LEARNING DATA-DRIVEN MODELS FOR DECISION-MAKING
IN INTELLIGENT PHYSICAL SYSTEMS

A Dissertation in
Mechanical Engineering
by
Nurali Virani

© 2017 Nurali Virani

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Doctor of Philosophy

May 2017
The dissertation of Nurali Virani was reviewed and approved* by the following:

Asok Ray  
Distinguished Professor of Mechanical Engineering  
Dissertation Co-advisor, Committee Co-chair

Sean Brennan  
Professor of Mechanical Engineering  
Dissertation Co-advisor

Shashi Phoha  
Graduate Faculty of Electrical Engineering  
Director, Information Science & Technology Center,  
Applied Research Laboratory  
Committee Co-chair

Minghui Zhu  
Assistant Professor of Electrical Engineering

Ji-Woong Lee  
Researcher, Control Theory and Statistical Learning  
Special Member

Karen Thole  
Department Head, Mechanical and Nuclear Engineering

*Signatures are on file in the Graduate School.
Abstract

Intelligent physical systems use machine learning for a variety of tasks from health monitoring to control. As the dependence on autonomous decision-making agents increases, it is of importance to understand and quantify the uncertainty associated with the decisions from machine learning frameworks. In order to facilitate the interaction with human agents (e.g., maintenance engineers and medical doctors) as well as to enable robust control for safety (e.g., autonomous navigation and sensor network adaptation), density estimation enables quantification of uncertainty in the output of a learning framework. In statistical learning, density estimation is a core problem, where the objective is to identify the underlying distribution from which the data are being generated. In this work, density estimation is established as a practical tool for data-driven modeling. A new and simple technique for density estimation is developed using concepts from statistical learning and optimization theory. Along with detection, classification, estimation, and tracking, which are crucial in learning and control, these models can also quantify uncertainty in their outputs.

This dissertation uses density estimation for developing new methods to solve practical problems of learning and decision-making. A few restrictive assumptions have been eliminated from these problems, yet tractable and accurate methods have been developed in this research. Specifically, in the sequential classification problem, the naïve Bayes' assumption of conditional independence between measurements, given state, is relaxed. A novel technique to learn a unified context from multi-modal sensor data is developed. This knowledge of context is used to achieve tractable and accurate multi-modal sensor fusion, which cannot be achieved using the naïve Bayes' assumption. Additionally, the context-aware measurement models are also used for unifying state estimation and dynamic sensor selection problems in a stochastic control framework. In sequential hypothesis testing with streaming data, the assumption that the observation sequence is independent and...
identically distributed (IID) has been removed by developing sequential tests for Markov models of time-series data. Further, density estimation has been used to create Markov models from multidimensional time-series data by developing a unified formulation for alphabet-size selection and measurement-space partitioning. In sequential tracking, the assumption of additive Gaussian noise has been eliminated by learning nonparametric density estimation-based measurement models, which can capture all the uncertainties in a given set of data. These measurement models have been used for state estimation and tracking with particle filters. In a sequential measurement model learning setting, the labels provided by instructors are allowed to be incorrect as the assumption of the instructor being perfect has not been used. A recursive density estimation algorithm has been developed and analyzed to show that correct models can be obtained even with noisy labels.

In physical systems, the assumptions noted above, usually do not hold due to causal dependencies, physical constraints, operating conditions, various uncertainties, etc., and hence have been selectively relaxed. The theoretical frameworks developed in this research have been validated using simulations and real-world experiments. The practical applications covered in this dissertation include target detection and classification in border surveillance, indoor localization of smart wheelchairs with user-assistance for safety during navigation, and detection of combustion instability using streaming data. These widely differing real-world problems have been used to illustrate the general applicability of the results developed in this thesis. It is envisioned that the formulations and results from this dissertation will be useful in data-driven modeling, real-time decision-making, and robust control of physical systems, making them more intelligent.
# Contents

List of Figures ix
List of Tables xii
Acknowledgments xiii

## Chapter 1

**Introduction** 1

1.1 Motivation ......................................................... 2
1.2 Research Overview and Research Problems ...................... 4
  1.2.1 Classification of Decisions ................................. 4
  1.2.2 Schematic of an Intelligent Physical System .............. 6
  1.2.3 Research Problems ......................................... 7
1.3 Applications ..................................................... 14
  1.3.1 Border Surveillance System ................................. 14
  1.3.2 Combustion Instability Detection .......................... 15
  1.3.3 User-assistance for Safety in Smart Wheelchairs ......... 16
1.4 Major Contributions of the Thesis .............................. 17
1.5 Organization of the Thesis ...................................... 18

## Chapter 2

**Nonparametric Density Estimation for Data-driven Modeling** 21

2.1 Introduction ..................................................... 21
2.2 Optimization Problems in Learning ............................... 23
  2.2.1 Empirical Risk Minimization ............................... 23
  2.2.2 Optimization Problems for Density Estimation ............ 24
  2.2.3 Linear Program for Density Estimation (LPDE) ............ 28
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.2.4</td>
<td>Other Optimization Problems</td>
<td>29</td>
</tr>
<tr>
<td>2.3</td>
<td>Analysis</td>
<td>30</td>
</tr>
<tr>
<td>2.4</td>
<td>Implementation and Simulation Results</td>
<td>33</td>
</tr>
<tr>
<td>2.4.1</td>
<td>Optimization Parameter Selection</td>
<td>33</td>
</tr>
<tr>
<td>2.4.2</td>
<td>MATLAB Implementation</td>
<td>34</td>
</tr>
<tr>
<td>2.4.3</td>
<td>Problem I: Density Estimation</td>
<td>35</td>
</tr>
<tr>
<td>2.4.4</td>
<td>Problem II: Anomaly Detection and Classification</td>
<td>39</td>
</tr>
<tr>
<td>2.5</td>
<td>Conclusion and Future Work</td>
<td>40</td>
</tr>
<tr>
<td>Chapter 3</td>
<td>Learning Context and Context-aware Sensor Fusion</td>
<td>42</td>
</tr>
<tr>
<td>3.1</td>
<td>Introduction</td>
<td>42</td>
</tr>
<tr>
<td>3.2</td>
<td>Mathematical Definition of Context</td>
<td>46</td>
</tr>
<tr>
<td>3.3</td>
<td>Learning Context from Data</td>
<td>49</td>
</tr>
<tr>
<td>3.4</td>
<td>Cardinality Reduction of Context Sets</td>
<td>52</td>
</tr>
<tr>
<td>3.4.1</td>
<td>Compression by Maximal Clique Enumeration</td>
<td>53</td>
</tr>
<tr>
<td>3.4.2</td>
<td>Compression by Subset Selection</td>
<td>57</td>
</tr>
<tr>
<td>3.4.3</td>
<td>Experiment and Results</td>
<td>60</td>
</tr>
<tr>
<td>3.5</td>
<td>Context-aware Sensor Fusion</td>
<td>64</td>
</tr>
<tr>
<td>3.5.1</td>
<td>Experiment and Results</td>
<td>65</td>
</tr>
<tr>
<td>3.6</td>
<td>Conclusion</td>
<td>66</td>
</tr>
<tr>
<td>Chapter 4</td>
<td>Learning Markov Models for Time-series Data</td>
<td>68</td>
</tr>
<tr>
<td>4.1</td>
<td>Introduction</td>
<td>68</td>
</tr>
<tr>
<td>4.2</td>
<td>Partitioning, Symbolization, and Modeling</td>
<td>70</td>
</tr>
<tr>
<td>4.2.1</td>
<td>Density Estimation</td>
<td>70</td>
</tr>
<tr>
<td>4.2.2</td>
<td>Partitioning and Symbolization</td>
<td>72</td>
</tr>
<tr>
<td>4.2.3</td>
<td>Markov Models of Time-series Data</td>
<td>72</td>
</tr>
<tr>
<td>4.3</td>
<td>Results and Discussion</td>
<td>73</td>
</tr>
<tr>
<td>4.4</td>
<td>Conclusion</td>
<td>78</td>
</tr>
<tr>
<td>Chapter 5</td>
<td>Sequential Hypothesis Testing with Models of Time-series Data</td>
<td>79</td>
</tr>
<tr>
<td>5.1</td>
<td>Introduction</td>
<td>79</td>
</tr>
<tr>
<td>5.2</td>
<td>Problem Formulation</td>
<td>81</td>
</tr>
<tr>
<td>5.2.1</td>
<td>Sequential Detection: IID Case</td>
<td>81</td>
</tr>
<tr>
<td>5.2.2</td>
<td>Sequential Detection: D-Markov Case</td>
<td>82</td>
</tr>
<tr>
<td>5.2.3</td>
<td>Problem Statement</td>
<td>83</td>
</tr>
<tr>
<td>5.3</td>
<td>Proposed Approach</td>
<td>84</td>
</tr>
</tbody>
</table>
Chapter 8
Learning Generative Measurement Models for State Estimation, Tracking, and Safety 145
8.1 Introduction ......................................................... 145
8.2 Measurement Model Learning Techniques 147
  8.2.1 Regression .................................................. 148
  8.2.2 Nonparametric Density Estimation ...................... 149
8.3 Particle Filtering for Tracking .............................. 152
  8.3.1 Motion Models .............................................. 152
  8.3.2 Particle Filters ............................................. 153
8.4 Results from Simulation .................................... 155
8.5 Results from Experiment ................................. 159
8.6 Safety Constraint for Input Classification .............. 162
  8.6.1 Problem Formulation and Solution .................. 163
  8.6.2 Results and Discussion ................................. 165
8.7 Conclusion and Future Work .............................. 167

Chapter 9
Conclusions and Research Vision 169
9.1 Summary of Results ........................................... 169
9.2 Suggestions for Future Work ............................ 172
  9.2.1 Research Directions ................................... 172
  9.2.2 Potential Applications ................................. 173

References 175

Appendix A
Testbeds for Experimental Validation 187
A.1 Border Control Testbed .................................. 187
A.2 Combustion Instability Detection Problem .......... 189
A.3 Smart Wheelchair Testbed .................................. 190
  A.3.1 Simulation Setup .................................... 190
  A.3.2 Hardware Setup .................................... 191

Appendix B
Symbolic Time-series Analysis 196
# List of Figures

1.1 Classification of types of decisions for data-driven systems ................. 4
1.2 Schematic of an intelligent physical system ................................. 6
1.3 The concept of a dynamic border control system .............................. 14
1.4 Robotic wheelchair platform built by Intelligent Vehicles and Systems Group (IVSG). Photo Credit: Kelilah Wolkowicz .................... 16
1.5 Chapter dependency and learning/estimation problems in each chapter .................. 19

2.1 Popular loss functions ......................................................... 27
2.2 Time to solve LPDE versus number of training points ....................... 34
2.3 Effect of kernel bandwidth on estimation error and complexity ................ 36
2.4 Distance from true model and comparison with Parzen density estimation ................................. 36
2.5 Pareto front ................................................................. 37
2.6 All Pareto-optimal solutions for Problem I .................................. 38
2.7 Simulation study for density estimation for detection problem ............... 39

3.1 Concept of intrinsic context-based classifier adaptation ....................... 44
3.2 Concept of extrinsic context-based classifier adaptation .......................... 45
3.3 Concept of context set compression .......................................... 53
3.4 Seismic sensor time series data for walking and running classes ............. 61
3.5 Mean and range of cardinality of the compressed context set ................. 61
3.6 Mean and range of classification accuracy with compressed context sets ........................................ 62

4.1 Sample phase trajectories from the Duffing model ............................... 74
4.2 Phase-space partitions obtained by the proposed technique. The red crosses indicate the seed point locations......................... 76
4.3 Sparsity of coefficient vector $\pi$ in density estimation .......................... 77
5.1 Log-likelihood ratio trajectories, when \( H_0 \) is true (shown in red) and when \( H_1 \) is true (shown in blue), for (a) Case 1 and (b) Case 2. The expected value of log-likelihood ratio in each case is shown for \( H_0 \) (with \(-\cdot\) line) and \( H_1 \) (with \(--\) line) .................................................. 96

5.2 Validation of convergence of average increment of expected log-likelihood ratio to the computed limit under each hypothesis .......................................... 97

5.3 Average sample length and estimated length for different values of \( p_d \) and constant \( p_{fa} = 0.001 \) .......................................................... 97

5.4 Histograms of sample length under each hypothesis with \( p_d = 0.999 \) and constant \( p_{fa} = 0.001 \) .......................................................... 98

5.5 Observed probability of detection for different values of \( p_d \) and constant \( p_{fa} = 0.001 \) .......................................................... 98

5.6 Average sample length and estimated length for different values of \( p_d \) and constant \( p_{fa} = 0.01 \) .......................................................... 101

5.7 Observed probability of detection for different values of \( p_d \) and constant \( p_{fa} = 0.01 \) .......................................................... 101

6.1 Information space and belief space discretization ............................................. 119

6.2 Sensor communication topology and activation permission graph ................... 120

6.3 Dynamic sensor team performance compared with all fixed teams .................. 122

6.4 Dynamic sensor team size for different cost structures ................................. 123

6.5 Dynamic sensor team size for time-varying cost structure ............................ 123

6.6 Dynamic sensor team size with and without context awareness .................... 125

7.1 Probabilistic graphical model showing dependencies ..................................... 131

7.2 Relative sample complexity for different misclassification rates .................... 139

7.3 Comparison of sequential updates of measurement densities for different approaches .......................................................... 141

7.4 Comparison of error in density estimation using different approaches ................ 142

8.1 Cost from \( \varepsilon \)-margin loss function (Source: [108]) ................................ 148

8.2 Example of tracking result with the reference model .................................. 157

8.3 Example of tracking result with the proposed model .................................. 158

8.4 Regression-based expected observation ..................................................... 159

8.5 Likelihood function from density estimation ................................................ 160

8.6 Global localization and tracking results for both kinematic models with regression (SVR) and density estimation-based (DE) mapping techniques .......................................................... 161

8.7 Trajectory comparison of the both kinematic models with SVR and DE mapping techniques .......................................................... 161
8.8 Posterior probability at each iteration for density estimation-based map with two different models. The true trajectory is shown as black solid line. ......................................................... 162
8.9 Schematic of a patient controlling a wheelchair using a BCI. Few inputs can lead the wheelchair to unsafe states. ................. 163
8.10 Demonstration of unsafe/safe input identification: Case 1 ............ 166
8.11 Demonstration of unsafe/safe input identification: Case 2 ............ 167
A.1 Location of testbed at the Penn State Test Track .................... 188
A.2 Sensor layout in the testbed ............................................ 188
A.3 UGS and Environment sensors used in the experiment .............. 189
A.4 Data acquisition and power distribution ................................ 189
A.5 Simulation setup with interface for human input .................... 191
A.6 The current configuration of the wheelchair. The enlarged image of the magnetometer is credited to [111] ........................ 192
A.7 Wheelchair with nine reflective object markers (encircled) for ground truth measurements ................................. 193
A.8 Histogram of the noise in the magnetometer for the cases of the motor running and stopped ............................... 194
A.9 Mardia’s Test Q-Q plot .................................................. 194
A.10 The three-meter linear path of the wheelchair and Vicon motion capture camera system ................................. 195
List of Tables

3.1 Classification performance for context-aware fusion .................. 66

4.1 Density estimation, partitioning, and classification performance with the proposed technique and symbolization rules .............. 75

5.1 $D$-Markov models used in simulation ............................... 94

5.2 Comparison of the performance ($p_{fa}$) and average sample length (ASL) of the proposed SPRT and maximum likelihood (ML) classifier (fixed length ($N$)) for same $p_d$. ................................. 99

5.3 Comparison of the performance ($p_d$) and average sample length (ASL) of the proposed SPRT and the maximum likelihood (ML) classifier (fixed length $N$) for the same $p_{fa}$ ............................... 102

6.1 Dynamic sensor team performance for different misclassification costs 122

6.2 Dynamic sensor team performance for different sequences of extrinsic context ................................................................. 123

6.3 Team formation examples for two different sequences of extrinsic context ................................................................. 124

6.4 Dynamic sensor team performance with and without context awareness ................................................................. 125

8.1 Tracking error comparison between the reference model and the proposed model ................................................................. 157

8.2 Number of observations used to represent the map .......................... 160

A.1 Operating conditions .......................................................... 190
Acknowledgments

This work would not have been possible without discussions and regular meetings with my committee members. Specifically, I would like to thank Dr. Asok Ray for his guidance and for encouraging me to collaborate with other members of my doctoral committee. I would like to thank Dr. Sean Brennan for the opportunity to work with Intelligent Vehicles and Systems Group (IVSG) and for giving me interesting ideas to pursue. I would like to thank Dr. Shashi Phoha for involving me in the Dynamic Data-driven Application Systems (DDDAS) project, for the feedback in weekly meetings, for the freedom to explore new areas, and for the opportunity to present in the project review meetings. I would like to thank Dr. Ji-Woong Lee for the brain-storming sessions, for the help in formulation of some of the problems in this work, and for the help in improving my technical writing. I would also like to thank Dr. Minghui Zhu for the discussions and for being adviser for M.S. degree in Electrical Engineering. This work has been supported in part by the U.S Air Force Office of Scientific Research (AFOSR) under Grant No. FA9550-12-1-0270 and Grant No.FA9550-15-1-0400 of the DDDAS program. I would like to take this opportunity to thank Dr. Frederica Darema (AFOSR) for motivating and inspiring me in the project review meetings and conferences.

