vol. 9, pp. 4797–4819, 2020.

[79] N. Rajabli, F. Flammini, R. Nardone, and V. Vittorini, “Software Verification and Validation of Safe Autonomous Cars: A Systematic Literature Review,” *IEEE Access*, pp. 4797–4819, 2020.

[80] A. Abdulazim, M. Elbahaey, and A. Mohamed, “Putting Safety of Intended Functionality SOTIF into Practice,” *SAE Technical Paper Series*, vol. 1, no. 1, pp. 1–11, 2021.

[81] J. J. Valls, M. García-gordillo, I. T. De Informática, and S. Sáez, “Scenario-based Validation & Verification : The ENABLE-S3 Approach,” vol. 40, no. 1, pp. 79–84, 2020.

[82] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, “DeepFace: Closing the gap to human-level performance in face verification,” *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 1701–1708, 2014.

[83] M. Safar, M. A. El-Moursy, M. Abdelsalam, A. Bakr, K. Khalil, and A. Salem, “Virtual verification and validation of automotive system,” *Journal of Circuits, Systems and Computers*, vol. 28, no. 04, p. 1950071, 2019.

[84] M. A. Niazi, A. Hussain, and M. Kolberg, “Verification & validation of agent based simulations using the VOMAS (virtual overlay multi-agent system) approach,” *arXiv preprint arXiv:1708.02361*, 2017.

[85] C. Neurohr, L. Westhofen, M. Butz, M. Bollmann, U. Eberle, and R. Galbas, “Criticality Analysis for the Verification and Validation of Automated Vehicles,” *IEEE Access*, vol. 9, no. i, 2021.

[86] D. A. Reynolds, T. F. Quatieri, and R. B. Dunn, “Speaker verification using adapted Gaussian mixture models,” *Digital signal processing*, vol. 10, no. 1–3, pp. 19–41, 2000.

[87] M. Arief *et al.*, “Deep probabilistic accelerated evaluation: A certifiable rare-event simulation methodology for black-box autonomy,” *arXiv*, vol. 130, 2020.

[88] Y. Ma, Z. Wang, H. Yang, and L. Yang, “Artificial intelligence applications in the development of autonomous vehicles: A survey,” *IEEE/CAA Journal of Automatica Sinica*, vol. 7, no. 2, pp. 315–329, 2020.

[89] M. Savargaonkar and A. A. Chehade, “VTrackIt: A Synthetic Self-Driving Dataset with Infrastructure and Pooled Vehicle Information,” *arXiv preprint arXiv:2207.11146*, 2022.

[90] P. Ren *et al.*, “A survey of deep active learning,” *ACM computing surveys (CSUR)*, vol. 54, no. 9, pp. 1–40, 2021.

[91] K. Smith, A. Saxon, M. Keyser, B. Lundstrom, Z. Cao, and A. Roc, “Life prediction model for grid-connected Li-ion battery energy storage system,” in *2017 American Control Conference (ACC)*, 2017, pp. 4062–4068.

[92] B. Wang, M. Xu, and L. Yang, “Study on the economic and environmental benefits of different EV powertrain topologies,” *Energy Conversion and Management*, vol. 86, pp. 916–926, 2014.

[93] R. Xiong, J. Cao, Q. Yu, H. He, and F. Sun, “Critical review on the battery state of charge estimation methods for electric vehicles,” *Ieee Access*, vol. 6, pp. 1832–1843, 2017.

[94] M. A. Hannan, M. S. H. Lipu, A. Hussain, and A. Mohamed, “A review of lithium-ion battery state of charge estimation and management system in electric vehicle applications: Challenges and recommendations,” *Renewable and Sustainable Energy Reviews*, vol. 78, pp. 834–854, 2017.

[95] Y. Zheng, M. Ouyang, X. Han, L. Lu, and J. Li, “Investigating the error sources of the online state of charge estimation methods for lithium-ion batteries in electric vehicles,” *Journal of Power Sources*, vol. 377, pp. 161–188, 2018.

[96] S. Rodrigues, N. Munichandraiah, and A. K. Shukla, “A review of state-of-charge indication of

batteries by means of ac impedance measurements," *Journal of power Sources*, vol. 87, no. 1–2, pp. 12–20, 2000.

[97] X. Hu, F. Sun, and Y. Zou, "Comparison between two model-based algorithms for Li-ion battery SOC estimation in electric vehicles," *Simulation Modelling Practice and Theory*, vol. 34, pp. 1–11, 2013.

[98] W.-Y. Chang, "The State of Charge Estimating Methods for Battery: A Review," *ISRN Applied Mathematics*, vol. 2013, no. 1, pp. 1–7, 2013.

[99] F. Zheng, Y. Xing, J. Jiang, B. Sun, J. Kim, and M. Pecht, "Influence of different open circuit voltage tests on state of charge online estimation for lithium-ion batteries," *Applied energy*, vol. 183, pp. 513–525, 2016.

[100] H. He, R. Xiong, X. Zhang, F. Sun, and J. Fan, "State-of-charge estimation of the lithium-ion battery using an adaptive extended Kalman filter based on an improved Thevenin model," IEEE Transactions on vehicular technology, vol. 60, no. 4, pp. 1461–1469, 2011.