I would like to thank Mr. Dennis Kovalick, Mr. David Klinikowski, and Dr. Sean Brennan for being very supportive and allowing us to use the Penn State Test Track facility for our border control experiments. Thanks to Kevin Fisher, Dr. Eric Keller, Mr. Michael Hauser, Ms. Pritthi Chattopadhyay, Mr. Noboru Takahashi, Mr. Yusuke Seto, Dr. Brian Smith, Mr. Zhenyuan Yuan, Mr. Jesse Perlstein, Mr. Zeyu Zhang, Mr. Chuong Nyugen, and Mr. Sihan Xiong, for help with experiment setup and data collection. I would also like to thank Kelilah Wolkowicz, Anthony Trezza, Christopher Miller, and other members of IVSG for the development of the smart wheelchair research platform, which I could use for experimental validation of the methods proposed in this dissertation. I would also

---

1 Any opinions, findings, and conclusions or recommendations expressed in this dissertation are those of the author and do not necessarily reflect the views of the sponsoring agencies.
like to thank Dr. Dominic Santavicca and Mr. Jihang Li for kindly providing the experimental data on combustion that is used for validation in this thesis.

I am very grateful to my colleagues in Dr. Ray’s research group for the exciting discussions on various challenging problems. I would like to especially thank Dr. Soumalaya Sarkar and Dr. Devesh Jha for their help and feedback. I would also like to thank my friends at Penn State for making these years interesting and memorable. I am thankful to my roomies Dr. Yajuvendra Singh Shekhawat, Dr. Lokanath Mohanta, and Mr. Partha Pratim Mishra for together creating a home away from home.

Finally, I am grateful to my mom, my dad, and my sister for supporting my urge to study more and explore further. I would like to thank my wife (and my best friend) Rucha Muley for her love and support in this long process. Without the patience, optimism, and unwaivering support of my family this work would not have come into existence.
Dedication

I dedicate this dissertation to my beloved mom and dad, who will always be my motivation.
Introduction

Intelligent systems often interact with uncertain and unmodeled environment, thus they need the ability to understand what they see in the environment. Further, they also need to understand their own behavior in a multitude of tasks, since practical physical systems can have errors, internal faults, and power issues. Several intelligent physical systems are being developed today and will only continue to grow in number, complexity, and reasoning abilities over time. Few examples of intelligent physical systems include:

- Mobile robots and vehicles with wide array of on-board sensors, such as camera, LIDAR, motion sensors, and global positioning system (GPS), for localization, navigation, and safety.
- Automated monitoring of physical systems such as gas turbine engines [100], shipboard auxiliary system [99], and battery systems [57].
- Border security and surveillance systems for detecting, classifying, and tracking intruders using unattended ground sensors, cameras, and radars [19].
- Remote monitoring of patients by body sensor networks for recording heart rate, respiration rate, muscular activity, and blood pressure as well as to report any abnormal observation [60].
- Environmental sensing by weather stations, soil condition monitoring, and livestock tracking for farm situation awareness in order to plan and improve crop yield and quality [116].

In order for the physical systems to have intelligent abilities mentioned above, the systems need several capabilities, including, but not limited to, mode estimation.
(e.g., nominal, faulty, or anomalous), state estimation (e.g., localization), and pattern recognition (e.g., pedestrian detection, object classification, etc.). Broadly these problems can be categorized as anomaly detection, classification, and regression. This thesis develops a new framework for data-driven modeling by using density estimation. The framework is used to address several practical problems in learning and decision-making by developing theoretical formulations and solving them using concepts from statistical learning, optimization theory, stochastic control, and nonlinear filtering. This chapter introduces and motivates the research problems and lists all the major contributions of this study.

1.1 Motivation

Data-driven models are needed for several systems as physics-based models are too complicated to derive or are infeasible to be implemented for on-board decision & control tasks. However, the main criticism of data-driven models is the lack of uncertainty quantification in learning and also during decision-making. Although, decision-making under uncertainty is a well-studied field in literature [11, 118], uncertainty in decision-making with learning systems is not yet well understood. If the decisions from data-driven models are accompanied by output uncertainty, then frameworks from stochastic control and/or motion planning [54], such as Markov decision processes (MDP) [11], can be incorporated to obtain controllers based on data-driven models of intelligent systems. Uncertainty quantification for output of learning systems also enables the system to be self-aware of the limitations of its decision-making abilities and it also enables design of robust controllers for autonomy, making the systems safer. Unlike classifiers from discriminative supervised learning [12], the generative models, which capture uncertainty, might be able to infer the change or detect any system anomalies from the data, when the underlying distribution changes due to some extrinsic effects. Thus, knowledge of uncertainty is useful and valuable for decision-making in physical systems.

In practical systems, the structure of the relation between variables in the dynamic model (current state $X_t$, current input $U_t$, and next state $X_{t+1}$) and/or measurement model (state/target class $X_t$ and observation/feature $Y_t$) need not be known. Moreover the uncertainty, due to measurement noise, process noise,
Nonparametric regression using kernel functions, such as support-vector regression (SVR) [108] and Gaussian process (GP) regression [85], can learn an accurate structure of the relation, but it cannot capture the distribution of the uncertainty. Thus, nonparametric density estimation is a proposed solution to learn a generative model between the variables in the practical systems, which can capture the correct structure as well as the uncertainty. Moreover, if the density estimate can be represented with a sparse set of kernels, then real-time decision-making can also be achieved, which is desirable in intelligent systems. The above discussion informs that, intelligent systems require us to extend the state-of-the-art of machine learning to simultaneously address three issues: (1) uncertainty quantification; (2) nonparametric density modeling; and (3) real-time inference.

Intuitively, the knowledge of uncertainty in estimation can help to choose better control actions and eventually incorporate safety in artificial intelligence-based control systems or decision-support systems. This can be illustrated with the following examples:

1. **Medical Testing and Diagnosis:** An automated system for detecting a disease (e.g., diabetes) uses data from few tests (e.g., blood sugar) to diagnose that disease. If the system gives only a binary response, then future course of action, such as more medical tests, emergency medical attention, or prescription of medicines, is difficult to decide even for medical care-givers. The knowledge of uncertainty in the decision can help to choose the next step. For example, the case with 51% confidence might need more tests, whereas, a system with binary response might suggest medication.

2. **Condition-based Maintenance of Equipments:** An automated system for scheduling maintenance of heavy machinery assets might use regression models to capture remaining useful life of various sub-systems and classification models to detect faults/anomalies during operation. The optimal action in case of a non-critical fault detected with poor confidence might be to fuse information with another on-board sensor, whereas detection with high confidence might be to schedule the equipment for sub-system repair.
or replacement after few more runs depending on the remaining useful life distribution.

These examples reinforce the belief that knowledge of uncertainty can improve decision-making abilities. In this work, the state-of-the-art is improved for uncertainty quantification for learning systems as well as in output uncertainty quantification for sequential decision-making tasks. The pattern classification, state estimation, and anomaly detection problems will be addressed by the developed approach for intelligent systems to detect anomalies in their own behaviors, identify targets in the environment, and estimate location for navigation, during real-time operation. This uncertainty-aware learning and decision-making framework will enable us to incorporate robustness in decision-making and safety constraints in control and execution.

1.2 Research Overview and Research Problems

This section will introduce the decision-making problems that this work will address with data-driven models. Then, a unified framework of the decision processes will be shown as an intelligent physical system. The specific research problems will be identified, motivated, and explained, which will address the learning and decision-making problems to add more intelligence in physical systems.

1.2.1 Classification of Decisions

![Classification of decisions](image)

Figure 1.1: Classification of types of decisions for data-driven systems
Decision-making problems for intelligent systems can be interpreted in many different ways, due to a wide variety of decisions we can think of in practical scenarios. In order to define the scope of this research in terms of the types of decisions to be considered, first a classification of decisions is provided in Fig. 1.1. The terminology used in this classification has been developed in the Data Fusion Information Group (DFIG) model [38] of information fusion.

**Decision for Awareness:** These include any decisions about the state of the system using tools from machine learning, detection/estimation theory, and/or data mining. This decision-making process is the forward problem for the intelligent physical system. The decisions included in this type are:

1. *Source Assessment:* Assessing performance-related metrics of a source in order to aid in model adaptation or process refinement. For e.g., sensor health, power level, context-dependent trust [3, 117], and contextual value of information.

2. *Object Assessment:* Information fusion for detection, event classification, state estimation (e.g., position, velocity, etc.), determining appearance attributes, etc.

3. *Situation Assessment:* Automated reasoning of observations for context-aware interpretation, determining link between spatio-temporal events, etc. is done as a part of this assessment for situation refinement.

4. *Impact Assessment:* This includes projection of the current situation into the future to assess threats, risks, vulnerabilities, and identify enemy intent.

**Decision for Adaptation:** This includes any decisions for adapting the system to improve performance of the forward problem using tools from optimization theory and/or control theory. This decision-making process is the inverse problem for the intelligent physical system.

1. *Sensor Network Adaptation:* The adaptation of the sensing system in the physical space by activation/deactivation of sensors.

2. *Model Adaptation:* The adaptation of the sensing system in the cyber space by changing the decision boundaries due to context and by updating the measurement models using online learning techniques [24, 130, 63].

3. *Feature Extraction Adaptation:* Usually every sensor in an intelligent sys-
tem has a feature extraction process. Adaptation of this process for individual sensors and/or adaptation of the feature level fusion process to give contextually-relevant features is considered here.

The decisions made by most of the data-driven intelligent systems can be assigned to one of the above cases. This research will focus on the following decisions for assessment: source and object assessment as well as the idea of context-aware interpretation from situation assessment; and all of the decisions for adaptation. These decision-making processes are put together in a unified framework for intelligent physical systems in the next section.

1.2.2 Schematic of an Intelligent Physical System

An intelligent physical system is represented as a cyber-physical system, where the plant or the machine is in the physical domain and decision & control-related modules are in the cyber domain. The sensor data helps to perceive the physical world and the inputs selected in the cyber domain actuates the plant physically. In the Figure 1.2, the forward flow of information and the information feedback are shown with blue and red arrows, respectively. The state of the physical system to be estimated or target to be detected or event to be classified is denoted by an unknown

![Figure 1.2: Schematic of an intelligent physical system](image-url)
variable $X$ and several (possibly) heterogeneous sensors $S = \{S_1, S_2, \cdots, S_N\}$ can be used to observe the state $X$. The switches denote that the sensor can be activated or deactivated as per decisions made for sensor network adaptation. Data from all the active sensors undergoes some form of preprocessing and feature extraction. These features can be processed individually and the local decisions are then fused for awareness of the system or the features can be fused to make more accurate and robust decisions. Various decisions or estimations for perception of the environment and the system itself are together called as decision for awareness and the different feedback mechanisms are called as decisions for adaptation (See Section 1.2.1 for more details). The models which are used to build the knowledge base of the system is also shown in the Figure 1.2. In a classification scenario, models can include representative patterns for different possible events of interest and/or classifiers to discriminate between different targets. In general, these models represent the measurement models of the sensors and target models for the underlying states of interest. These models can be updated/adapted with new knowledge using the model adaptation feedback. The output of the perception module is used by the controller to choose suitable inputs and that input is sent back to the physical domain for actuation.

### 1.2.3 Research Problems

Although, the presented general framework of an intelligent physical system itself has wide scope of in-depth research and novel contributions, few research problems have been identified keeping in mind the motivation of learning data-driven models for decision-making with uncertainty quantification. The list of these research problems is given below:

1. Problem of density estimation for data-driven modeling,
2. Problem of tractable sensor fusion,
3. Problem of information modeling of time-series data,
4. Problem of dynamic sensor selection,
5. Problem of sequential learning with imperfect information,
6. Problem of sequential decision-making with time-series data,
7. Problem of input classification for safety.

Each of these problems will be described ahead. In order to define the scope of the work, a research question will be posed for each one of these problems. Note that all these problems contribute to adding more intelligence to the intelligent physical system.

**Problem of density estimation for data-driven modeling:** The research motivation in Section 1.1 informed us that in the existing machine learning literature, there is a lack of frameworks which can simultaneously achieve (1) uncertainty quantification; (2) nonparametric density modeling; and (3) real-time inference. Thus, there is a need for development of a generative data-driven modeling framework using nonparametric density estimation to capture all the uncertainty in the data with a sparse representation model. This research specifically tries to answer the following question:

*How to develop a nonparametric model to concisely represent density of a random variable?*

and also, accomplish the following goals:

1. **Learning Error/Uncertainty Quantification:** Quantification of the uncertainty in the density estimate provided by the learning framework.

2. **Decision-making with Output Uncertainty Quantification:** Development of a decision-making framework with discrete and then continuous state spaces, which can provide the output uncertainty along with the state estimates for classification as well as regression.

The framework developed in this research learns generative measurement models as conditional probability densities of observations (Y) given states (X), that is, \(p(Y \mid X)\). The conditional density can address the classification and regressions tasks as follows:

\[
\hat{x}(y) = \arg \max_{x \in X} p(Y = y \mid X = x),
\]
where $\mathcal{X}$ is the state hypothesis set, if $\mathcal{X}$ is finite discrete set, then this is a classifier for a classification problem, whereas, if $\mathcal{X}$ is continuous set, then this is a function solving a regression problem. Moreover, this provides output uncertainty with the decisions/estimates using local measurements in real time, and hence can be used for anomaly detection as well. This framework in used for an intelligent physical system shown in Figure 1.2 to learn measurement models, information models from sensors, sensor fusion, decision processes for awareness, and to develop model adaptation techniques. Thus, density estimation is core problem of this research.

**Problem of Tractable Sensor Fusion:** Given measurements or features from several sensors reaching a consensus on state estimate by sensor fusion is the forward problem of the intelligent system, as shown in Figure 1.2. Multi-modal sensor fusion for state estimation with output uncertainty quantification can be systematically addressed in a Bayesian setting. For example, if $X$ is a random variable denoting the unknown fixed state of the system and $Y_1, Y_2, Y_3$ are measurements from three distinct sensors, then sequential update can be accomplished using the Bayes’ rule [12] as follows:

$$p(X \mid Y_1, Y_2, Y_3) = \frac{p(Y_3 \mid X, Y_1, Y_2)p(X \mid Y_1, Y_2)}{\sum_{x \in \mathcal{X}} p(Y_3 \mid x, Y_1, Y_2)p(x \mid Y_1, Y_2)}.$$

This posterior distribution captures the uncertainty in our knowledge of the state. However, this update step requires the system to know the likelihood distribution $p(Y_3 \mid X, Y_1, Y_2)$ conditioned on all previously obtained measurements, which can get intractable even for finite discrete-valued measurements of these sensors. In several works in literature, in order to simplify the math and obtain tractability, it is assumed that the different measurements are conditionally independent given the state. This is called the naïve Bayes’ assumption [12], which gives $p(Y_1, Y_2, Y_3 \mid X) = p(Y_1 \mid X)p(Y_2 \mid X)p(Y_3 \mid X)$. However, in presence of contextual effects, the naïve Bayes’ assumption does not hold [125] and wrongly assuming such independence can adversely affect the control system. Thus, the following question is important to answer for sensor fusion in practical systems:

*How to tractably fuse information from sensors without assuming conditional independence between measurements given the state?*
This research addresses the above question by introducing a latent variable called context, which gives conditional independence along with state. The latent variable is obtained by solving an optimization problem for density estimation, which guarantees the desired conditional independence by design. The details of the optimization problem and resulting tractable and accurate sensor fusion mechanism is given in Chapter 3.

Problem of Information Modeling from Time-series Data: As shown in Figure 1.2, some feature can be extracted from the time-series data in physical systems for decision-making, such as principal component analysis (PCA) [12], statistical moments [79], and symbolic time-series analysis (STSA) [91]. Particularly, STSA has been successfully implemented in several applications, such as monitoring of rotorcraft stability [109], battery health [57], and combustion stability [95], target detection with seismic sensors [128], fault detection in shipboard auxiliary systems [98, 99], robot gait modeling and classification [106], etc. In STSA, each measurement is assigned a symbol by creating a partition of the measurement space and then univariate statistics of the symbol sequences are computed to create Markov models with finite memory [72] (See Appendix B for more details). These Markov models are called information models. If the time-series data is 1D, then STSA techniques have been well-established to create a partition of that 1D measurement space, including uniform, maximum entropy [91], and symbolic false nearest neighbor partitioning [49]. The challenge is its extension to vector-valued time series that are commonly encountered in many human-engineered systems. It appears that either there is no rigorous way to extend these 1D techniques to higher dimensions or they become computationally expensive/intractable for higher dimensions. Thus, the following question needs to be answered:

How to represent information from a multidimensional time-series data using STSA?