[101] W. He, N. Williard, C. Chen, and M. Pecht, "State of charge estimation for Li-ion batteries using neural network modeling and unscented Kalman filter-based error cancellation," International Journal of Electrical Power & Energy Systems, vol. 62, pp. 783–791, 2014.

[102] D. Sun, X. Yu, C. Zhang, C. Wang, and R. Huang, "State of charge estimation for lithium-ion battery based on an intelligent adaptive unscented Kalman filter," International Journal of Energy Research, vol. 44, no. 14, pp. 11199–11218, 2020.

[103] M. Wei, M. Ye, J. B. Li, Q. Wang, and X. Xu, "State of Charge Estimation of Lithium-Ion Batteries Using LSTM and NARX Neural Networks," IEEE Access, vol. 8, pp. 189236–189245, 2020.

[104] S. Shen, M. Sadoughi, X. Chen, M. Hong, and C. Hu, "A deep learning method for online capacity estimation of lithium-ion batteries," Journal of Energy Storage, vol. 25, p. 100817, 2019.

[105] M. A. Hannan, D. N. T. How, M. Mansor, M. S. H. Lipu, P. J. Ker, and K. M. Muttaqi, "State-of-Charge Estimation of Li-ion Battery at Variable Ambient Temperature with Gated Recurrent Unit Network," in *2020 IEEE Industry Applications Society Annual Meeting*, 2020, pp. 1–8.

[106] J. Tian, R. Xiong, W. Shen, and J. Lu, "State-of-charge estimation of LiFePO4 batteries in electric vehicles: A deep-learning enabled approach," *Applied Energy*, vol. 291, p. 116812, 2021.

[107] M. Savargaonkar and A. Chehade, "An Adaptive Deep Neural Network with Transfer Learning for State-of-Charge Estimations of Battery Cells," in *2020 IEEE Transportation Electrification Conference & Expo (ITEC)*, 2020, pp. 598–602.

[108] M. Lucu *et al.*, "Data-driven nonparametric Li-ion battery ageing model aiming at learning from real operation data-Part B: Cycling operation," *Journal of Energy Storage*, vol. 30, p. 101410,

2020.

[109] M. Lucu *et al.*, "Data-driven nonparametric Li-ion battery ageing model aiming at learning from real operation data–Part A: Storage operation," *Journal of Energy Storage*, vol. 30, p. 101409, 2020.

[110] M. R. Palacín, "Understanding ageing in Li-ion batteries: a chemical issue," *Chemical Society Reviews*, vol. 47, no. 13, pp. 4924–4933, 2018.

[111] M. Lucu, E. Martinez-Laserna, I. Gandiaga, and H. Camblong, "A critical review on self-adaptive Li-ion battery ageing models," *Journal of Power Sources*, vol. 401, pp. 85–101, 2018.

[112] M. Fasahat and M. Manthouri, "State of charge estimation of lithium-ion batteries using hybrid autoencoder and Long Short Term Memory neural networks," *Journal of Power Sources*, vol. 469, p. 228375, 2020.

[113] A. Makhzani and B. Frey, "K-sparse autoencoders," *arXiv preprint arXiv:1312.5663*, 2013.

[114] W. H. L. Pinaya, S. Vieira, R. Garcia-Dias, and A. Mechelli, "Autoencoders," in *Machine learning*, Elsevier, 2020, pp. 193–208.

[115] P. Baldi, "Autoencoders, unsupervised learning, and deep architectures," in *Proceedings of ICML workshop on unsupervised and transfer learning*, 2012, pp. 37–49.

[116] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.

[117] G. Dong, G. Liao, H. Liu, and G. Kuang, "A review of the autoencoder and its variants: A comparative perspective from target recognition in synthetic-aperture radar images," *IEEE Geoscience and Remote Sensing Magazine*, vol. 6, no. 3, pp. 44–68, 2018.

[118] D. Bank, N. Koenigstein, and R. Giryes, "Autoencoders," *arXiv preprint arXiv:2003.05991*, 2020.

[119] A. Ng, "Sparse autoencoder," *CS294A Lecture notes*, vol. 72, no. 2011, pp. 1–19, 2011.

[120] A. Sankaran, M. Vatsa, R. Singh, and A. Majumdar, "Group sparse autoencoder," *Image and Vision Computing*, vol. 60, pp. 64–74, 2017.

[121] J. Xu *et al.*, "Stacked sparse autoencoder (SSAE) for nuclei detection on breast cancer histopathology images," *IEEE transactions on medical imaging*, vol. 35, no. 1, pp. 119–130, 2015.

[122] D. N. Schreiber-Gregory, "Ridge Regression and multicollinearity: An in-depth review," *Model Assisted Statistics and Applications*, vol. 13, no. 4, pp. 359–365, 2018.

[123] J. Corchado and B. Lees, "Artificial neural networks in pattern recognition: multicollinearity and heterocedasticity," in *Colloquium On Knowledge Discovery. LONDON, UK*, 1998.