This research proposes a nonparametric density estimation–based information-space partitioning and symbolization technique for capturing and representing the underlying statistical behavior from time-series data. This technique is described in detail in Chapter 4 and a previous version was published in [126].
Problem of Dynamic Sensor Selection: The sensor selection and sequential decision-making problems have already been discussed in literature [11, 45, 7, 110], but none of these works include the effect of operational conditions which can affect the sensor data or alter the cost of decision-making. For example, in border surveillance sensor networks the following scenarios are relevant:

- An inexpensive acoustic sensor on a calm summer day can give good human-vehicle classification results, but it will not be much of use in windy conditions; similarly, an expensive camera may not be very useful in poor visibility conditions, but provide high accuracy results on a clear sunny day.

- The penalty of misclassification of targets in certain regions can change with emergency conditions or intelligence reports.

Thus, in current literature the following question has not been answered:

*How to dynamically select the most relevant sensor team for state estimation when the environmental conditions affecting those data are unknown?*

The sensor selection framework developed in this research aims to integrate the effects of context on the measurements and the consequential impact on hypothesis testing performance in a systematic manner. A finite-horizon minimax dynamic programming formulation has been developed for solving state estimation and sensor selection problem, which guarantees optimal performance under uncertainty by minimizing the expected cost. In Figure 1.2, this sensor selection framework enables to make data-driven feedback control for network adaptation. An initial version of this dynamic context-aware sensor selection problem was published in [124] and it will be addressed in detail in Chapter 6.

Problem of Sequential Learning with Imperfect Information: When higher-fidelity sensors are activated, they might give information feedback for model adaptation to lower-fidelity sensors, as shown in Figure 1.2. This information feedback is essentially the decision made by the higher-fidelity sensor alone. This information might be incorrect as the sensor, which is acting as an instructor, is not perfect. Nonetheless, one might wonder that can the “student” sensor learn
its own true model over time by using this feedback from its imperfect instructors. In other words, can cameras train seismic sensors to become better over time by labeling their observations. If shown to be possible without using too many restrictive assumptions, this concept of learning from others has a widespread application. Although machine learning literature has few approaches to deal with label noise in training set [30, 75, 105], they are suitable only for learning in batches and not suitable for sequential learning. Thus, the question to explore is as follows:

*Can a sensor sequentially learn its correct measurement model from other imperfect sensors?*

The answer to this question was found to be *Yes*. In this research, the classical work on recursive kernel density estimation [130, 63, 64] is leveraged to develop a novel learning approach to incorporate imperfect instructions from instructors in a streaming fashion. A recursive density estimator has been derived obtain the generative model of the measurements for each discrete state. The motivation, prior work, theory, and results are shown in detail in Chapter 7.

**Problem of Sequential Decision-making with Time-series Data:** Sequential probability ratio test (SPRT) has been traditionally used for binary hypothesis testing [129, 107] and is known to be an optimal detector when the observation sequences are independent and identically distributed (IID) [14]. However, sequential data from physical systems is typically not independent as causal relations, due to physics, lead to dependencies. The time-series corresponding to nominal and faulty cases are obtained by controlled experiments and the information models of time-series data can be constructed using the STSA approach. In [35, 16, 31], the SPRT was extended for data generated from a Hidden Markov models (HMM) [12]. However, existing approaches do not specifically address sequential tests with Markov models of time-series data. Thus, the following question is considered in this research under decision-making with data-driven models:

*How to develop a sequential hypothesis test for streaming data without using the assumption of IID data?*

This problem considers a decision process in which there is only one sensor with a very high sampling rate, and the target models are already available from training
phase, and the binary states have to be estimated with streaming data during operation. To address this problem, the SPRT is extended to be used with Markov models of symbolized time-series data. It is shown that the test can achieve any user-defined performance and it can work with high-speed streaming data. This approach is explained in Chapter 5.

**Problem of Input Classification for Safety:** In several practical systems, there are fail-safe rules or safety-check conditions, which are manually designed and checked during execution for safety. In complex physical systems obtaining the list of such rules can be an onerous task and missing out on certain rules can lead to failures in some rare unforeseen situations. Thus, an intelligent system needs a systematic approach for input classification to ensure that the unsafe regions of the state space are not visited by the system. Although this problem applies to both discrete and continuous state spaces, here the state set is considered to be continuous and possibly vector-valued, making it more challenging. Before developing an input classification approach, the system needs to have a state estimation framework. The physics-based or data-driven models used to represent the measurement models of several complex systems, do not have enough flexibility to capture all the uncertainty in relationship or noise in the system, as discussed above in Section 1.1. Thus, the measurement model is learned via conditional kernel density estimation of a measurement $Y$ at a state $X$, i.e. $p(Y \mid X)$, which does not put any restriction on the structure or on the noise statistics and uses a concise model to represent the density. The measurement model is then used for tracking with particle filters [4] to obtain the posterior density over the state. After having this state estimation framework in place, the question to address in this research is as follows:

*How to develop an input classification technique for safety in physical systems?*

In this work, an input classification approach is developed to use the posterior particle density, intended input, and the motion model to infer if the estimated state of the intended input is safe. This work also propagates the uncertainty in the predicted state distribution for identifying the probability of safety of the system. The theory and results for state estimation as well as input classification
are shown in Chapter 8 and an initial version of this work on indoor localization of smart wheelchairs was published in [120].

Each of these problems has been formulated mathematically, solution approach has been developed, the proposed technique is supported by theoretical results, and the developed framework is then illustrated in simulations and/or experiments. The experimental validation in this research focuses on three application areas. These areas are explained ahead in the next section.

1.3 Applications

In this work, the research directly addresses three different applications—namely, border surveillance systems, combustion instability detection in gas turbine engines, and user-assistance for safety in wheelchair navigation. The applications touch upon several areas of study, such as multi-modal sensor fusion for state estimation, prognostics and health monitoring, human-in-the-loop systems, nonlinear filtering and estimation, and stochastic control. These application areas with problems in learning and/or decision-making will be highlighted ahead in this section.

1.3.1 Border Surveillance System

A typical border surveillance system [19] includes seismic, acoustic, and infrared sensors. These sensors are together called as unattended ground sensors (UGS). The signals from these sensors are prone to be affected by the environment [66] and

![Figure 1.3: The concept of a dynamic border control system](image)
their low-fidelity signals limit the decision-making and inference capabilities. As an alternative, a border surveillance network which has multiple layers of sensing has been proposed. In this work, the border control problem has several sensing modalities with (possibly) different fidelities. The sensors can be dynamically invited to a team of active sensors, which participates in decision-making for target classification and situation awareness. A simplified schematic to illustrate this problem is shown in Figure 1.3, where a human target is walking across a border region and the network has to detect and then classify the target as human or animal or vehicle. In the Figure 1.3, if the sensor team \{S_1, S_3\} is unable to confidently classify the target, then it can adapt the measurement system to include any one sensor from \{S_4, S_5, S_7, S_9\} to again decide between target classification or adding a new sensor. This state estimation (i.e., target classification) and sensor selection problem together is referred to as the border control problem in this work. Under border control, the specific research problems considered in this work are of sensor fusion, information modeling, dynamic sensor selection, and sequential learning from imperfect instructions.

1.3.2 Combustion Instability Detection

Combustion instability is, in general, characterized by self-sustained growth of large-amplitude pressure tones that are caused by a positive feedback arising from complex coupling of localized hydrodynamic perturbations, heat energy release, and acoustics of the combustor [95]. The prediction or early detection of combustion instability is of particular interest in gas turbine combustion for performance monitoring and fault diagnosis as well as for choosing corrective control actions. Since dynamics of the combustion system has a very fast timescale, instability detection needs to be accomplished in near real-time. In this work, a gas-turbine engine monitoring problem is considered, where the system needs to detect combustion instabilities from time-series data of pressure measurements. In this particular problem, the data comes at a sampling rate of about 8kHz and the objective is to detect instability from that high-speed streaming data. Sequential hypothesis tests for time-series data will be explored for this application.
1.3.3 User-assistance for Safety in Smart Wheelchairs

Smart/robotic wheelchairs are being developed to be used by patients with paralysis or neurodegenerative diseases, such as amyotrophic lateral sclerosis (ALS), Parkinson’s, Alzheimer’s, and Huntington’s. One such wheelchair has been custom-built by the Intelligent Vehicles and Systems Group (IVSG) at Penn State, as shown in Figure 1.4a, for research purposes. Since, a human operator is needed for controlling this wheelchair using their own perception and planning, the motor skills and hand-eye coordination of the patients might not always be suitable for joystick-based control. Electroencephalography (EEG) signals can be used to acquire the intended motion from the user. The signal processing and feature extraction module can be trained for a particular patient. If the health of the patient deteriorates and/or if the learning system misclassifies the intended motion, it can lead to severe consequences. For example, when the wheelchair is near a staircase (see Figure 1.4b), an incorrect input can cause the wheelchair to fall off the stairs. However, if the wheelchair knows the approximate location on the map and it knows the unsafe zones, such as staircases, then a user-assistance module can be incorporated on the smart wheelchair to assist the user by detecting unsafe inputs.
The input classification algorithm needs an indoor localization framework, which can provide output uncertainty over the estimated state. The localization problem is addressed by Bayesian filtering [118] using a dynamic motion model for the wheelchair and a measurement model for the sensors. Magnetometers were used to measure ambient magnetic fields and these measurements were used for localization. Research problems of conditional density estimation and input classification will be explored in this work.

These widely different applications have been used to illustrate the general applicability of the results developed in this thesis. The experimental testbeds used in the study have been explained in Appendix A.

1.4 Major Contributions of the Thesis

The major contributions of this thesis to the field of data-driven modeling, learning, decision-making, and safety for intelligent physical systems are as follows:

- Formulation of a linear optimization problem for scalable, nonparametric density estimation, using statistical learning and optimization theory. This simplifies the problem of learning the density to be easier than that of learning a classifier.

- Mathematical formalization of the notion of context as enabler of conditional independence to help with multi-modal sensor fusion and learning of unified context from multi-modal data.

- Formulation of a density-estimation scheme to learn context from sensor data, which guarantees conditional independence given state-context pair.

- Formulation of two algorithms to compress the set of contexts and obtain the corresponding error bounds using graph theory and subset selection.

- Development of a density-estimation scheme for simultaneously solving the problems of alphabet-size selection and information-space partitioning to perform symbolic time-series analysis with multi-dimensional time-series data.
• Extension of the classical SPRT for sequential hypothesis testing with Markov models of time-series data.

• Formulation of a unified sensor selection and state estimation problem as an optimal control problem, which includes spatiotemporal evolution of context and arbitrary switching of the cost structures too. The finite horizon min-max dynamic programming approach is made tractable by using machine learning approaches, such as, dimensionality reduction, feature extraction, and supervised learning.

• Development of the problem of learning from other sensors as a sequential learning task, which is solved using recursive kernel-density estimation and linear algebra. The rate of convergence is also computed as a function of number of instructions from each instructor and the instructor’s own classification accuracy.

• Formulation of a conditional density estimation problem to learn measurement models from data, which do not have restrictive assumptions on structure or noise. The measurement model is used as likelihood function in a Bayesian filter for tracking.

• Development of an input classification algorithm, which uses estimated state uncertainty, estimate of intended user input, and motion models to classify if an input is safe, before executing it on a physical system.

These theoretical contributions have been illustrated and validated with simulations and/or experiments.

1.5 Organization of the Thesis

The thesis is divided into 9 chapters. The dependency of these chapters and the important learning/estimation problems covered in each chapter are given in Figure 1.5. The dependency chart re-emphasizes the role of density estimation in the present thesis. The contents of each chapter are briefly given ahead:
Chapter dependency and learning/estimation problems in each chapter

1. Chapter 1 has introduced and motivated the various problems for intelligent physical systems related to density estimation for data-driven modeling and decision-making under uncertainty.

2. Chapter 2 develops a constrained empirical risk minimization problem for density estimation, shows theoretical analysis of the nonparametric density estimation approach, and illustrates the approach with simulation results.

3. Chapter 3 explains the notion of and formalizes the definition of context. It also develops density estimation-based technique to learn context from multi-modal sensor data. Further, a tractable and accurate sensor fusion approach has been discussed with validation experiments.

4. Chapter 4 covers the modeling of multi-dimensional time-series data using symbolic time-series analysis, where density estimation has been used to select the alphabet-size and partition the measurement space.

5. Chapter 5 develops and explains a sequential hypothesis test for models of time-series data and validates the framework with simulation and data from combustion experiments.

6. Chapter 6 presents the dynamic sensor selection framework, which uses context awareness and sensor fusion, as developed in Chapter 3, to formulate
and solve a *decision-making under uncertainty* problem of optimal sensor selection and state estimation.

7. Chapter 7 develops a sequential learning technique to obtain generative measurement models for sensors by using imperfect instructions from other sensors in the network.

8. Chapter 8 presents a new paradigm for learning measurement models for data and implements the model for tracking of robotic wheelchairs using magnetic fields. An input classification algorithm is also developed to ensure safety of the wheelchair by identifying unsafe inputs.

9. Chapter 9 summarizes the conclusions and provides few research directions for further investigation.

The chapters are followed by two appendices: one for explaining the testbeds used in various experiments in this work, and second for explaining the framework of symbolic time-series analysis, which has been used and referred in quite a few sections in this thesis. The Figure 1.5 shows that Chapter 5 and Chapter 6 are mainly focused on decision-making and rely on density estimation via Chapter 4 and 3, respectively.
Chapter 2

Nonparametric Density Estimation for Data-driven Modeling

2.1 Introduction

In the last few decades, a large number of machine learning approaches have been developed to solve the problems of anomaly detection, classification, and regression. Most of these techniques provide point estimates of the class using a classifier or of the state using a regression function, which is available to a decision-maker or a controller. However, none of these approaches for anomaly detection (one-class support vector machines), classification (support vector machines (SVM), k-nearest neighbor (KNN), neural networks, discriminant analysis, random forests, etc.), and regression (support vector regression (SVR), neural networks, etc.) provide any additional information of the uncertainty in the output. A few existing techniques in machine learning can provide output probabilities, along with the decisions, which represent the confidence associated with the decision. Platt scaling [83] for SVM fits a parametric logistic model using distance from the classifier hyperplane; linear as well as quadratic discriminant analysis assumes data in each class is Gaussian [28]; and the softmax function used in neural networks is a generalization of the logistic function [32]. However, these techniques enforce either a parametric model for output uncertainty or a parametric model for the data, which can be restrictive or even incorrect for several applications. Moreover, the discriminative
learning framework is incapable of detecting changes in the underlying distribution generating the data, as the probability density of the data cannot be captured. Thus, generative models are needed for detecting such anomalies in the physical systems. In robotics, Gaussian process (GP) is being used for data-driven modeling with uncertainty quantification [51, 132], but the complete training set is needed during inference, making it intractable. In order to make GP practical for larger training set sizes, approximate sparse Gaussian processes are being developed [85]; however, the limitation of Gaussian models for uncertainty still holds. The classical nonparametric density estimation method from Parzen [80] and Rosenblatt [92] captures complete uncertainty in the data to produce a generative model. However, using this generative model is not possible in real-time for practical data sets, because it needs to consider all the observations with equal weights. Thus, in the existing machine learning literature, there is a lack of framework which can simultaneously achieve (1) uncertainty quantification; (2) nonparametric density modeling; and (3) real-time inference.

To bridge this gap, a nonparametric density estimation technique is developed to learn generative models as sparse mixture models. This approach eliminates restriction on the structure of the relationship between state and observation (such as, linear, polynomial, etc.) as well as removes restriction on noise statistics (such as, additive, uni-modal, and/or parametric). The nonparametric density model enables uncertainty quantification as output uncertainty can be computed in the decision-making process. The sparse representation of the measurement model enables real-time inference too.

This chapter has 5 sections including the present one. In Section 2.2, the mathematical framework for learning as an optimization problem is reviewed and the density estimation problem is formulated using statistical learning and optimization theory. Section 2.3 provides some theoretical analysis for the developed formulation using optimality conditions. The implementation of the optimization problem and simulation results for anomaly detection as well as classification are given in Section 2.4. Some concluding remarks are given in Section 2.5.
2.2 Optimization Problems in Learning

Optimization is used in learning to obtain a map which corresponds to minimization of a certain cost function. Optimization problems are used in classification, (e.g., logistic regression [12], neural networks [12], and SVM [122]), regression (e.g., linear regression [12] and SVR [15]), and density estimation (e.g., support vector machine-based approaches [73, 131, 122]). In this section, a general optimization framework widely used in learning is discussed, followed by its use in formulating density estimation problems.

2.2.1 Empirical Risk Minimization

In supervised learning problems, the objective is to learn a mapping function \( f \) from input \( \mathcal{X} \) to output space \( \mathcal{Y} \), i.e., \( f : \mathcal{X} \to \mathcal{Y} \). Given a hypothesis space of several (possibly infinite number of) functions, the optimal function \( f^* \) is chosen by minimization of some suitable cost function and by using a set of training examples \((x_1, y_1), \ldots, (x_N, y_N)\). Formally, it is assumed that there is a joint probability distribution \( P \) over \( \mathcal{X} \times \mathcal{Y} \) and the training set consists of \( N \) IID samples \((x_1, y_1), \ldots, (x_N, y_N)\) drawn from \( P \). Also, a real-valued non-negative loss function \( l(y, f(x)) \) measures how much the estimate \( f(x) \) is different from true output \( y \). The risk associated with the hypothesis \( f \) is defined as:

\[
R(f) = \mathbb{E}[l(f(x), y)] = \int l(f(x), y) dP(x, y).
\]

The goal of a learning algorithm is to find the optimal hypothesis \( f^* \) from a hypothesis set \( \mathcal{H} \), i.e., \( f^* = \arg \min_{f \in \mathcal{H}} R(f) \). Usually, the probability distribution \( P \) is unknown in practical problems, thus the risk cannot be computed. However, an approximation called empirical risk can be computed as follows:

\[
R_{\text{emp}}(f) = \frac{1}{N} \sum_{i=1}^{N} l(y_i, f(x_i)).
\]
The empirical risk minimization (ERM) principle [122] states that the learning algorithm should choose a hypothesis \( \hat{f} \), which minimizes the empirical risk:

\[
\hat{f} = \arg \min_{f \in \mathcal{H}} R_{\text{emp}}(f).
\]

An optimization problem has to be solved in ERM for learning a classifier or a regression function. Several popular algorithms used in practice for regression, such as support vector regression (SVR) [108, 15], lasso regression [28], ridge regression [28], etc., are different variants of ERM with regularization. Regularization is added to avoid over-fitting, to obtain a solution with small coefficients (\( \ell_2 \)), or to obtain a sparse solution (\( \ell_1 \)), etc. The general ERM problem with regularization is given as follows:

\[
\hat{f} = \arg \min_{f \in \mathcal{H}} \frac{1}{N} \sum_{i=1}^{N} l(y_i, f(x_i)) + g(\|f\|),
\]

where \( l : \mathcal{Y} \times \mathcal{Y} \rightarrow \mathbb{R} \) is a loss function and \( g : \mathbb{R} \rightarrow \mathbb{R} \) is a monotonic regularization function. Although the function space is infinite-dimensional, computationally efficient techniques exist to identify the regularized risk minimizer, because of the Representer Theorem [18]. This theorem guarantees that an optimal solution is a linear combination of kernels placed at observations from the training dataset.

### 2.2.2 Optimization Problems for Density Estimation

Given a sample of \( N \) instances, it is assumed that the sample is IID and is drawn from a probability distribution \( P \). Assuming that \( P \) admits a density, the objective of a density estimation problem is to learn a probability density function (PDF) \( f(x) \) from which the \( N \) instances have been drawn. Recall that the density function in probability theory is defined as the solution of the integral equation

\[
\int_{-\infty}^{x} f(t)dt = F(x), \quad (2.1)
\]

where \( F(x) \) is the probability distribution function or cumulative distribution function (CDF). In the problem of probability density estimation, the training dataset
includes only the measurements \( \{x_1, x_2, \ldots, x_N\} \), with \( x_i \in \mathcal{X} \) for all \( i \), and the desired regression output is unknown; moreover, the definition of PDF is given by an integral equation. Thus estimates of CDF are needed to solve the equation (2.1).

The Glivenko-Cantelli (GC) theorem (1933) of statistical learning theory \[122\] is one of the most fundamental results of statistics. It states that the empirical CDF uniformly converges in probability to the true CDF. This theorem holds for any CDF and the distribution-free uniform estimation error bounds are known as well \[122\]. That is, we have

\[
\sup_{x \in \mathcal{X}} |F(x) - F_N(x)| \to 0
\]

in probability, uniformly over all CDFs \( F \), where \( F_N \) is the empirical CDF with \( N \) measurements, which is defined by

\[
F_N(x) = \sum_{i=1}^{N} \left( \prod_{d=1}^{n} I_{[x^d_i, \infty)}(x^d) \right)
\] (2.2)

for all \( x = (x^1, \ldots, x^n) \in \mathcal{X} \). Here, \( I_S(\cdot) \) is the indicator function on a set \( S \), and \( x_i = (x^1_i, \ldots, x^n_i) \) for \( i = 1, \ldots, N \). Although the original GC theorem assumes that the measurement sequence is IID, generalization of the GC theorem for strictly stationary measurement sequences \[121\], and for data coming from uniformly ergodic Markov chains \[134\], are also available in the literature.

Although the GC theorem shows that the convergence holds, the rate of convergence for real-valued \( x \) is given by the Dvoretzky-Kiefer-Wolfowitz (DKW) inequality \[65\]. For every \( \varepsilon > 0 \), the following inequality holds

\[
\Pr \left( \sup_{x \in \mathbb{R}} |F(x) - F_N(x)| > \varepsilon \right) \leq 2e^{-2n\varepsilon^2}.
\]

The fast rate of convergence shows that empirical cumulative density can be used practically as a good surrogate for the true unknown value of the density. Thus, a density estimation problem can be posed as a regression problem using the training set \( \{(x_i, F_N(x_i)) : i \in \{1, 2, \ldots, N\}\} \) and by addition of suitable constraints to solve the integral equation (2.1).

Density estimation is then posed as a constrained empirical risk minimization
problem,
\[
\hat{f} = \arg \min_{f \in \mathcal{H}} \frac{1}{N} \sum_{i=1}^{N} l(y_i, F(x_i)),
\]
where the hypothesis set \( \mathcal{H} = \{ f : f(x) \geq 0, \forall x \in \mathcal{X} \text{ and } \int_{\mathcal{X}} f(t) dt = 1 \} \) consists of well-defined density functions, which are always non-negative and integrate to 1. In this work, the density functions lie in a reproducing kernel Hilbert space (RKHS) denoted as \( \tilde{\mathcal{H}} \) and the associated positive-definite kernel function is \( K \). The density estimator is represented as a finite linear combination of the kernels placed at the points from the training set, i.e.,
\[
f(x) = \sum_{i=1}^{N} \pi_i K(x, x_i). \tag{2.3}
\]
Note that, if all \( \pi_i = \frac{1}{N} \), then this function is identical to the Parzen density estimator. Thus, by introducing the weights \( \pi_i \), this estimator allows assigning different weights or importance to different observations. The kernel function is required to satisfy \( \int_{\mathcal{X}} K(z, x) dz = 1 \) for any \( x \in \mathcal{X} \). Two examples of kernel functions \( K(x, \tilde{x}) \) that satisfy this requirement are given by:
\[
\text{Laplace: } \frac{1}{(2\gamma)^n} \exp \left( - \frac{\sum_{d=1}^{n} |x^d - \tilde{x}^d|}{\gamma} \right), \tag{2.4a}
\]
\[
\text{Gaussian: } \frac{1}{(2\pi\gamma)^{n/2}} \exp \left( - \frac{\sum_{d=1}^{n} (x^d - \tilde{x}^d)^2}{2\gamma} \right) \tag{2.4b}
\]
for all \( x = (x^1, \ldots, x^n) \) and \( \tilde{x} = (\tilde{x}^1, \ldots, \tilde{x}^n) \), where \( \gamma > 0 \) is the kernel shape parameter. The function in (2.3) represents a well-defined density, if the elements of the parameter vector are non-negative and if they sum up to 1.

Let \( 1_N = [1 \cdots 1]^T \), and \( \pi = [\pi_1 \cdots \pi_N]^T \) be column vectors with \( N \) elements. Let \( \tilde{\mathbf{K}} = [\tilde{K}(x_i, x_j)] \) be the \( N \times N \) matrix whose entry \((i, j)\) is equal to \( \tilde{K}(x_i, x_j) = \int_{-\infty}^{x_j} K(x_i, z) dz \) for \( x \in \mathcal{X} \) and for \( i, j = 1, \ldots, N \) and \( \tilde{K}_i \) is the \( i \)th row of the \( \tilde{\mathbf{K}} \) matrix, then the general optimization problem for density estimation is given as:
\[
\min_{\pi \in \mathbb{R}^N} \frac{1}{N} \sum_{i=1}^{N} l(y_i, \tilde{K}_i \pi) \quad \text{subject to: } \begin{cases} 
\pi \geq 0, \\
1^T_N \pi = 1.
\end{cases}
\]

The choice of loss function would give rise to different optimization problems for density estimation. The three popular loss functions used in statistical learning, shown in Figure 2.1, give rise to the following three optimization problems:

1. Absolute-error loss:

\[
\min_{\pi \in \mathbb{R}^N} \frac{1}{N} \sum_{i=1}^{N} \left| y_i - \tilde{K}_i \pi \right| \quad \text{subject to: } \begin{cases} 
\pi \geq 0, \\
1^T_N \pi = 1.
\end{cases}
\] (P1)

2. Squared-error loss:

\[
\min_{\pi \in \mathbb{R}^N} \frac{1}{N} \sum_{i=1}^{N} (y_i - \tilde{K}_i \pi)^2 \quad \text{subject to: } \begin{cases} 
\pi \geq 0, \\
1^T_N \pi = 1.
\end{cases}
\] (P2)

3. Hinge loss (or \( \sigma \)-insensitive loss):

\[
\min_{\pi \in \mathbb{R}^N} \frac{1}{N} \sum_{i=1}^{N} \max(0, |y_i - \tilde{K}_i \pi| - \sigma) \quad \text{subject to: } \begin{cases} 
\pi \geq 0, \\
1^T_N \pi = 1,
\end{cases}
\] (P3)

Figure 2.1: Popular loss functions
where $\sigma \in \mathbb{R}^+$ is a user-defined allowable error margin.

Unlike optimization problems P1 and P2, in P3 there is a user-defined parameter $\sigma$, which denotes the allowable error margin. However, for a general dataset, one cannot know \emph{a priori} the correct $\sigma$ to select in the optimization problem. Moreover, the $\sigma$-margin is not an upper bound of the error, thus it is difficult to design a guideline to pick a suitable $\sigma$. An adaptive margin hinge-loss function is proposed for the optimization problem, so that the error margin is chosen according to the dataset. The proposed optimization problem is as follows:

\[
\min_{\pi \in \mathbb{R}^N, \sigma \in \mathbb{R}} \frac{1}{N} \sum_{i=1}^{N} \max(0, |y_i - \tilde{K}_i \pi| - \sigma) + \nu \sigma \tag{P4}
\]

subject to:

\[
\begin{aligned}
\pi &\geq 0, \sigma \geq 0, \\
1^T \pi &\leq 1,
\end{aligned}
\]

where $\nu \in \mathbb{R}^+$ is a user-defined linear penalty on the size of the error margin. It seems that the problem of selection of $\sigma$ has been modified to selection of the parameter $\nu$. However, the selection of parameter $\nu$ is not arbitrary and the guidelines for selection of $\nu$ will be listed in Section 2.4 ahead. Note that $\nu$ can be viewed as a Lagrange multiplier corresponding to the inequality $\sigma \leq \sigma_{MAX}$, where $\sigma_{MAX}$ would be the user-defined parameter, which would need tuning. The parameter $\nu$ penalizes the size of the error margin and hinge-loss increases as the error margin decreases, thus the optimization problem is a trade-off between the size of error margin and the estimation error.

All the optimization problems from P1–P4 are convex, since the cost function as well as the constraint set is convex. The constrained optimization problem P4 is nonlinear, due to the presence of $\max(\cdot, \cdot)$ and $|\cdot|$ functions in the cost function. However, it can be converted to a linear program by adding more optimization variables. This linear program for density estimation is given in the next section.

### 2.2.3 Linear Program for Density Estimation (LPDE)

Let $y = [F_N(x_1) \ldots F_N(x_N)]^T$, where $1_N = [1 \ldots 1]^T$, and $\pi = [\pi_1 \ldots \pi_N]^T$ be column vectors with $N$ elements. Let $\tilde{K} = [\tilde{K}(x_i, x_j)]$ be the $N \times N$ matrix whose
entry \((i, j)\) is equal to \(\tilde{K}(x_i, x_j)\) for \(i, j = 1, \ldots, N\). The optimization problem (equivalent to P4) for density estimation is then formulated as:

\[
\min_{\sigma \in \mathbb{R}, \pi \in \mathbb{R}^N, \xi^+, \xi^- \in \mathbb{R}^N} \nu \sigma + \frac{1}{N} 1^T_N (\xi^+ + \xi^-) \tag{P5}
\]

subject to:

\[
y - \tilde{K}\pi \leq \sigma 1_N + \xi^+,
\]

\[-y + \tilde{K}\pi \leq \sigma 1_N + \xi^-,
\]

\[\xi^+ \geq 0, \quad \xi^- \geq 0, \quad \sigma \geq 0,
\]

\[\pi \geq 0, \quad 1^T_N \pi = 1,
\]

where \(\nu \in \mathbb{R}^+\) is the given weight for the error margin \(\sigma\), which is a variable to optimize.

The cost function includes a penalty on the size of \(\sigma\) and a loss term which penalizes the absolute error in estimation only if it is larger than the margin \(\sigma\). The estimates whose error is not penalized are said to be within the \(\sigma\)-tube of the desired value. This problem has in total \(3N + 1\) variables, \(5N + 1\) inequality constraints, and one equality constraint, where \(N\) is the number of measurements in the training set. The cost function and constraints are all linear, so this is a linear program and can be solved efficiently with the interior-point method \cite{47}. Note that, although the computation time of the kernel integration matrix \(\tilde{K}\) and the desired output vector \(y\) depends on the dimensionality of the information space, the optimization problem itself is not dependent on it.

### 2.2.4 Other Optimization Problems

From the problem P5, if hinge-loss term is removed from the cost function, then we will have the following linear program:

\[
\min_{\sigma \in \mathbb{R}, \pi \in \mathbb{R}^N} \sigma \tag{P6}
\]
subject to:

\[ y - \tilde{K}\pi \leq \sigma 1_N, \]
\[ -y + \tilde{K}\pi \leq \sigma 1_N, \]
\[ \sigma \geq 0, \quad \pi \geq 0, \quad 1_N^T\pi = 1. \]

This program minimizes the maximum error over the training set, i.e.,
\[ \max_{i \in \{1,2,...,N\}} |y_i - \tilde{K}_i\pi|. \] The resulting \( \sigma \) is an upper bound on the estimation error, which is obtained after solving the problem. On the other hand, if a predetermined upper bound is given, then one can solve the optimization problem from [73]:

\[
\min_{\pi \in \mathbb{R}^N} \pi^T G \pi
\]

subject to:

\[ y - \tilde{K}\pi \leq \sigma 1_N, \]
\[ -y + \tilde{K}\pi \leq \sigma 1_N, \]
\[ \pi \geq 0, \quad 1_N^T\pi = 1, \]

where \( G \) is the gram matrix, i.e., \( G = [K(x_i, x_j)] \).

### 2.3 Analysis

This section analyzes the problem formulated in P5 using tools from optimization theory to characterize the solutions of that problem. The Lagrangian for the optimization problem P5 is given as:

\[
\mathcal{L} = \nu \sigma + \frac{1}{N} 1_N^T (\xi^+ + \xi^-)
+ (\alpha^+)^T (y - \tilde{K}\pi - \sigma 1_N - \xi^+) + (\beta^+)^T (-\xi^+)
+ (\alpha^-)^T (-y + \tilde{K}\pi - \sigma 1_N - \xi^-) + (\beta^-)^T (-\xi^-)
+ \phi(-\sigma) + \delta^T(-\pi) + \lambda (1_N^T\pi - 1),
\]
where $\phi, \lambda \in \mathbb{R}$ and $\delta, \alpha^+, \alpha^-, \beta^+, \beta^- \in \mathbb{R}^N$ are the dual variables. Since $P5$ is a linear program, the linearity constraint qualification is satisfied, thus the minimizer of the linear program satisfies the Karush-Kuhn-Tucker (KKT) conditions. Using the KKT conditions, the primal and dual solutions need to satisfy the following relations for optimality:

- Stationarity and dual feasibility:

$$\frac{\partial L}{\partial \pi} = -\tilde{K}^T(\alpha^+ - \alpha^-) - \delta + \lambda 1_N = 0,$$
$$\frac{\partial L}{\partial \sigma} = \nu - 1^T_N(\alpha^+ + \alpha^-) - \phi = 0,$$
$$\frac{\partial L}{\partial \xi^+} = \frac{1}{N} 1_N - \alpha^+ - \beta^+ = 0,$$
$$\frac{\partial L}{\partial \xi^-} = \frac{1}{N} 1_N - \alpha^- - \beta^- = 0,$$
$$\alpha^+, \alpha^-, \beta^+, \beta^-, \delta, \phi \geq 0.$$  

(2.5a)  
(2.5b)  
(2.5c)  
(2.5d)  
(2.5e)

- Complementary Slackness:

$$\begin{align*}
(\alpha^+)^T(y - \tilde{K}\pi - \sigma 1_N - \xi^+) &= 0, \\
(\alpha^-)^T(-y + \tilde{K}\pi - \sigma 1_N - \xi^-) &= 0, \\
(\beta^+)^T(-\xi^+) &= 0, \\
(\beta^-)^T(-\xi^-) &= 0, \\
\phi(-\sigma) &= 0, \\
\delta^T(-\pi) &= 0, \\
\lambda(1^T_N\pi - 1) &= 0.
\end{align*}$$

(2.6a)  
(2.6b)  
(2.6c)  
(2.6d)  
(2.6e)  
(2.6f)

These relations help us understand the role of the design parameter $\nu$ and justify why one can expect that the solution vector $\pi$ is sparse. The parameter $\nu$ linearly penalizes the error margin $\sigma$, so a higher value of $\nu$ should make the margin tighter leading to more estimates outside the $\sigma$-tube. The following theorem suggests that this intuition is correct for our optimization problem.