[124] Z. Shi, M. Savargaonkar, A. A. Chehade, and A. A. Hussein, "A Long Short-Term Memory

Network for Online State-of-Charge Estimation of Li-ion Battery Cells," in *2020 IEEE Transportation Electrification Conference & Expo (ITEC)*, 2020, pp. 594–597.

[125] C. Bian, H. He, and S. Yang, "Stacked bidirectional long short-term memory networks for state-of-charge estimation of lithium-ion batteries," *Energy*, vol. 191, p. 116538, 2020.

[126] I. Oyewole, M. Savargaonkar, A. Chehade, and Y. Kim, "A Hybrid Long Short-Term Memory Network for State-of-Charge Estimation of Li-ion Batteries," in *2021 IEEE Transportation Electrification Conference & Expo (ITEC)*, 2021, pp. 469–473.

[127] F. Yang, X. Song, F. Xu, and K.-L. Tsui, "State-of-charge estimation of lithium-ion batteries via long short-term memory network," *Ieee Access*, vol. 7, pp. 53792–53799, 2019.

[128] I. Oyewole, A. Chehade, and Y. Kim, "A controllable deep transfer learning network with multiple domain adaptation for battery state-of-charge estimation," *Applied Energy*, vol. 312, p. 118726, 2022.

[129] B. Saha and K. Goebel, "Battery data set," *NASA AMES prognostics data repository*, 2007.

[130] K. A. Severson *et al.*, "Data-driven prediction of battery cycle life before capacity degradation," *Nature Energy*, vol. 4, no. 5, pp. 383–391, 2019.

[131] W. Diao, S. Saxena, and M. Pecht, "Accelerated cycle life testing and capacity degradation modeling of LiCoO2-graphite cells," *Journal of Power Sources*, vol. 435, p. 226830, 2019.

[132] G. O. Sahinoglu, M. Pajovic, Z. Sahinoglu, Y. Wang, P. V Orlik, and T. Wada, "Battery state-of-charge estimation based on regular/recurrent Gaussian process regression," *IEEE Transactions on Industrial Electronics*, vol. 65, no. 5, pp. 4311–4321, 2017.

[133] L. Ma *et al.*, "Robust state of charge estimation based on a sequence-to-sequence mapping model with process information," *Journal of Power Sources*, vol. 474, p. 228691, 2020.

[134] T. Tanizawa, T. Suzumiya, and K. Ikeda, "Cloud-connected battery management system supporting e-mobility," *Fujitsu Sci. Tech. J*, vol. 51, no. 4, pp. 27–35, 2015.

[135] X. Zhao, S. He, and M. Xie, "Utilizing experimental degradation data for warranty cost optimization under imperfect repair," *Reliability Engineering & System Safety*, vol. 177, pp. 108–119, 2018.

[136] X. Wang, B. Liu, and X. Zhao, "A performance-based warranty for products subject to competing hard and soft failures," *International Journal of Production Economics*, vol. 233, p. 107974, 2021.

[137] B. K. Rai, "Warranty spend forecasting for subsystem failures influenced by calendar month seasonality," *IEEE Transactions on Reliability*, vol. 58, no. 4, pp. 649–657, 2009.

[138] Y. Wang, Y. Liu, X. Li, and J. Chen, "Multi-phase reliability growth test planning for repairable products sold with a two-dimensional warranty," *Reliability Engineering & System Safety*, vol.

189, pp. 315–326, 2019.

[139] G. S. Wasserman, “An application of dynamic linear models for predicting warranty claims,” *Computers and Industrial Engineering*, vol. 22, no. 1, pp. 37–47, 1992.

[140] M. R. Karim and K. Suzuki, “Analysis of warranty claim data: A literature review,” *International Journal of Quality and Reliability Management*, vol. 22, no. 7, pp. 667–686, 2005.

[141] X. Wang, W. Xie, Z. S. Ye, and L. C. Tang, “Aggregate discounted warranty cost forecasting considering the failed-but-not-reported events,” *Reliability Engineering and System Safety*, vol. 168, no. April, pp. 355–364, 2017.

[142] S. K. Gupta, S. De, and A. Chatterjee, “Warranty forecasting from incomplete two-dimensional warranty data,” *Reliability Engineering and System Safety*, vol. 126, pp. 1–13, 2014.

[143] A. Dai, Z. Zhang, P. Hou, J. Yue, S. He, and Z. He, “Warranty Claims Forecasting for New Products Sold with a Two-Dimensional Warranty,” *Journal of Systems Science and Systems Engineering*, vol. 28, no. 6, pp. 715–730, 2019.

[144] K. Lin and Y. Chen, “Analysis of two-dimensional warranty data considering global and local dependence of heterogeneous marginals,” *Reliability Engineering & System Safety*, vol. 207, p. 107327, 2021.

[145] X. Li, Z. Liu, Y. Wang, and M. Li, “Optimal burn-in strategy for two-dimensional warranted products considering preventive maintenance,” *International Journal of Production Research*, vol. 57, no. 17, pp. 5414–5431, 2019.

[146] Z.-S. Ye, D. N. P. Murthy, M. Xie, and L.-C. Tang, “Optimal burn-in for repairable products sold with a two-dimensional warranty,” *IIE transactions*, vol. 45, no. 2, pp. 164–176, 2013.