**Theorem 2.1.** The fraction $f_e$ of the training sample whose estimate lies outside the $\sigma$-tube around the desired value is bounded above by $\nu$; that is, $f_e \leq \nu$. 


Proof. The fraction of errors is given as
\[
f_e = \frac{1}{N} \sum_{i=1}^{N} (I_{(0,\infty)}(\xi_i^+) + I_{(0,\infty)}(\xi_i^-)) .
\]
The complementary slackness conditions from (2.6c) imply
\[
f_e \leq \frac{1}{N} \sum_{i=1}^{N} (I_{[0]}(\beta_i^+) + I_{[0]}(\beta_i^-)) .
\]
Thus, by the stationarity conditions (2.5c) and (2.5d), along with (2.5b) and (2.5e), we obtain
\[
f_e \leq \frac{1}{N} \sum_{i=1}^{N} \left( I_{\left\{ \frac{1}{N} \right\}}(\alpha_i^+) + I_{\left\{ \frac{1}{N} \right\}}(\alpha_i^-) \right)
\leq \sum_{i=1}^{N} \alpha_i^+ + \alpha_i^- \leq \nu,
\]
which is the desired result. \qed

This theorem also highlights that automatic tuning of the error margin as in [15] is achieved, where \( \nu \) controls the estimation accuracy. The range of values \( \nu \) should take is \((0, 1]\). If we had \( \nu > 1 \), then the inequalities \( \alpha_i^+, \alpha_i^- \leq 1/N \) (from (2.5c), (2.5d), and (2.5e)) and equalities \( \alpha_i^+, \alpha_i^- = 0 \) (from (2.6a) and (2.6b)) holding for all \( i \) would imply that \( \phi > 0 \) (from (2.5b)), so the complementary slackness condition (2.6d) would give \( \sigma = 0 \). That is, if \( \nu > 1 \), then the optimization problem reduces to mean absolute error minimization, i.e., P1.

In order for the proposed density estimation technique to be useful for real-time decision-making, the solution vector \( \pi \) must be sparse. In view of (2.6e), we know that \( \pi_i = 0 \) if \( \delta_i > 0 \), and that \( \pi_i > 0 \) only if \( \delta_i = 0 \). Thus, sparsity of \( \pi \) is bounded above by the number of zero entries in \( \delta \). Suppose \( \delta_i = 0 \) for some \( i \). Then, it follows from (2.5a) that
\[
\lambda = \tilde{K}_i^T (\alpha^+ - \alpha^-),
\]
where \( \tilde{K}_i \) is the \( i \)th column of matrix \( \tilde{K} \) and this matrix depends on the set of all measurements as well as the chosen kernel. As \( \tilde{K} \) needs not have a special structure,
irrespective of $\alpha^+, \alpha^-$, and $\lambda$, it is expected that the equality (2.7) will not be satisfied too often, thus the solution vector $\pi$ is expected to be sparse. This has been verified in simulation; a few of these results are discussed in Section 2.4. Note that, in order to induce sparsity in $\pi$, the usual $\ell_1$-norm minimization approach cannot be used, as the $\ell_1$ norm of $\pi$ is constrained to be equal to 1.

2.4 Implementation and Simulation Results

2.4.1 Optimization Parameter Selection

The guidelines for selection of $\nu$ parameter in P4 and P5 are given as follows:

1. Since $\nu$ is an upper-bound on the fraction of errors according to Theorem 2.1, the value of $\nu$ must be chosen from $(0, 1]$.

2. The $\nu$ parameter indicates the fraction of training points which will not satisfy the error margin $\sigma$, thus small value of $\nu$ will give few errors, $\sigma$ will be increased accordingly in the optimization; however, that $\sigma$ margin will be representative of the error bound for $(1 - \nu)$ fraction of the training points.

The guidelines for selection of the kernel parameter in all problems P1–P7 are given as follows:

1. The kernel parameter is chosen using cross-validation by doing a grid search over several parameters. The cross-validation uses some performance metric, such as accuracy, in case of classification problems.

2. Usually in P4 and P5, decreasing the kernel parameter reduces the $\sigma$ parameter and increases the number of components. Thus, a trade-off in accuracy with model complexity can be used to select the best $\gamma$ parameter. In general, a score similar to Akaike information criterion (AIC) [12, 2] can be used to capture complexity and accuracy for model parameter selection. In this work, a multi-objective optimization problem is formulated ahead for kernel parameter selection, and user preferences are used to choose a solution from the Pareto front.
2.4.2 MATLAB Implementation

The optimization problems given in this chapter are implemented in MATLAB using several different solvers and packages. MATLAB has an Optimization toolbox which includes a linear program solver \texttt{linprog} and a quadratic programming solver \texttt{quadprog}. In this work \texttt{linprog} was used to develop one of the implementations. MATLAB also supports CVX [34] (a popular package for convex optimization) and YALMIP [61]. Both of these free packages enable rapid prototyping of optimization problems by easy-to-code optimization formulations; however, these packages need solvers to obtain the solution of the specific problems. The solver used for implementation was MOSEK [71]. The details of the implementation are given ahead in this section.

The optimization problem given in \textbf{P5} is a linear program, which needs the kernel integration matrix \((K)\) and the empirical density estimates \((y)\) to be computed before the solver is called. This computation leverages the MATLAB parallel computing toolbox as the computation for each matrix entry is independent. The computation time for this step is given as \(O\left(\frac{N^2 d}{W} + s(W)\right)\), where \(N\) is the number of training points, \(d\) is the dimensionality of the measurement, \(W\) is the number of

![Figure 2.2: Time to solve LPDE versus number of training points](image)

Figure 2.2: Time to solve LPDE versus number of training points
parallel workers, and the function \( s(W) \) is the synchronization overhead of having \( W \) workers communicate and update each other. The linear program is then coded in MATLAB and uses \texttt{mosekopt} function from the MOSEK toolbox. It uses the interior-point algorithm [47] and uses parallel computing by executing the algorithm on multiple threads for fast computation. A typical runtime versus number of data points plot is shown in Figure 2.2. The plot is generated by solving the linear program with different numbers of training points and for each case repeating the optimization 30 times by using different kernel bandwidths. The observed time values are then fitted with an order 3 polynomial with the bias term set to 0 and we get R-squared fit of 0.9989. This strong fit matches with the literature, which mentions that the time complexity of the interior point method using Karmarkar's algorithm for linear programs is \( O(N^{3.5}L) \), where \( L \) is the number of bits in the input. The coefficient of \( N^3 \) in the fit (with 95% confidence bounds) was found to be \( 5.307 \times 10^{-9} \) (\( 3.094 \times 10^{-9} \), \( 7.521 \times 10^{-9} \)). This implies that the cubic growth factor becomes prominent after 1000 training points. The CVX and YALMIP packages check for correctness of the structure of the problem, find if there is a special structure in the problem, and then parse the problem for the appropriate solver, thus it is expected to take more time than the direct MOSEK implementation. The implementation with \texttt{mosekopt} in MATLAB with parallel computing toolbox was found to be the fastest of all the implementations and was used to generate the results throughout this dissertation.

### 2.4.3 Problem I: Density Estimation

The LPDE formulation does not have any explicit regularization, however, the kernel parameter controls whether the solution will have over-smoothing or over-fitting. In order to understand this statement, we consider a density estimation problem with 1000 measurements sampled from a mixture of Gaussian and Laplace distribution. The LPDE problem P5 is solved with 100 different kernel bandwidths chosen by uniformly dividing the interval of \( 10^{-3} \) and 10 in logarithmic (base 10) scale. The value of \( \nu \) fixed at 0.01. The optimized error margin and number of components for the resulting density estimates is shown in Figure 2.3 to illustrate the effect of kernel bandwidth. The distance between the estimated density and
true model in terms of Kullback-Leibler (K-L) divergence [17] is shown in Figure 2.4. Moreover, in order to illustrate the benefits of using this approach, a comparison of this distance is shown with that from Parzen density estimation [80] in the same figure. Although Parzen density estimate has lower error (e.g., 1.397 in lieu of 3.21 for $\gamma = 0.89$), it uses all the 1000 measurements to arrive at the
Figure 2.5: Pareto front

density estimate; whereas, the proposed approach used only 22 observations from the training set.

The knowledge of test error will not be available in practice, thus some parameter selection rule is needed to choose the kernel parameter, as discussed in Section 2.4.1. A multi-objective optimization to trade-off accuracy and model complexity is formulated for kernel parameter selection. The optimal parameter is the minimizer of a convex combination of error margin ($\sigma$) and fraction of components ($f_n$), that is, 

$$
\gamma^*(\alpha) = \arg\min_{\gamma} \alpha \sigma(\gamma) + (1 - \alpha) f_n(\gamma).
$$

In Figure 2.4, a Pareto front of solutions for $\gamma^*$ is obtained by sweeping over all values of $\alpha \in [0,1]$. The test error corresponding to Pareto-optimal solutions is shown in Figure 2.5. Out of the 100 values of $\gamma$ in this simulation study, only 10 solutions were found to be on the Pareto front. The density estimate corresponding to these optimal solutions is shown in Figure 2.6. Note that, for smaller values of $\gamma$, the model is over-fitting the data, and for higher values of $\gamma$, the model has over-smoothing and less accuracy. For example, the density estimate with $\gamma = 0.61$ has 22 components (i.e., $f_n = 0.022$); whereas, the estimate with $\gamma = 0.006$ has 591 components (i.e., $f_n = 0.591$). Thus, the user has flexibility to choose a solution on the Pareto front, which best serves the desired purpose.
Figure 2.6: All Pareto-optimal solutions for Problem I
2.4.4 Problem II: Anomaly Detection and Classification

Consider a binary classification problem for target detection. The training set has 400 observations for each class and the above density estimation approach is implemented for each set for several values of kernel bandwidths. The result of the learning process is shown in Figure 2.7a for class 1 and Figure 2.7b for class 2. The fraction of components denotes the ratio of the number of nonzero values of $\pi_i$ and the size of the training set, which shows that sparse mixture models can be obtained from the framework. The error margin denoted by $\sigma$ in P4 is an estimation error for the obtained density estimate. Figure 2.7c shows the true density (dashed line) and the estimated density (solid line). The kernel bandwidth was selected with a heuristic rule minimizing the sum of error margin and fraction of components.
For observation $y = 0.646$, the density estimates give $p(y|x = 1) = 0.347$ and $p(y|x = 2) = 0.008$ as the likelihood, thus the output uncertainty, using Bayes’ rule and uniform priors, is given as $p(x|y = 0.646) = [0.976, 0.024]$. Similarly, $p(x|y = 3.4347) = [0.0006, 0.9994]$. Moreover, these generative models enable us to detect anomalies, as the data not closer to the training set has very small likelihood in both the models (example: $y = -4$ and $y = 7$), and thus can be used to infer anomaly in the observations. This also shows that any classifier can be augmented with a region in which it can make confident decisions, and the agent can be aware of its decision-making capabilities by knowing its region of trust.

### 2.5 Conclusion and Future Work

In this chapter, a nonparametric density estimation technique using constrained empirical risk minimization was formulated. A linear program for density estimation was proposed and implemented in MATLAB. The optimization problem was implemented with an off-the-shelf solver and the computation time was studied to understand the time complexity. The optimality conditions were used to analyze the proposed method to show that the density estimates obtained are accurate and sparse. A multi-objective optimization was proposed and illustrated with simulation for kernel parameter selection. The developed technique was compared with Parzen density estimation in a simulation example and it was inferred that Parzen density estimate can have better modeling accuracy, but the model complexity is very high as it uses all the data to represent the estimate. This method was simulated to illustrate the data-driven modeling, classification, and anomaly detection capabilities, where the output uncertainty was also available. Thus, a practical density modeling technique, which is as simple as learning a classifier, was developed in this chapter to provide data-driven models for intelligent systems.

In this work, the time complexity of optimization is not affected by dimensionality of the dataset; however, the optimal kernel selection process can get computationally intensive for large and high-dimensional datasets. The learning of kernel function for multi-variate datasets using concepts from [68] will be explored in the future. This research did not dwell into numerical optimization aspects to study the convergence and to further optimize the computation for this particular prob-
lem for density estimation. This exercise will be carried out in the future using recent results with interior-point methods [33].
Learning Context and Context-aware Sensor Fusion

3.1 Introduction

Intelligent systems have to make decisions at various time scales for situation awareness of the system and for adaptation, using the multi-modal sources of information by a suitable fusion mechanism. In this process of monitoring and decision-making, various man-made or natural factors affect the sensor data or its interpretation apart from the statistical behavior or event of interest in the observed system. These factors are collectively called as context [81, 127]. Considering the importance of correct interpretation of data in the decision-making process, ignoring context can lead to poor performance and can even have severe consequences. Let us consider several intuitive examples of context for intelligent physical systems in the following list:

1. Ambient light intensity and fog affect the data frames being processed from the camera and the initial chosen set of features for lane detection might perform poorly, if the ambient conditions change [41].
2. Ultrasonic sensors, which are used for proximity detection, are affected by change in temperature as the speed of sound in air changes by 1.8% for a $10^\circ C$ change of temperature.
3. LIDAR sensors are affected by the reflectivity, absorptivity, and transparency
of the surface as the reflected light intensity depends on these parameters.

4. Sensors monitoring gas turbine engines in airplanes observe different fault patterns during ascent/takeoff of the flight as compared to the cruise mode [97].

5. The response from seismic sensors for the same event can be totally different before and after precipitation as the response depends on soil moisture, porosity, etc. [66].

6. The response from ground penetrating radar, which is used for detection of buried objects, is affected by soil moisture, surface roughness, and the amount of subsurface clutter [88].

7. The activity performed by the patient (walking, sitting, sleeping, jogging, etc.) is going to affect the response of the sensor. High heart rate during one activity would be an alarming situation, while during another it would be expected to be normal.

8. If the border patrol agents have information about possible drug smugglings in a particular region of the border, the penalty for mis-detections of human or vehicle targets can be increased for the unattended ground sensors in that area. Thus, the decision boundary is shifted to allow more false alarms but avoid missing any detections.

9. If a particular high modality sensor in a network is down for maintenance, then the lower modality sensors will not be able to activate it to gather more information. A measurement which would have triggered a sensor activation action would now be classified to one of the possible target classes, hence the interpretation of that data was changed due to the additional information.

A wide variety of examples of contexts have been listed above some of which have a few fundamental differences. This motivates to classify the context depending on the way it affects the decision-making process. Context can be classified as intrinsic context, i.e., factors which directly affect sensor measurements for a given event (e.g., 1-7), and extrinsic context, i.e., factors which do not affect the sensor measurements directly, but do affect the interpretation of the collected data (e.g., 8-9). The intuition behind these concepts can be understood using two figures. In Figure 3.1, a binary classification problem is shown in which the extracted features from a sensor are shown in some feature space along with the optimal decision
Figure 3.1: Concept of intrinsic context-based classifier adaptation

boundary. These features shift in the space due to change in intrinsic context, since this context affects the sensor data. The original decision boundary is not optimal anymore and needs to be adjusted to give acceptable performance. On the other hand, in Figure 3.2, a binary classification problem with a decision boundary which is optimal for a given nominal classification cost structure is shown. Extrinsic context influences the cost structure, in order to penalize the misclassification of Class 2 severely, hence the classifier shifts due to extrinsic context. It is important to note that for the same features, the space in between the old and new classifiers would now be classified as Class 2 instead of Class 1. This highlights the point that extrinsic context leads to change in interpretation of data without affecting the data itself.

In the literature [26, 27, 78], the word “context” has been used in several different ways. It is usually task-specific in nature, and often differs across sensing modalities. For example, research in image processing generally assumes the visual scene to be the context for object recognition [78]; for natural language processing tasks such as speech recognition, handwriting recognition, and machine translation, the intended meaning of an ambiguous word might depend on the text which precedes the word in question, thus the preceding text would be considered as context [93]; and, for ubiquitous or mobile computing, the context set consists of the user location as well as activity attributes [101]. In a multi-sensor operational environment, involving a multitude of sensing modalities, a broad unified notion of
context is needed. This notion should characterize situations in the physical, electronic, and tactical environments that affect the acquisition and interpretation of heterogeneous sensor data for machine perception and adaptation. Furthermore, it is often necessary to iteratively update the belief about the spatio-temporal context automatically and treat it as a latent variable to be estimated. Different clustering techniques [81, 29, 127] and mixture modeling methods [90] were previously developed and used to identify the context set from measurements.