[147] N. D. Singpurwalla and S. Wilson, “The warranty problem: its statistical and game-theoretic aspects,” *SIAM review*, vol. 35, no. 1, pp. 17–42, 1993.

[148] G. S. Wasserman and A. Sudjianto, “A comparison of three strategies for forecasting warranty claims,” *IIE Transactions (Institute of Industrial Engineers)*, vol. 28, no. 12, pp. 967–977, 1996.

[149] S. Wu and A. Akbarov, “Support vector regression for warranty claim forecasting,” *European Journal of Operational Research*, vol. 213, no. 1, pp. 196–204, 2011.

[150] R. Khoshkangini, S. Pashami, and S. Nowaczyk, “Warranty claim rate prediction using logged vehicle data,” *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 11804 LNAI, pp. 663–674, 2019.

[151] S. Shokouhyar, S. Ahmadi, and M. Ashrafzadeh, “Promoting a novel method for warranty claim prediction based on social network data,” *Reliability Engineering & System Safety*, vol. 216, p. 108010, 2021.

[152] J.-G. Lee, T. Kim, K. W. Sung, and S. W. Han, "Automobile parts reliability prediction based on claim data: The comparison of predictive effects with deep learning," *Engineering Failure Analysis*, vol. 129, p. 105657, 2021.

[153] Z. Xu and J. H. Saleh, "Machine learning for reliability engineering and safety applications: Review of current status and future opportunities," *Reliability Engineering & System Safety*, p. 107530, 2021.

[154] A. Kleyner and P. Sandborn, "A warranty forecasting model based on piecewise statistical distributions and stochastic simulation," *Reliability Engineering and System Safety*, vol. 88, no. 3, pp. 207–214, 2005.

[155] M. Fredette and J. F. Lawless, "Finite-horizon prediction of recurrent events, with application to forecasts of warranty claims," *Technometrics*, vol. 49, no. 1, pp. 66–80, 2007.

[156] V. V Krivtsov and A. Y. Yevkin, "Comparative analysis of optimal maintenance policies under general repair with underlying weibull distributions," *Енергетика та комп'ютерно-інтегровані технології в АПК*, no. 2, pp. 57–64, 2014.

[157] K. D. Majeske, "A non-homogeneous Poisson process predictive model for automobile warranty claims," *Reliability Engineering and System Safety*, vol. 92, no. 2, pp. 243–251, 2007.

[158] M. A. T. Figueiredo and A. K. Jain, "Unsupervised learning of finite mixture models," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 24, no. 3, pp. 381–396, 2002.

[159] D. A. Reynolds, "Gaussian Mixture Models.," *Encyclopedia of biometrics*, vol. 741, 2009.

[160] C. Bouveyron, G. Celeux, T. B. Murphy, and A. E. Raftery, *Model-based clustering and classification for data science: with applications in R*, vol. 50. Cambridge University Press, 2019.

[161] C. Biernacki, G. Celeux, and G. Govaert, "Choosing starting values for the EM algorithm for getting the highest likelihood in multivariate Gaussian mixture models," *Computational Statistics & Data Analysis*, vol. 41, no. 3–4, pp. 561–575, 2003.

[162] G. Celeux and G. Govaert, "A classification EM algorithm for clustering and two stochastic versions," *Computational statistics & Data analysis*, vol. 14, no. 3, pp. 315–332, 1992.

[163] S. Wu and A. Akbarov, "Forecasting warranty claims for recently launched products," *Reliability Engineering and System Safety*, vol. 106, pp. 160–164, 2012.

[164] V. V. Krivtsov, "Practical extensions to NHPP application in repairable system reliability analysis," *Reliability Engineering and System Safety*, vol. 92, no. 5, pp. 560–562, 2007.

[165] H. He, Y. Wang, R. Han, M. Han, Y. Bai, and Q. Liu, "An improved MPC-based energy management strategy for hybrid vehicles using V2V and V2I communications," *Energy*, vol. 225, p. 120273, 2021.

[166] E. Farsimadan, F. Palmieri, L. Moradi, D. Conte, and B. Paternoster, "Vehicle-to-everything (V2X) communication scenarios for vehicular ad-hoc networking (VANET): an overview," in *International Conference on Computational Science and Its Applications*, 2021, pp. 15–30.

[167] A. R. Khan *et al.*, "DSRC technology in Vehicle-to-Vehicle (V2V) and Vehicle-to-Infrastructure (V2I) IoT system for Intelligent Transportation System (ITS): a review," *Recent Trends in Mechatronics Towards Industry 4.0*, pp. 97–106, 2022.

[168] C. Xu, H. Wu, Y. Zhang, S. Dai, H. Liu, and J. Tian, "A Real-Time Complex Road AI Perception Based on 5G-V2X for Smart City Security," *Wireless Communications and Mobile Computing*, vol. 2022, 2022.

[169] V.-T. Ta and A. Dvir, "A secure road traffic congestion detection and notification concept based on V2I communications," *Vehicular Communications*, vol. 25, p. 100283, 2020.