In this chapter, after mathematically formalizing the intuition of context as discussed above, the main focus is to present an unsupervised context-learning approach that addresses, or mitigates, the aforementioned issues. This approach is based on the postulation that the context of a system, along with the system state, completely conditions sensor measurements. That is, extending the common, but often incorrect, assumption that the measurements are conditionally independent given the system state, we hypothesize that the sensor measurements are independent conditioned on the state-context pair. This postulation allows for a definition of context that is application-specific, and yet uniform across different sensor modalities. The arbitrary nature of clustering and mixture modeling approaches is avoided through a kernel-based unsupervised context learning, where the context set and a context-aware measurement model are automatically generated by the machine. Moreover, this chapter answers the question posed in Chapter 1 of tractable and multi-modal sensor fusion by introducing the notion of context to obtain a context-aware sensor fusion approach, which enables tractable
and accurate information fusion.

In this section, the intuition for context, context-aware adaptation, and types of context are explained with several examples. Section 3.2 formalizes the notion of context as used in this research to allow machines to derive context from data. Section 3.3 gives the problem of learning context from data as an optimization problem of density estimation, which gives a finite set of contexts. Section 3.4 gives two different approaches to control/compress the size of the context set and illustrates the efficacy of compression by means of experimental results with seismic sensors for target classification. Section 3.5 explains the context-aware sensor fusion process for classification and illustrates the benefits of context-awareness in target classification with seismic sensors.

3.2 Mathematical Definition of Context

Let \( S \) be a nonempty finite set of sensors, possibly with different modalities, and let \( X \) be the random system state that takes values in a finite set \( \mathcal{X} \). For each sensing modality \( s \in S \), let \( Y(s) \) be the random measurement, or the feature vector obtained after preprocessing, filtering, and/or dimensionality reduction, associated with the observation of the system state \( X \) from sensor \( s \). Before introducing a modality-independent context notion suitable for unsupervised, machine-generation of the context set, let us present a modality-specific context definition, and context types (i.e., intrinsic and extrinsic contexts), that are suitable for supervised learning.

**Definition 3.1.** (Context Elements) For each \( s \in S \), let \( \mathcal{L}(s) \) be a nonempty finite set of labels. Each element of \( \mathcal{L}(s) \) is called a context element. Every context element is a natural or man-made physical phenomenon, which is relevant to the sensing modality \( s \) used to observe the system state. It is assumed that the context elements are enumerated in \( \mathcal{L}(s) \) in such a way that no two elements can occur simultaneously.

The assumption in this definition is not restrictive. If it is possible for two context elements \( l \) and \( m \) to occur simultaneously, then a new context element \( k \) representing \( l \) and \( m \) occurring together can be added to \( \mathcal{L}(s) \). For \( s \in S \) and
\( l \in \mathcal{L}(s) \), let \( p(Y(s) \mid X, l) \) be the probability density of sensor measurements of modality \( s \) for the state \( X \) under a given context element \( l \).

**Definition 3.2. (Extrinsic and Intrinsic Subsets of Contexts)** For \( s \in \mathcal{S} \), a nonempty set \( \tilde{C} \subseteq \mathcal{L}(s) \) is called extrinsic relative to the state \( X = x \) and its measurement \( Y(s) = y \) if

\[
p(y \mid x, l) = p(y \mid x, \tilde{l}) \quad \text{for all } l, \tilde{l} \in \tilde{C}.
\]

Otherwise, the set \( \tilde{C} \) is called intrinsic relative to the state \( X = x \) and its measurement \( Y(s) = y \).

It is at times impractical to precisely distinguish extrinsic context elements from intrinsic ones. If the observation densities are overlapping and very close to each other under different context elements, then it is deduced that these context elements have nearly the same effect on the sensor data. Thus, an alternative approach is to obtain sets of context elements that are approximately indistinguishable for a given threshold parameter \( \varepsilon > 0 \) and a metric \( d(\cdot, \cdot) \) on the space of observation densities, and let them define contexts.

**Definition 3.3. (Modality-Specific Context and Context Set)** For \( s \in \mathcal{S} \) and \( x \in \mathcal{X} \), let \( \mathcal{C}(s, x) \) be a set cover of \( \mathcal{L}(s) \). Then, the collection \( \mathcal{C}(s, x) \) is called a context set and each (nonempty) set \( c(s, x) \in \mathcal{C}(s, x) \) is called a context provided that \( c(s, x) \) is a maximal set satisfying the following condition:

\[
d\left(p(Y(s) \mid x, l), p(Y(s) \mid x, m)\right) < \varepsilon \quad \text{for all } l, m \in c(s, x).
\]

In order to obtain a context set \( \mathcal{C}(s, x) \) based on Definition 3.3, the set \( \mathcal{L}(s) \) of all context elements must be known \textit{a priori}, in which case a supervised context modeling approach [81] can be used to reduce the problem of context learning to that of finding all maximal cliques in an undirected graph [13]. However, in many cases, the set \( \mathcal{L}(s) \) is unknown, and thus unsupervised context modeling techniques must be used to directly obtain \( \mathcal{C}(s, x) \) from the data. Several works of research have modeled contexts as the clusters of features together in some feature space and have used techniques such as fuzzy C-means clustering [29], Gaussian mixture models (GMM) [89], graph theoretic clustering [81] using [77], and Dirichlet process
mixture models [90] for learning the underlying structure in the feature space. The resulting context sets are modality-specific as in Definition 3.3. Although this notion of context is intuitive, it is also modality-specific, and thus it can learn context only for a particular sensor. This approach has two limitations:

1. Usually given only one sensor, it might be difficult or even impossible to separate the effect of context and state. Hence, learning context from data will be difficult.

2. If there are two or more different modalities of sensors in a network, their definition of context will be incoherent or independent. Consequently, heterogeneous sensor fusion will not be able to leverage the benefit of the knowledge of context.

The rest of this subsection is aimed at presenting an alternative definition of context, which facilitates learning a unified, modality-independent context set from a multi-modal sensor set [125]. This approach of context learning does not need a defined set of context elements and thus it is an unsupervised way of context modeling. Since different sensors might have disparate contextual sensitivities, it is possible to separate the effect of state and context by using several sensors/modalities together in the new framework. Moreover, the notion of context is unified across the sensor network, so the sensors can leverage the context-awareness for information fusion.

Let $Y_1 = Y(s_1)$ and $Y_2 = Y(s_2)$ be random measurements of the state $X$ from sensors $s_1, s_2 \in S$. For example, $Y_1$ and $Y_2$ are finite-length seismic sensor recordings from two different sensor locations or time periods, and $X$ is a binary variable representing whether a target/intruder is present or absent. Let $p(Y_1, Y_2 \mid X)$ denote the joint likelihood function of the pair $(Y_1, Y_2)$. For $i = 1, 2$, let $p_i(Y_i \mid X)$ denote the marginal likelihood function of $Y_i$. A common practice in sequential, statistical inference tasks is to assume, for the sake of convenience, that the measurements are statistically independent conditioned on the state [52]. Clearly, this assumption is incorrect unless the state $X$ completely determines all
factors that condition the measurements. That is, in general, we have

\[ p(Y_1, Y_2 \mid X) \neq p_1(Y_1 \mid X)p_2(Y_2 \mid X). \]

For example, two seismic sensor measurements in binary location testing are expected to be correlated, even if they are conditioned on the true location of a target, because the location alone does not specify the target type, soil conditions (e.g., moisture and porosity), etc., that affect seismic sensor measurements.

Therefore, we define the context as a parameter that, together with the system state, completely conditions the measurements.

**Definition 3.4. (Context and Context Set)** Suppose that the measurements \( Y_1 \) and \( Y_2 \) take values in \( \mathcal{Y}_1 \) and \( \mathcal{Y}_2 \), respectively. Suppose that the state \( X \) takes values from a finite set \( \mathcal{X} \). Then, a nonempty finite set \( \mathcal{C}(X) \) is called the context set and each element \( c \in \mathcal{C}(X) \) of the set is called a context, if the measurements \( Y_1 \) and \( Y_2 \) are mutually independent conditioned on the state-context pair \( (x, c) \) for all \( x \in \mathcal{X} \) and for all \( c \in \mathcal{C}(X) \).

According to this definition, the following relation holds:

\[ p(Y_1, Y_2 \mid X, c) = p_1(Y_1 \mid X, c)p_2(Y_2 \mid X, c) \quad \text{for all } c \in \mathcal{C}(X). \tag{3.1} \]

Here, the left-hand side of (3.1) denotes the conditional density of \((Y_1, Y_2)\) given \((X, c)\), and the right-hand side gives the product of conditional densities of \(Y_1\) and \(Y_2\) given \((X, c)\). It is now of interest to generate a context set \( \mathcal{C}(x) \) for each \( x \in \mathcal{X} \), so that (3.1) holds.

### 3.3 Learning Context from Data

In general, it is a difficult task to identify the context set and a probability distribution on it, such that the prior information about all possible contexts and the true context is correctly represented by the measurement model. In fact, it is unclear whether there even exists a nontrivial variable \( C \) that satisfies an equation of the form (3.1). This difficulty is addressed using a special type of mixture models, where each component density is a product of marginal component densities.
Suppose that the measurements $Y_1$ and $Y_2$ take values in $\mathcal{Y}_1$ and $\mathcal{Y}_2$, respectively. Suppose that the state $X$ takes values from a finite hypothesis set $\mathcal{X}$, and that, conditioned on $X$, the context $C$ takes values in a finite, state-dependent context set $\mathcal{C}(X)$. Then, a context-aware measurement model gives a likelihood function of the form

$$p(Y_1, Y_2 \mid X) = \sum_{c \in \mathcal{C}(X)} \pi_c(X) p(Y_1, Y_2 \mid X, c)$$

$$= \sum_{c \in \mathcal{C}(X)} \pi_c(X) p_1(Y_1 \mid X, c) p_2(Y_2 \mid X, c), \quad (3.2)$$

where $\pi_c(X)$ is the prior probability that $C = c$ conditioned on $X$. It is immediate from (3.2) that the marginal likelihoods

$$p_i(Y_i \mid X) = \sum_{c \in \mathcal{C}(X)} \pi_c(X) p_i(Y_i \mid X, c)$$

for $i = 1, 2$. Note that in (3.2), the component densities are products of marginal component densities. For example, Gaussian mixture models with block diagonal covariance matrices are of this type of mixture model. More specifically, we propose that mixture models of the form (3.2) be used conditioned on the state $X$, where the context set is of finite cardinality $N(X)$:

$$\mathcal{C}(X) = \{1, 2, \ldots, N(X)\}. \quad (3.3)$$

Conditioned on the state $X$, the latent variable plays the role of a machine-defined context variable $C$ that takes values in $\mathcal{C}(X)$ and satisfies the conditional independence requirement (3.1) by construction.

If the marginal component densities $p_i(Y_i \mid X, C)$ are assumed Gaussian, then the expectation maximization algorithm [25] or the variational Bayesian method [89] can be used to obtain a mixture model of the form (3.2). In this case, the number of contexts $N(x)$ may be determined for each state value $x \in \mathcal{X}$ based on a model selection criterion such as the Akaike and Bayesian information criteria [2, 104]. Alternatively, a Dirichlet process prior can be put over $N(X)$ and then a Gaussian mixture density model can be estimated together with the op-
timal number of component densities [87]. However, these parametric estimation approaches do not scale up to high-dimensional measurement spaces, especially with small sample sizes, and also their applicability is limited to Gaussian component densities.

In order to address these issues, we shall use a kernel-based non-parametric method. A kernel function defines an inner product on an implicit, possibly infinite-dimensional, feature space. The standard topology of such a feature space is that of the reproducing kernel Hilbert space induced by a (continuous) Mercer kernel [102, 18]. On the other hand, it is shown in [56] that, if one uses a discontinuous kernel, the resulting feature space can be taken to be the space \( \ell^2 \) (of square-summable sequences) endowed with its weak topology [48]. Let \( K : (\mathcal{Y}_1 \times \mathcal{Y}_2)^2 \rightarrow \mathbb{R} \) be a kernel function of the form

\[
K \left( \begin{bmatrix} s_1 \\ s_2 \end{bmatrix}, \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} \right) = K_1(s_1, y_1)K_2(s_2, y_2), \quad (3.4a)
\]

with

\[
\int_{\mathcal{Y}_i} K_i(s_i, z_i) \, dz_i = 1, \quad (3.4b)
\]

for \( s_i, y_i \in \mathcal{Y}_i \) and \( i = 1, 2 \). Then, conditioned on the state \( X \), the density estimation problem \( P_5 \) from Chapter 2 with kernel \( K \) leads to a mixture model of the form

\[
p(Y_1, Y_2 | X) = \sum_{c=1}^{N(X)} \pi_c(X) K_1(s_1^{(c)}(X), Y_1)K_2(s_2^{(c)}(X), Y_2) \quad (3.5)
\]

where \((s_1^{(c)}(X), s_2^{(c)}(X))\), \( c = 1, \ldots, N(X) \), are the representative vectors chosen by the machine from the available data, and the number of representative vectors \( N(X) \) can be controlled by tuning the underlying penalty parameter \( \nu \) or the kernel shape parameter too.

Note that, with (3.3) and

\[
K_i(s_i^{(c)}(X), Y_i) = p_i(Y_i | X, C) \quad (3.6)
\]
for $i = 1, 2$, the kernel-based model (3.5) leads to a mixture model of the desired form (3.2), and that the representative vectors can be taken to be the machine-defined contexts, provided that, in addition to (3.4), the extra constraints

\[
\sum_{c=1}^{N(X)} \pi_c(X) = 1 \tag{3.7a}
\]

and

\[
\pi_c(X) \geq 0, \quad c = 1, \ldots, N(X) \tag{3.7b}
\]

are satisfied. In the current framework, a mixture model needs to be obtained in this fashion for every value of $X$, and so in general the context set depends on $X$. Thus, the context set can be computed by solving the linear program $P5$ for each $x \in \mathcal{X}$, which already incorporates the constraints (3.7) in the optimization problem. The kernel used in the optimization needs to satisfy (3.4) and then the resulting mixture model will be a joint density conditioned on the state $X$.

The state-dependent context set, context priors, and context-aware measurement model are obtained by solving a single linear optimization problem. Directly constraining the size of context set makes the optimization non-convex and combinatorial, thus other methods for controlling the cardinality of context sets is explored ahead.

### 3.4 Cardinality Reduction of Context Sets

The size of the context set directly affects the memory required and computation time of the context-aware decision-making approaches, such as context-aware sensor selection [124] and pattern recognition [81]. In wireless sensor network applications, such as border surveillance, where power, memory, and execution time are severely constrained, we need to be able to restrict the size of context sets to enable tractable execution of context-aware approaches on resource-constrained platforms. Thus, we are exploring different approaches for context set compression. The context learning approach in [125] relies on a convex optimization formulation using the concept of kernel-based density estimation [73]. Thus, adding any explicit sparsity constraint makes the problem non-convex and hard to solve. Moreover,
enforcing strict sparsity constraint on the solution can severely affect the accuracy of the solution as model order and accuracy are known to be competing objectives [12], and one might have to repeat the non-convex optimization several times before obtaining a solution with acceptable error. This motivates the need for augmenting the original convex optimization formulation with a separate compression step in which the additional maximum error incurred is directly controlled.

This section will explain the two proposed techniques for cardinality reduction of context sets using the maximal clique enumeration algorithm from graph theory and a simple subset selection approach. Both techniques assume that the density estimation problem for unsupervised context learning has already been solved and use that estimate for reduction.

### 3.4.1 Compression by Maximal Clique Enumeration

In graph theory, a clique is a complete subgraph, and it is maximal if it is not contained in a bigger clique. Enumeration of maximal cliques is a classical problem in graph theory, which was addressed in detail in [13, 119] using depth-first search strategy. This concept is used to identify all machine-derived contexts whose effect on sensor measurements is almost identical. The context set is used as the vertex set of a weighted graph and the edge weights denote the pairwise distance between contextual observation densities. This proposed approach is explained in detail ahead in this section.
Algorithm 1: Context Set Compression by Maximal Clique Enumeration

Input: Observation densities $p(Y \mid X, L)$ and threshold $\epsilon$.
Output: Context set $C(X)$.

1 for all $x \in X$ do
2 Compute weight matrix $W(x)$;
3 $G_{x,\epsilon} = \text{ConstructGraph}(W(x), \epsilon)$;
4 $M = \text{MCE}(G_{x,\epsilon})$;
5 $C(x) = \text{Minterms}(M)$.