[170] D. Negrut, R. Serban, A. Elmquist, D. Hatch, E. Nutt, and P. Sheets, "Autonomous Vehicles in the Cyberspace: Accelerating Testing via Computer Simulation," *SAE Technical Papers*, vol. 2018-April, pp. 1–14, 2018.

[171] H. Li, F. Liu, Z. Zhao, and M. Karimzadeh, "Effective safety message dissemination with vehicle trajectory predictions in V2X networks," *Sensors*, vol. 22, no. 7, p. 2686, 2022.

[172] J. P. Jeong *et al.*, "Vehicular Communications," 2021.

[173] H. Caesar *et al.*, "nuscenes: A multimodal dataset for autonomous driving," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 11621–11631.

[174] M. F. Chang *et al.*, "Argoverse: 3D tracking and forecasting with rich maps," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2019-June, pp. 8740–8749, 2019.

[175] X. Huang *et al.*, "The apolloscape dataset for autonomous driving," *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, vol. 2018-June, pp. 1067–1073, 2018.

[176] P. Sun *et al.*, "Scalability in perception for autonomous driving: Waymo open dataset," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 2446–2454.

[177] H. Yu *et al.*, "Dair-v2x: A large-scale dataset for vehicle-infrastructure cooperative 3d object detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 21361–21370.

[178] X. Weng, Y. Man, J. Park, Y. Yuan, M. O'Toole, and K. M. Kitani, "All-In-One Drive: A Comprehensive Perception Dataset with High-Density Long-Range Point Clouds," 2021.

[179] I. Kurzidem, A. Saad, and P. Schleiss, "A systematic approach to analyzing perception

architectures in autonomous vehicles," in *Model-Based Safety and Assessment: 7th International Symposium, IMBSA 2020, Lisbon, Portugal, September 14–16, 2020, Proceedings 7*, 2020, pp. 149–162.

[180] V. Punzo, M. T. Borzacchiello, and B. Ciuffo, "On the assessment of vehicle trajectory data accuracy and application to the Next Generation SIMulation (NGSIM) program data," *Transportation Research Part C: Emerging Technologies*, vol. 19, no. 6, pp. 1243–1262, 2011.

[181] Y. Liao, J. Xie, and A. Geiger, "KITTI-360: A Novel Dataset and Benchmarks for Urban Scene Understanding in 2D and 3D," pp. 1–31, 2021.

[182] A. Breuer, J.-A. Termöhlen, S. Homoceanu, and T. Fingscheidt, "openDD: A large-scale roundabout drone dataset," in *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, 2020, pp. 1–6.

[183] S. Casas, W. Luo, and R. Urtasun, "Intentnet: Learning to predict intention from raw sensor data," in *Conference on Robot Learning*, 2018, pp. 947–956.

[184] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator," *arXiv preprint arXiv:1711.03938*, 2017.

[185] R. Xu, H. Xiang, X. Xia, X. Han, J. Liu, and J. Ma, "OPV2V: An Open Benchmark Dataset and Fusion Pipeline for Perception with Vehicle-to-Vehicle Communication," *arXiv preprint arXiv:2109.07644*, 2021.

[186] J.-E. Deschaud, "KITTI-CARLA: a KITTI-like dataset generated by CARLA Simulator," *arXiv preprint arXiv:2109.00892*, 2021.

[187] Y. Li, Z. An, Z. Wang, Y. Zhong, S. Chen, and C. Feng, "V2X-Sim: A Virtual Collaborative Perception Dataset for Autonomous Driving," *arXiv preprint arXiv:2202.08449*, 2022.

[188] E. Arnold, S. Mozaffari, and M. Dianati, "Fast and robust registration of partially overlapping point clouds," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 1502–1509, 2021.

[189] E. Arnold, M. Dianati, R. de Temple, and S. Fallah, "Cooperative perception for 3D object detection in driving scenarios using infrastructure sensors," *IEEE Transactions on Intelligent Transportation Systems*, 2020.

[190] A. Best, S. Narang, L. Pasqualin, D. Barber, and D. Manocha, "Autonovi-sim: Autonomous vehicle simulation platform with weather, sensing, and traffic control," in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 2018, pp. 1048–1056.

[191] S. R. Richter, Z. Hayder, and V. Koltun, "Playing for benchmarks," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2213–2222.

[192] B. Wymann, E. Espié, C. Guionneau, C. Dimitrakakis, R. Coulom, and A. Sumner, "Torcs, the open racing car simulator," *Software available at http://torcs. sourceforge. net*, vol. 4, no. 6, p. 2, 2000.

[193] S. Shah, D. Dey, C. Lovett, and A. Kapoor, “Airsim: High-fidelity visual and physical simulation for autonomous vehicles,” in *Field and service robotics*, 2018, pp. 621–635.

[194] J. W. Hall, K. L. Smith, L. Titus-Glover, J. C. Wambold, T. J. Yager, and Z. Rado, “Guide for pavement friction,” *Final Report for NCHRP Project*, vol. 1, p. 43, 2009.

[195] P. C. Anastasopoulos and F. L. Mannering, “The effect of speed limits on drivers’ choice of speed: a random parameters seemingly unrelated equations approach,” *Analytic methods in accident research*, vol. 10, pp. 1–11, 2016.