If $l_1, l_2, \ldots, l_{|L(x)|}$ are distinct machine-derived contexts for the state $x \in X$, the algorithm defines a weight matrix $W(x) = [w_{ij}(x)] \in \mathbb{R}^{L(x) \times |L(x)|}$ by

$$w_{ij}(x) = d(p(Y \mid X = x, L = l_i), p(Y \mid X = x, L = l_j))$$

for $x \in X$ and $i, j = 1, \ldots, |L(x)|$, where $Y = (Y_1, Y_2, \ldots, Y_N)$ is the concatenated measurement from all sensors. Here, denoted by $d(\cdot, \cdot)$ is a distance function on the space of observation densities, such as symmetric Kullback-Leibler divergence [40] and Bhattacharyya distance [1]. For a chosen positive real number $\epsilon$, let $G_{x,\epsilon}$ denote the $\epsilon$-context graph for state $x \in X$, which is defined by the tuple $(L(x), E(x, \epsilon))$, where the vertex set $L(x)$ represents the set of all machine-derived contexts corresponding to the state $x \in X$ and the edge set is given as

$$E(x, \epsilon) = \{(l_i, l_j) \in L(x)^2 : w_{ij}(x) \leq \epsilon, i, j = 1, \ldots, |L(x)|\}$$

for each $x \in X$. This graph $G_{x,\epsilon}$ is constructed in the ConstructGraph function. The edge set $E(x, \epsilon)$ represents all pairs of context whose measurement densities are at most distance $\epsilon$ away from each other. This graph $G_{x,\epsilon}$ is then processed by the Maximal Clique Enumeration function, i.e., MCE, which implements the depth-first search strategy given in [119], to obtain the set of all maximal cliques denoted by $M$. Each maximal clique is a subset of the context set consisting of contexts which are all mutually at most distance $\epsilon$ away from each other. The maximal cliques will form a set cover of the set $L(x)$, but they can end up being overlapping, thus denoting each clique as a context can lead to the loss of the desired conditional independence property. Moreover, it is known that for an
The maximum number of maximal cliques is given by $3^{n/3}$ [70], thus the resulting context set might become exponentially larger. Hence, the function $\text{Minterms}(\mathcal{M})$ uses the method in [6] to evaluate all minterms of $\mathcal{M}$ (i.e., nonempty set differences and intersections formed by the members of $\mathcal{M}$) to obtain a mutually exclusive and exhaustive collection $\mathcal{C}(x)$ of cliques that partition $\mathcal{L}(x)$; for example, $\text{Minterms}({\{1, 2, 3, 5\}, \{2, 4\}})$ gives $\{1, 3, 5\}, \{2\}, \{4\}$. These steps are given in the Algorithm 1, which lead to the construction of the compressed context set.

Each element $c \in \mathcal{C}(X)$ of the context set is a collection of the machine-defined contexts $l \in \mathcal{L}(X)$. The corresponding contextual observation density and prior distribution need to be derived for the compressed context set. We will first assign values to $p(c \mid X, l)$ as follows:

$$p(c \mid X, l) = \begin{cases} 1, & \text{if } l \in c \\ 0, & \text{otherwise}. \end{cases} \quad (3.8)$$

This conditional density is well-defined as $\mathcal{C}(X)$ is a partition of $\mathcal{L}(X)$ and it assumes the value of 1 for only one $c \in \mathcal{C}(X)$. Now, we can compute the prior density using (3.8) as follows:

$$p(c \mid X) = \sum_{l \in \mathcal{L}(X)} p(c \mid X, l)p(l \mid X) = \sum_{l \in c} p(l \mid X), \quad (3.9)$$

where $p(l \mid X)$ is the state-dependent prior probability of the machine-defined context which is known. The observation density can be shown to be given accurately by the mixture model

$$p(Y \mid X, c) = \sum_{l_i \in c} \frac{p(l_i \mid X)}{p(c \mid X)} p(Y \mid X, l_i). \quad (3.10)$$

The overall model complexity stays the same as the number of mixture components still remain the same. In order to reduce the model complexity, we define the observation density $p(Y \mid X, c) = p(Y \mid X, l^*)$, where, $l^*$ is an element in $c$. Theorem 3.1 derives an approach to choose $l^*$ and provides a bound for the error induced by this process.
Theorem 3.1 (Bound for error introduced in compression by clique enumeration). If the distance function $d$ used in Algorithm 1 is symmetric Kullback-Leibler divergence ($sKL$), then for any fixed threshold $\varepsilon > 0$, the error induced by defining $p(Y \mid X, c) = p(Y \mid X, l^*)$ for some $l^* \in c$ is upper-bounded by the value $\varepsilon \left(1 - \frac{p(l^* \mid X)}{p(c \mid X)}\right)$, which is strictly less than $\varepsilon$. This error bound is minimized for $l^* = \arg \max_{l \in c} p(l \mid X)$.

Proof. At first, a known result from the literature will be shown and then, we will use it to prove the theorem. Let $p_0$ denote a mixture model with component densities $f_i^0$ and weights $\pi_i^0$ for $i \in \{1, \ldots, n_0\}$, and similarly let $p_1$ denote another mixture model with $n_1$ components. The convexity upper bound on KL-divergence [40] is given by

$$\text{KL}(p_0 \parallel p_1) \leq \sum_{i=1}^{n_0} \sum_{j=1}^{n_1} \pi_i^0 \pi_j^1 \text{KL}(f_i^0 \parallel f_j^1)$$

$$\implies d(p_0, p_1) \leq \sum_{i=1}^{n_0} \sum_{j=1}^{n_1} \pi_i^0 \pi_j^1 d(f_i^0, f_j^1)$$

(3.11)

since $d(p_0, p_1) = sKL(p_0, p_1) = \text{KL}(p_0 \parallel p_1) + \text{KL}(p_1 \parallel p_0)$. Let us consider $p_0$ to be the density from (3.10) with $\pi_i^0 = \frac{p(l_i \mid X)}{p(c \mid X)}$, $f_i^0 = p(Y \mid X, l_i)$, and $n_0 = |c|$, and assign $p_1 = p(Y \mid X, l^*)$. Using (3.11) and substituting for the mixture models, we get

$$d(p_0, p_1) \leq \sum_{l_i \in c \setminus \{l^*\}} \pi_i^0 d(p(Y \mid X, l_i), p(Y \mid X, l^*))$$

since $l^*$ belongs to $c$ and the distance function is positive definite. Since we know that $\varepsilon$ is the distance threshold and clique $c$ consists only of elements $l$, whose observations densities are at most $\varepsilon$ away from each other, we obtain $d(p_0, p_1) \leq \sum_{l_i \in c \setminus \{l^*\}} \pi_i^0 \varepsilon$. Substituting the values for $\pi_i^0$ and using (3.9), we obtain the desired
result as follows:

\[ d(p_0, p_1) \leq \varepsilon \sum_{l_i \in c \setminus \{l^*\}} \frac{p(l_i \mid X)}{p(c \mid X)} = \varepsilon \left(1 - \frac{p(l^* \mid X)}{p(c \mid X)}\right). \]  

(3.12)

Since \( l^* \) belongs to \( c \), we have \( p(l^* \mid X) \leq p(c \mid X) \). Thus, we verify that \( d(p_0, p_1) \) is indeed less than \( \varepsilon \) and the error bound is minimized for \( l^* = \arg \max_{l \in c} p(l \mid X) \). \( \square \)

This result shows us that, if we choose an acceptable level of error (\( \varepsilon \)), then we can use it as the graph threshold in the function \texttt{ConstructGraph} of Algorithm 1. The two limitations of this approach are that we do not know a priori the compression we will get, and that we will have to redo the computation if we decide to change the value of \( \varepsilon \). In order to alleviate both these issues, we will look at another approach ahead in this section.

### 3.4.2 Compression by Subset Selection

In the subset selection approach, we directly choose the size of the desired compressed context set (say \( k \)) and not the acceptable error. The proposed approach is to select a set of \( k \) distinct contexts from the machine-derived context set \( \mathcal{L}(X) \) and assign those to a compressed context set \( \mathcal{C}_k(X) \). Thus, we end up with \( \mathcal{C}_k(X) \subset \mathcal{L}(X) \) for \( k < |\mathcal{L}(X)| \). This section explains the approach to select the subset and derive a bound on the error introduced by subset selection.

Let us denote \( \bar{\mathcal{C}}_k(X) \) as the relative complement of \( \mathcal{C}_k(X) \) with respect to \( \mathcal{L}(X) \), given by \( \bar{\mathcal{C}}_k(X) = \mathcal{L}(X) \setminus \mathcal{C}_k(X) \). We consider that the set \( \mathcal{C}_k(X) \) is constructed by arbitrary selection of any \( k \) elements from the set \( \mathcal{L}(X) \). Theorem 3.2 will derive a bound for the error introduced by subset selection and then we provide a technique to choose the subset in a systematic way.

**Theorem 3.2** (Bound for error introduced in compression by subset selection). Let \( p_\mathcal{C}(Y \mid X) \) be the density estimated using the machine-derived context set \( \mathcal{L}(X) \) for the state \( X \), which is given as

\[ p_\mathcal{C}(Y \mid X) = \sum_{i \in \mathcal{L}(X)} \alpha_i K_X(Y, y_i), \]  

(3.13)
where $K_X(\cdot, \cdot)$ is a kernel function and $\alpha_i$ is the context prior probability associated with the context $i \in L(X)$. If $C_k(X)$ denotes a subset of machine-derived context set $L(X)$ consisting of $k$ elements, such that $\sum_{i \in C_k(X)} \alpha_i > 0$, then the density estimate obtained using this subset is given as

$$p_{C}(Y \mid X) = \sum_{i \in C_k(X)} \tilde{\alpha}_i K_X(Y, y_i), \quad (3.14)$$

where $\tilde{\alpha}_i = \frac{\alpha_i}{\sum_{i \in C_k(X)} \alpha_i}$ is the associated prior. The upper bound of the supremum norm of the error in density estimation due to subset selection is proportional to the sum of context priors from the set $\bar{C}_k(X)$, i.e., $L(X) \setminus C_k(X)$. In other words,

$$\|p_{C}(Y \mid X) - p_{\bar{C}}(Y \mid X)\|_\infty \leq \beta_X \sum_{i \in \bar{C}_k(X)} \alpha_i \quad (3.15)$$

where $\beta_X \in \mathbb{R}$ satisfies $0 \leq K_X(\cdot, \cdot) \leq \beta_X < \infty$.

Proof. Using (3.13) and (3.14), after some algebraic manipulations one can show that the difference in estimates for any $y \in Y$ is given as

$$p_{C}(y \mid X) - p_{\bar{C}}(y \mid X) = \frac{1}{\sum_{i \in C_k(X)} \alpha_i} \left( \sum_{i \in C_k(X)} \sum_{j \in \bar{C}_k(X)} \alpha_i \alpha_j (K_X(y, y_i) - K_X(y, y_j)) \right).$$

The supremum norm in this setting of continuous functions is given as

$$\|p_{C}(Y \mid X) - p_{\bar{C}}(Y \mid X)\|_\infty = \sup_{y \in Y} |p_{C}(y \mid X) - p_{\bar{C}}(y \mid X)|.$$

Using the absolute homogeneity and triangle inequality properties of the norm, we obtain

$$\|p_{C}(Y \mid X) - p_{\bar{C}}(Y \mid X)\|_\infty \leq \frac{1}{\sum_{i \in C_k(X)} \alpha_i} \left( \sum_{i \in C_k(X)} \alpha_i \alpha_j \sup_{y \in Y} |K_X(y, y_i) - K_X(y, y_j)| \right)$$

$$\leq \frac{1}{\sum_{i \in C_k(X)} \alpha_i} \left( \sum_{i \in C_k(X)} \alpha_i \alpha_j \beta_X \right).$$

Since $\beta_X$ is the maximum value assumed by the nonnegative-valued kernels, we
obtain the desired result,

\[ \| p_C(Y \mid X) - p_L(Y \mid X) \|_\infty \leq \beta_X \sum_{j \in \mathcal{C}_k(X)} \alpha_j. \]

**Remark 3.1** (Optimal k-subset). Since the error upper bound is directly proportional to \((1 - \sum_{i \in \mathcal{C}_k(X)} \alpha_i)\), the k-subset with minimum error bound is one for which \(\sum_{i \in \mathcal{C}_k(X)} \alpha_i\) is maximum. Thus, if the elements in the context set \(\mathcal{L}(X)\) are sorted in descending order of their priors \(p(l_i \mid X)\), i.e., \(\alpha_i\), then the best subset \(\mathcal{C}_k^*(X)\) corresponds to the first k elements of this sorted sequence. The corresponding error upper bound is \(\beta_X(1 - \sum_{j \in \mathcal{C}_k^*(X)} \alpha_j)\).

**Remark 3.2** (Optimal choice of k). Without loss of generality, we can assume that \(\alpha_1 \geq \alpha_2 \geq \cdots \geq \alpha_{|\mathcal{L}(X)|}\) represents the sorted sequence of context priors. If the error upper bound for a k-subset for any \(k \in \{1, 2, \ldots, |\mathcal{L}(X)|\}\) is denoted by \(e_k\), then \(e_k = \beta_X(1 - \sum_{j=1}^{k} \alpha_j)\) by Remark 3.1. We can verify that, the sequence \(\{e_k\}\) is monotonically decreasing with \(e_{|\mathcal{L}(X)|} = 0\). These values represent the accuracy of representation of the density. If in certain application we also have a model complexity function \(g(k)\), then we can trade-off accuracy with complexity using a criterion, such as Akaike Information Criterion [2], to find the optimal value of k that minimizes the chosen criterion.

The error bound derived in Theorem 3.2 is usually conservative, but this conservative analysis leads to a simple expression for \(e_k\), which can readily be evaluated for all \(k \in \{1, 2, \ldots, |\mathcal{L}(X)|\}\). Unlike the technique in Section 3.4.1, this approach can give a relationship of subset size or compression ratio with maximum error in estimation, which in turn can lead to choosing appropriate compression as shown in Remark 3.2. However this approach does not include contexts with low priors and does not use information of overlap/distance between individual components, which might not be desirable for certain applications.

This section presented two techniques for compression of context sets along with the main results of the upper bound of error due to approximation. The error bound evaluated in the first approach was in terms of statistical distance functions, whereas in the second case we derive a more intuitive bound in terms of the supremum norm. In the next section, we will use these techniques for
cardinality reduction of context sets derived from multiple seismic sensor data for a target classification problem.

### 3.4.3 Experiment and Results

The procedure and results of experimental validation of the context set compression techniques will be presented in this section. Field experiments were conducted to collect data from unattended ground sensors, such as seismic, acoustic, and passive infrared sensors, for the border-crossing target detection and classification problem. The details of the experiment setup is given in Appendix A. In this study, time-series data is used from two different seismic sensors that are separated by 7 meters, and the target is passing almost parallel to the line joining the two sensors at various distances between 2 to 8 meters. The hypothesis set consists of human walking ($x = 1$) and human running ($x = 2$) classes, and the goal is to classify the activity of the target using data from both the seismic sensors and quantify the effect of context set compression.

The dataset consists of 110 runs for walking and 118 runs for running. We partition the sample into a training set and testing set consisting of 60% and 40% data respectively. All results are generated for 10 different partitions of the sample and average results are presented ahead for the different steps. In the first step, low-dimensional features are extracted from time-series data using symbolic dynamic filtering (SDF) [91]. In SDF, we partition the measurement space into several regions and assign a symbol to each region. The set of these symbols is known as the alphabet. Bias is removed from the measurement time-series data to make it zero-mean and it is also normalized to have unit variance to remove the effect of target distance on signal amplitude. The resulting time-series data is then represented by a symbol sequence, and the statistics of evolution of this sequence is represented by a $D$-Markov model [72]. In this analysis, we used an alphabet size of 6 and a depth $D$ of 2, resulting in a $D$-Markov model of 7 states after state-splitting and state-merging. The left eigenvector of state transition matrix of the $D$-Markov model corresponding to the eigenvalue of 1 is the stationary state probability vector, which is used as a low-dimensional feature vector for each time-series data. The details on the $D$-Markov model construction and feature
Figure 3.4: Seismic sensor time series data for walking and running classes

(a) MCE for \(x = 1\)

(b) Subset selection for \(x = 1\)

(c) MCE for \(x = 2\)

(d) Subset selection for \(x = 2\)

Figure 3.5: Mean and range of cardinality of the compressed context set extraction techniques are given in [72].

The second step is of unsupervised context learning which uses nonparametric density estimation for obtaining machine-derived context sets from kernel-based mixture models as shown in Section 3.3. The density estimation process is used for computing the joint likelihood of obtaining a feature \(Y_1\) from seismic sensor 1 and a feature \(Y_2\) from seismic sensor 2, given that the state is \(X\). The kernels used
in the mixture modeling process are Gaussian with diagonal covariance matrix having identical entries, i.e., $K_i(y, y_i) = (2\pi\gamma)^{-d_i/2}\exp(-\frac{(y-y_i)^T(y-y_i)}{2\gamma^2})$, where $d_i$ is the dimensionality of feature $Y_i$ for $i = 1, 2$ and $\gamma$ is the kernel shape parameter. Using $\gamma = 0.01$, the resulting context sets for state 1 have cardinality of 14.80 on average with standard deviation of 1.47, whereas for state 2, we have cardinality of 20.60 on average with standard deviation of 1.65. In this analysis, we use a maximum likelihood decision rule for classification, thus the state estimate is given as

$$\hat{x} = \arg\max_{x \in \mathcal{X}} p(Y_1, Y_2 \mid x) = \arg\max_{x \in \mathcal{X}} \sum_{c \in C(x)} p(c \mid x) p(Y_1, Y_2 \mid x, c).$$

(3.16)

For the machine-derived context set with $\gamma = 0.01$, the classification accuracy was 99.78% on average using the decision rule in (3.16).