[196] “Automated Vehicles for Safety | NHTSA.” [Online]. Available: https://www.nhtsa.gov/technology-innovation/automated-vehicles-safety. [Accessed: 17-Jul-2020].

[197] “Naturalistic driving studies: | National Highway Traffic Safety Administration (NHTSA).” [Online]. Available: https://one.nhtsa.gov/Research/Human-Factors/Naturalistic-driving-studies: [Accessed: 17-Jul-2020].

[198] S. G. Klauer, T. A. Dingus, V. L. Neale, J. D. Sudweeks, and D. J. Ramsey, “The impact of driver inattention on near-crash/crash risk: An analysis using the 100-car naturalistic driving study data,” 2006.

[199] B. Sultan and M. Mcdonald, “Assessing The Safety Benefit of Automatic Collision Avoidance Systems (During Emergency Braking Situations),” *Proceedings of the 18th International Technical Conference on the Enhanced Safety of Vehicles (ESV)*, vol. 44, no. 0, pp. 1–13, 2003.

[200] C. G. Wilmot and M. Khanal, “Effect of speed limits on speed and safety: a review,” *Transport Reviews*, vol. 19, no. 4, pp. 315–329, 1999.

[201] P. Rämä, “Effects of weather-controlled variable speed limits and warning signs on driver behavior,” *Transportation Research Record*, vol. 1689, no. 1, pp. 53–59, 1999.

[202] X.-Y. Lu and S. E. Shladover, “Review of variable speed limits and advisories: Theory, algorithms, and practice,” *Transportation research record*, vol. 2423, no. 1, pp. 15–23, 2014.

[203] S. Lefèvre, D. Vasquez, and C. Laugier, “A survey on motion prediction and risk assessment for intelligent vehicles,” *ROBOMECH journal*, vol. 1, no. 1, pp. 1–14, 2014.

[204] M. Brännström, E. Coelingh, and J. Sjöberg, “Model-based threat assessment for avoiding arbitrary vehicle collisions,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 11, no. 3, pp. 658–669, 2010.

[205] S. Ammoun and F. Nashashibi, “Real time trajectory prediction for collision risk estimation between vehicles,” in *2009 IEEE 5th International Conference on Intelligent Computer Communication and Processing*, 2009, pp. 417–422.

[206] G. S. Aoude, V. R. Desaraju, L. H. Stephens, and J. P. How, “Driver behavior classification at intersections and validation on large naturalistic data set,” *IEEE Transactions on Intelligent*

Transportation Systems, vol. 13, no. 2, pp. 724–736, 2012.

[207] C. Laugier *et al.*, “Probabilistic analysis of dynamic scenes and collision risks assessment to improve driving safety,” *IEEE Intelligent Transportation Systems Magazine*, vol. 3, no. 4, pp. 4–19, 2011.

[208] F. Havlak and M. Campbell, “Discrete and continuous, probabilistic anticipation for autonomous robots in urban environments,” *IEEE Transactions on Robotics*, vol. 30, no. 2, pp. 461–474, 2013.

[209] Q. Tran and J. Firl, “Online maneuver recognition and multimodal trajectory prediction for intersection assistance using non-parametric regression,” in *2014 ieee intelligent vehicles symposium proceedings*, 2014, pp. 918–923.

[210] A. Eidehall and L. Petersson, “Statistical threat assessment for general road scenes using Monte Carlo sampling,” *IEEE Transactions on intelligent transportation systems*, vol. 9, no. 1, pp. 137–147, 2008.

[211] T. Buhet, E. Wirbel, and X. Perrotton, “Conditional vehicle trajectories prediction in carla urban environment,” *Proceedings - 2019 International Conference on Computer Vision Workshop, ICCVW 2019*, pp. 2310–2319, 2019.

[212] N. Deo and M. M. Trivedi, “Convolutional social pooling for vehicle trajectory prediction,” *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, vol. 2018-June, pp. 1549–1557, 2018.

[213] H. Wang, P. Cai, R. Fan, Y. Sun, and M. Liu, “End-to-end interactive prediction and planning with optical flow distillation for autonomous driving,” *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, vol. 1, pp. 2229–2238, 2021.

[214] Y. Ma, X. Zhu, S. Zhang, R. Yang, W. Wang, and D. Manocha, “TrafficPredict: Trajectory prediction for heterogeneous traffic-agents,” *33rd AAAI Conference on Artificial Intelligence, AAAI 2019, 31st Innovative Applications of Artificial Intelligence Conference, IAAI 2019 and the 9th AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019*, no. Kalman 1960, pp. 6120–6127, 2019.

[215] S. Pellegrini, A. Ess, K. Schindler, and L. Van Gool, “You’ll never walk alone: Modeling social behavior for multi-target tracking,” in *2009 IEEE 12th International Conference on Computer Vision*, 2009, pp. 261–268.

[216] N. Rao and S. Sundaram, “Spatio-Temporal Look-Ahead Trajectory Prediction using Memory Neural Network,” *Proceedings of the International Joint Conference on Neural Networks*, vol. 2021-July, 2021.

[217] L. Lin, “Prediction Using LSTMs With Spatial – Temporal Attention Mechanisms,” no. February 2021, pp. 2–12.

[218] F. Altche and A. De La Fortelle, “An LSTM network for highway trajectory prediction,” *IEEE*

Conference on Intelligent Transportation Systems, Proceedings, ITSC, vol. 2018-March, pp. 353–359, 2018.

[219] S. Casas, C. Gulino, R. Liao, and R. Urtasun, “Spagnn: Spatially-aware graph neural networks for relational behavior forecasting from sensor data,” in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 9491–9497.

[220] J. Gao *et al.*, “Vectornet: Encoding hd maps and agent dynamics from vectorized representation,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 11525–11533.

[221] N. Lee, W. Choi, P. Vernaza, C. B. Choy, P. H. S. Torr, and M. Chandraker, “Desire: Distant future prediction in dynamic scenes with interacting agents,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 336–345.

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

[223] Y. Chai, B. Sapp, M. Bansal, and D. Anguelov, “Multipath: Multiple probabilistic anchor trajectory hypotheses for behavior prediction,” *arXiv preprint arXiv:1910.05449*, 2019.

[224] J. Hong, B. Sapp, and J. Philbin, “Rules of the road: Predicting driving behavior with a convolutional model of semantic interactions,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 8454–8462.

[225] H. Cui *et al.*, “Multimodal trajectory predictions for autonomous driving using deep convolutional networks,” in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 2090–2096.

[226] L. Fang, Q. Jiang, J. Shi, and B. Zhou, “Tpnet: Trajectory proposal network for motion prediction,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 6797–6806.

[227] F. Leon and M. Gavrilescu, “A review of tracking and trajectory prediction methods for autonomous driving,” *Mathematics*, vol. 9, no. 6, p. 660, 2021.

[228] A. Rudenko, L. Palmieri, M. Herman, K. M. Kitani, D. M. Gavrila, and K. O. Arras, “Human motion trajectory prediction: A survey,” *The International Journal of Robotics Research*, vol. 39, no. 8, pp. 895–935, 2020.

[229] H. Winner, W. Wachenfeld, and P. Junietz, “Safety Assurance for Highly Automated Driving - The PEGASUS Approach,” *TRB Annual Meeting*, 2017.

[230] C. Rödel, S. Stadler, A. Meschtscherjakov, and M. Tscheligi, “Towards autonomous cars: the effect of autonomy levels on acceptance and user experience,” in *Proceedings of the 6th international conference on automotive user interfaces and interactive vehicular applications*, 2014, pp. 1–8.

[231] R. Salay, R. Queiroz, and K. Czarnecki, “An analysis of ISO 26262: Using machine learning safely in automotive software,” *arXiv preprint arXiv:1709.02435*, 2017.

[232] D. Nalic, T. Mihalj, M. Bäumler, M. Lehmann, A. Eichberger, and S. Bernsteiner, “Scenario based testing of automated driving systems: A literature survey,” in *FISITA web Congress*, 2020.

[233] A. Schnellbach and G. Griessnig, “Development of the ISO 21448,” in *European Conference on Software Process Improvement*, 2019, pp. 585–593.

[234] F. Klück, M. Zimmermann, F. Wotawa, and M. Nica, “Performance comparison of two search-based testing strategies for ADAS system validation,” in *IFIP International Conference on Testing Software and Systems*, 2019, pp. 140–156.

[235] W. Ding, C. Xu, H. Lin, B. Li, and D. Zhao, “A Survey on Safety-critical Scenario Generation from Methodological Perspective,” *arXiv preprint arXiv:2202.02215*, 2022.

[236] E. B. Fox, E. B. Sudderth, M. I. Jordan, and A. S. Willsky, “An HDP-HMM for systems with state persistence,” in *Proceedings of the 25th international conference on Machine learning*, 2008, pp. 312–319.

[237] J. M. Scanlon, K. D. Kusano, T. Daniel, C. Alderson, A. Ogle, and T. Victor, “Waymo simulated driving behavior in reconstructed fatal crashes within an autonomous vehicle operating domain,” *Accident Analysis & Prevention*, vol. 163, p. 106454, 2021.

[238] S. Suo, S. Regalado, S. Casas, and R. Urtasun, “Trafficsim: Learning to simulate realistic multi-agent behaviors,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 10400–10409.

[239] W. Ding, B. Chen, B. Li, K. J. Eun, and D. Zhao, “Multimodal safety-critical scenarios generation for decision-making algorithms evaluation,” *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 1551–1558, 2021.

[240] S. Kuutti, S. Fallah, and R. Bowden, “Training adversarial agents to exploit weaknesses in deep control policies,” in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 108–114.

[241] Y. Chen *et al.*, “Geosim: Realistic video simulation via geometry-aware composition for self-driving,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 7230–7240.

[242] W. Ding, B. Chen, M. Xu, and D. Zhao, “Learning to collide: An adaptive safety-critical scenarios generating method,” in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020, pp. 2243–2250.

[243] M. Xu *et al.*, “Accelerated Policy Evaluation: Learning Adversarial Environments with Adaptive Importance Sampling,” *arXiv preprint arXiv:2106.10566*, 2021.

[244] W. Ding, C. Xu, M. Arief, H. Lin, B. Li, and D. Zhao, “A Survey on Safety-Critical Driving Scenario Generation-A Methodological Perspective.,” *CoRR*, 2022.

[245] T. Menzel, G. Bagschik, and M. Maurer, “Scenarios for development, test and validation of automated vehicles,” in *2018 IEEE Intelligent Vehicles Symposium (IV)*, 2018, pp. 1821–1827.

[246] P. Cai, Y. Lee, Y. Luo, and D. Hsu, “Summit: A simulator for urban driving in massive mixed traffic,” in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 4023–4029.

[247] M. Zhou *et al.*, “Smarts: Scalable multi-agent reinforcement learning training school for autonomous driving,” *arXiv preprint arXiv:2010.09776*, 2020.

[248] Q. Li, Z. Peng, Z. Xue, Q. Zhang, and B. Zhou, “Metadrive: Composing diverse driving scenarios for generalizable reinforcement learning,” *arXiv preprint arXiv:2109.12674*, 2021.

[249] A. Rana and A. Malhi, “Building Safer Autonomous Agents by Leveraging Risky Driving Behavior Knowledge,” in *2021 International Conference on Communications, Computing, Cybersecurity, and Informatics (CCCI)*, 2021, pp. 1–6.

[250] D. McDuff *et al.*, “Causalcity: Complex simulations with agency for causal discovery and reasoning,” in *Conference on Causal Learning and Reasoning*, 2022, pp. 559–575.

[251] S. Shiroshita *et al.*, “Behaviorally diverse traffic simulation via reinforcement learning,” in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020, pp. 2103–2110.

[252] R. F. Benekohal and J. Treiterer, “CARSIM: Car-following model for simulation of traffic in normal and stop-and-go conditions,” *Transportation research record*, vol. 1194, pp. 99–111, 1988.

[253] J. Herman *et al.*, “Learn-to-race: A multimodal control environment for autonomous racing,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 9793–9802.

[254] T. V. Samak, C. V. Samak, and M. Xie, “AutoDRIVE Simulator: A Simulator for Scaled Autonomous Vehicle Research and Education,” in *2021 2nd International Conference on Control, Robotics and Intelligent System*, 2021, pp. 1–5.

[255] O. Michel, “Cyberbotics Ltd. Webots^{TM}: professional mobile robot simulation,” *International Journal of Advanced Robotic Systems*, vol. 1, no. 1, p. 5, 2004.

[256] M. Müller, V. Casser, J. Lahoud, N. Smith, and B. Ghanem, “Sim4cv: A photo-realistic simulator for computer vision applications,” *International Journal of Computer Vision*, vol. 126, no. 9, pp. 902–919, 2018.

[257] C. L. Azevedo *et al.*, “Simmobility short-term: An integrated microscopic mobility simulator,” *Transportation Research Record*, vol. 2622, no. 1, pp. 13–23, 2017.

[258] W. Schwarting *et al.*, "Deep latent competition: Learning to race using visual control policies in latent space," *arXiv preprint arXiv:2102.09812*, 2021.

[259] G. Rong *et al.*, "LGSVL Simulator: A High Fidelity Simulator for Autonomous Driving," *arXiv preprint arXiv:2005.03778*, 2020.

[260] J. H. Gao and L.-S. Peh, "RoadRunner: Infrastructure-less vehicular congestion control," 2014.

[261] Y. Sun, X. Yang, H. Xiao, and H. Feng, "An intelligent driving simulation platform: architecture, implementation and application," in *2020 International Conference on Connected and Autonomous Driving (MetroCAD)*, 2020, pp. 71–75.

[262] M. Haklay and P. Weber, "Openstreetmap: User-generated street maps," *IEEE Pervasive computing*, vol. 7, no. 4, pp. 12–18, 2008.

[263] M. Quigley *et al.*, "ROS: an open-source Robot Operating System," in *ICRA workshop on open source software*, 2009, vol. 3, no. 3.2, p. 5.

[264] A. Foster, M. Jankowiak, E. Bingham, Y. Whye Teh, T. Rainforth, and N. Goodman, "Variational Optimal Experiment Design: Efficient Automation of Adaptive Experiments," no. NeurIPS, 2018.

[265] B. T. Vincent and T. Rainforth, "The DARC Toolbox: automated, flexible, and efficient delayed and risky choice experiments using Bayesian adaptive design," *The DARC Toolbox: automated, flexible, and efficient delayed and risky choice experiments using Bayesian adaptive design*, 2017.

[266] Y. Gal and Z. Ghahramani, "Bayesian convolutional neural networks with Bernoulli approximate variational inference," *arXiv preprint arXiv:1506.02158*, 2015.

[267] Y. Gal, R. Islam, and Z. Ghahramani, "Deep bayesian active learning with image data," in *International Conference on Machine Learning*, 2017, pp. 1183–1192.

[268] A. I. Schein and L. H. Ungar, "Active learning for logistic regression: an evaluation," *Machine Learning*, vol. 68, no. 3, pp. 235–265, 2007.

[269] T. Danka and P. Horvath, "modAL: A modular active learning framework for Python," *arXiv preprint arXiv:1805.00979*, 2018.