In the third step, we use the two proposed context set cardinality reduction techniques to obtain the compressed context sets. In the maximal clique enumeration (MCE) technique, the contextual observation densities are multivariate Gaussian distributions with mean $\mu_i(x)$ and identical covariance matrix $\Sigma_\gamma(x)$, which is parametrized by the kernel shape parameter $\gamma$, thus $p(Y \mid X = x, L = l_i) \sim \mathcal{N}(\mu_i(x), \Sigma_\gamma(x))$. In order to construct the weight matrix, we use the closed form expression of Bhattacharyya distance for Gaussian densities [1], given as

$$w_{ij}(x) = d(p(Y \mid X = x, L = l_i), p(Y \mid X = x, L = l_j)) = \frac{1}{8}(\mu_i(x) - \mu_j(x))^T\Sigma_\gamma(x)^{-1}(\mu_i(x) - \mu_j(x))$$

(3.17)
for $i,j = 1,2,\ldots,|\mathcal{L}(x)|$. The threshold parameter $\varepsilon$ to be used in the ConstructGraph function of the MCE approach is varied from $10^{-3}$ to $10^1$ in 15 equal steps in the log scale. For the graph obtained from the ConstructGraph function, we perform the maximal clique enumeration process and compute minterms of the obtained set of cliques. Note that as threshold increases, the cardinality of the compressed set shows a non-monotonic reducing trend in Figure 3.5a and 3.5c as the number of cliques need not reduce monotonically with reduction of the edge set of the graph. The Minterms procedure ensures by definition that the number of cliques in the resulting set is upper-bounded by the cardinality of the machine defined context set, that is, $|\mathcal{C}(x)| \leq |\mathcal{L}(x)|$ for all $x \in \mathcal{X}$, and thus we get some compression. The classification performance summary using compressed context sets using MCE is given in Figure 3.6a. The results show that for $\varepsilon = 10^{0.42} = 2.68$, the mean cardinality of context sets is $|\mathcal{C}(1)| = 7.8$ and $|\mathcal{C}(2)| = 13.9$, and the average classification accuracy is the same as the full context set. This result demonstrates that cardinality reduction need not significantly affect the class performance. However, reducing cardinality further by increasing $\varepsilon$ leads to significant deterioration in performance in this case. Cross-validation can be used to choose the appropriate value for $\varepsilon$.

In the subset selection approach, the maximum size of the subset, denoted by $k$, is varied from 2 to 16. If the original context set is smaller than the chosen set size, we do not perform any other computation, else we choose the best $k$-subset using Remark 3.1. Thus, Figure 3.5b and 3.5d show a monotonic trend of context set size with the chosen parameter $k$. The classification performance shows an increasing trend with the size of context set. For $k = 8$, the performance is 99.56% and for $k > 8$, the performance is as good as the original set. Thus, compression of context sets can be achieved by subset selection techniques as well. A suitable context set size can be chosen by using cross-validation if classification accuracy is the selection criterion, else one can use the method outlined in Remark 3.2 to choose a context set size.
3.5 Context-aware Sensor Fusion

In the context-aware sensor fusion approach, using the definition of context, the measurements are conditionally independent given the state-context pair. In other words, the following relation holds for multi-sensor teams with $M$ (possibly heterogeneous) measurements:

$$p(Y_1, \ldots, Y_M \mid X, C) = \prod_{i=1}^{M} p_i(Y_i \mid X, C).$$

If the state space $\mathcal{X}$ is finite, then using Bayes’ rule, the following sequential update rule for the posterior distribution of the state-context pair $(X, C)$ is used:

$$P(X, C \mid Y_1, \ldots, Y_{i-1}, Y_i) = \frac{p_i(Y_i \mid X, C)P(X, C \mid Y_1, \ldots, Y_{i-1})}{\sum_{x \in \mathcal{X}} \sum_{c \in C(x)} p_1(Y_1, c)P(x, c \mid Y_1, \ldots, Y_{i-1})} \quad (3.18a)$$

for $i = 2, 3, \ldots, M$, where

$$P(X, C \mid Y_1) = \frac{p_1(Y_1 \mid X, C)\pi_C(X)p(X)}{\sum_{x \in \mathcal{X}} \sum_{c \in C(x)} p_1(Y_1, c)\pi_c(x)p(x)}. \quad (3.18b)$$

This update rule plays a crucial role in sequential inference and decision-making problems. In a sequential state estimation problem, for instance, one keeps track of the posterior probability of the state-context pair $P(X, C \mid Y_1, \ldots, Y_i)$, updates it to $P(X, C \mid Y_1, \ldots, Y_i, Y_{i+1})$ as a new sensor measurement $Y_{i+1}$ becomes available, and marginalizes out the context variable to obtain the posterior probability of the state $P(X \mid Y_1, \ldots, Y_i, Y_{i+1})$, from which an updated state estimate can be deduced. The posterior distribution over the state also captures the uncertainty in the state estimate and this uncertainty can be leveraged in decision-making tasks, such as sensor selection (as we will see in Chapter 6). Moreover, the question posed in Chapter 1 (How to tractably fuse information from multi-modal sensors without assuming conditional independence given state?) can now be answered. The solution is context-aware sensor fusion, which allows to tractably and accurately fuse information from multi-modal sensors without using the naïve Bayesian assumption. This framework will be validated using data from field experiment on border control.
3.5.1 Experiment and Results

The data used in this study is exactly the same as shown in Section 3.4.3, which was collected by field experiments at the Penn State Border Control testbed (See Appendix A). The hypothesis set consists of human walking ($x = 1$) and human running ($x = 2$) classes, and the goal is to classify the activity of the target using data from both the seismic sensors with and without modeling context.

Two examples of time-series data are shown in Figure 3.4 from each seismic sensor. The dataset consists of time-series of various length, consisting of 10000 to 29000 measurements. Using the symbolic dynamic filtering approach [72], stationary probability vectors are extracted as features from each time-series data. The alphabet-size for the time-series data was chosen to be 3, the depth of the Markov model was chosen to be 2, and the state size was 7, i.e., $|\Sigma| = 3, D = 2, |Q| = 7$. These 7 dimensional features are probability vectors, thus the last element of the vector is removed as it is dependent on the other entries. The likelihood models for each sensor is learned using the density estimation technique in Chapter 2 using P5 to obtain $p(Y_1|X)$ and $p(Y_2|X)$ for 5 different values of the kernel parameter. For each of this case, the context learning approach developed in this chapter is used to obtain the context-aware measurement model $p(Y_1|X, c)$ and $p(Y_2|X, c)$ for each $c$ in $C(X)$. The classification results from individual sensors as well as fusion assuming naïve Bayes’ condition ($Y_1 \perp_X Y_2$) holds is compared with that from context-aware sensor fusion. In this study, bootstrap cross-validation (60% training, 40% testing, 5 repetitions) is used to obtain the average classification performance with a maximum likelihood classifier, i.e., $\hat{x} = \arg \max_{x \in \{1, 2\}} p(Y | X = x)$. The results are given in Table 3.1. It is observed that context-aware sensor fusion consistently outperforms other approaches with significant margin and the best performance improves the accuracy by over 6.5% by using context awareness.

Note that in order to learn context from data, the features from both the sensors must be available synchronously, and then a joint density estimation problem is solved; whereas, under naïve Bayes’ assumption the product of marginal observation densities of the measurements gives us the joint density. Thus, the improvement in performance is due to the context variable, which captures the dependencies between measurements obtained for the same state.
Table 3.1: Classification performance for context-aware fusion

<table>
<thead>
<tr>
<th>$\gamma$</th>
<th>$Y_1$</th>
<th>$Y_2$</th>
<th>$Y_1 \perp_X Y_2$</th>
<th>$Y_1 \perp_{X,C} Y_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0100</td>
<td>81.76%</td>
<td>89.45%</td>
<td>91.65%</td>
<td>98.24%</td>
</tr>
<tr>
<td>0.0178</td>
<td>81.98%</td>
<td>85.05%</td>
<td>86.59%</td>
<td>97.36%</td>
</tr>
<tr>
<td>0.0316</td>
<td>83.52%</td>
<td>81.76%</td>
<td>83.96%</td>
<td>97.14%</td>
</tr>
<tr>
<td>0.0562</td>
<td>78.24%</td>
<td>81.76%</td>
<td>84.62%</td>
<td>97.36%</td>
</tr>
<tr>
<td>0.1000</td>
<td>76.48%</td>
<td>83.30%</td>
<td>85.27%</td>
<td>95.60%</td>
</tr>
<tr>
<td>MAX</td>
<td>83.52%</td>
<td>89.45%</td>
<td>91.65%</td>
<td>98.24%</td>
</tr>
</tbody>
</table>

3.6 Conclusion

In this chapter, the notion of context is mathematically defined to enable machines to extract context from multi-modal data. A data-driven method based on non-parametric density estimation (from Chapter 2) is developed using a special type of product kernels to determine the context set and learn a context-aware mixture model of sensor measurements. This method enables the machine to define an abstract context set from the available data, so that the measurements are conditionally independent given the state and context. The existing approaches for modeling contexts either need extensive domain expertise for each of the sensing modality in the system, or rely on clustering-based techniques that do not guarantee conditional independence of measurements, which is crucial for tractable and accurate information fusion in sequential decision-making. Thus, a context-aware sensor fusion method is developed for tractable multi-modal sensor fusion leveraging the conditional independence enabled due to context. The cardinality of the context set obtained from the proposed method need not be adequately sparse for implementation on a resource-constrained platform, hence a context set compression technique is developed using graph theory and subset selection.

These techniques have been validated with experimental data from seismic sensors to classify targets. It was inferred that compression reduces computation but does not proportionately affect performance, so these techniques can successfully be used in practical system. It was also demonstrated that knowledge of context im-
proves the classification accuracy (from 91.65\% to 98.24\%) for target classification using time-series data from two seismic sensors. The developed context-aware measurement models are applicable to several other detection and estimation problems. These models would directly be used in the framework of context-aware sensor selection for sequential hypothesis testing in Chapter 6. Future research directions include context-aware measurement modeling for systems with continuous states and switched systems with discrete modes. The development of context-aware models to address information fusion with medium-sized sensor networks having over 20-40 different sensors would also be considered, where the dependency structure between sensors is also inferred via context.
Learning Markov Models for Time-series Data

4.1 Introduction

Intelligent physical systems, under the paradigm of dynamic data-driven application systems (DDDAS) [20], need to control and make critical decisions about the underlying dynamics or statistical behaviors, based on the spatio-temporal information generated by a multitude of sensors. The space in which all the sensor measurements lie is known as the information space. Data-driven techniques, which construct generative probabilistic models by considering the temporal evolution of measurements in the information space, can be used for control and on-line monitoring of complex dynamical behaviors in DDDAS.

In the study of dynamical systems exhibiting chaos, in particular, symbolic time-series analysis (STSA) has previously been used to understand long-term statistical behavior in order to characterize the chaotic phenomenon [9]. Symbolic analysis has also been applied to characterizing the dynamics of physical systems with experimental data [22] as well as to predictive modeling of electro-mechanical systems [91, 97]. Symbolic evolution modeling primarily consists of the following three steps:

- **Alphabet-Size Selection**: The number of symbols needed to achieve the desired level of information-space discretization is selected.
• **Partitioning and Symbolization:** The information space is divided into mutually exclusive and exhaustive cells, and each data point in a measurement trajectory is assigned the symbol of the cell in which it lies.

• **Model Construction:** The resulting symbol sequence is then used to create a model that describes the spatio-temporal evolution of the sequence.

Several researchers have addressed issues related to the above three steps to make the theory of symbolic analysis applicable to problems of control and decision-making. For example, Sarkar et al. [96] reported an alphabet-size selection technique for improving the performance of pattern classification. Mukherjee and Ray [72] improved the procedure of constructing a statistical model from symbol sequences beyond what was originally reported by Ray [91].

Alphabet-size selection tools assume that a space partitioning technique has been decided, whereas partitioning techniques need the alphabet size to be known. However, these two steps are currently solved in a sequential manner, and thus the choice of partitioning technique to use in the analysis may not be decided in a systematic way. Hence, there is a need to unify alphabet-size selection and space partitioning steps into one technique. This crucial issue will be addressed in this chapter by formulating a linear program, whose solution simultaneously yields both the alphabet size and the information-space partition.

While several techniques have been used in the literature for one-dimensional information-space partitioning, including uniform, maximum entropy [91], and symbolic false nearest neighbor partitioning [49], the challenge is its extension to vector-valued time series that are commonly encountered in many human-engineered systems. It appears that either there is no rigorous way to extend these one-dimensional techniques to higher dimensions or they become computationally expensive/intractable for higher dimensions. For example, maximum entropy partitioning, which ensures (nearly) the same number of data points in each region, gives a unique set of intervals in the one-dimensional space; however, for higher dimensions there are infinitely many ways to partition the space with each case having the same entropy score. This chapter proposes a kernel-based density-estimation technique to address the multi-dimensional space partitioning problem. The proposed approach is scalable as the computational complexity of
the optimization is independent of the dimensionality of the information space.

The chapter also presents two different symbolization rules—namely, the maximum likelihood (ML) rule and the maximum a posteriori (MAP) rule—for assigning symbols to each measurement and thus indirectly generating the partition regions from the density estimate. Using such a symbolization rule, a measurement sequence can be symbolized to generate a symbol string. These symbol strings are coarse-grained representations of the measurement sequences, which can be used for clustering, classification, and anomaly detection, as demonstrated in [58].

In this chapter, temporal evolution models for measurement sequences are developed from symbol strings by using the theory presented in [72] to generate statistical models, which can be used for pattern classification. Specifically, in this chapter, temporal evolution models are built from simulated phase trajectories of a forced nonlinear oscillator, and the models are then used for parameter estimation. In summary, the main contributions of the chapter are as follows:

- Development of a density-estimation scheme for simultaneously solving the problems of alphabet-size selection and information-space partitioning;
- Validation of the proposed method with simulation and experiments.

The remaining chapter is organized into three sections. Section 4.2 describes how the density estimate is used for determining the alphabet size and partitioning the information space, and how the obtained symbol sequence is used for constructing statistical models. In Section 4.3, these models are obtained from a simulation testbed for pattern classification. Lastly, in Section 4.4, some concluding remarks are made.

### 4.2 Partitioning, Symbolization, and Modeling

This section describes the density estimation-based partitioning and symbolization method, which is then used for time-series modeling using STSA.

#### 4.2.1 Density Estimation

Density estimation from data can be addressed with parametric and nonparametric techniques. The Gaussian mixture model (GMM) method is a common example of
parametric approaches for density estimation, which needs to know or compute the number of components in the mixture model separately. Alternatively, one may use kernel-based density estimation [122], which is a nonparametric approach to modeling densities without having any prior knowledge of the data. Our objective in density estimation is to obtain a mixture model of the form

\[ f(x) = \sum_{s \in \Sigma} p(s)p(x \mid s), \quad (4.1) \]

where \( \Sigma \) is the alphabet set, \( s \) is a symbol from the set, \( p(s) \) is the probability of symbol \( s \), and \( p(x \mid s) \) is the likelihood of a measurement \( x \) sampled from the information space \( \mathcal{X} \subset \mathbb{R}^n \) given a symbol \( s \). It is desirable to have the alphabet size (i.e., the cardinality of \( \Sigma \)) to be much smaller than the sample size \( N \) of the training set; i.e., \( |\Sigma| \ll N \). A nonparametric technique to fit such a mixture model, where the number of components in the mixture is automatically determined, is proposed in this section.

In nonparametric regression techniques, such as support vector regression and regularization networks [108, 56], the estimate of the unknown function \( f \) is of the form

\[ f(x) = \sum_{i=1}^{N} \pi_i K(x, x_i), \quad (4.2) \]

where \( K(\cdot, \cdot) \) is the kernel function. The kernel functions are required to satisfy \( \int_\mathcal{X} K(z, x)dz = 1 \) for any \( x \in \mathcal{X} \). Two examples of kernel functions \( K(x, \tilde{x}) \) that satisfy this requirement are given by

\[ \text{Laplace:} \quad \frac{1}{(2\gamma)^n} \exp \left( -\frac{\sum_{d=1}^{n} |x^d - \tilde{x}^d|}{\gamma} \right), \quad (4.3a) \]

\[ \text{Gaussian:} \quad \frac{1}{(2\pi\gamma)^{n/2}} \exp \left( -\frac{\sum_{d=1}^{n} (x^d - \tilde{x}^d)^2}{2\gamma} \right) \quad (4.3b) \]

for all \( x = (x^1, \ldots, x^n) \) and \( \tilde{x} = (\tilde{x}^1, \ldots, \tilde{x}^n) \), where \( \gamma > 0 \) is the kernel shape parameter. In the problem of probability density estimation, as discussed in Chapter 2, the function \( f \) can be estimated by solving a linear optimization problem given in P5. The solution will give us a value of vector \( \pi \), which is expected to
be sparse. Let $\Sigma$ be the alphabet set, which consists of the set of indices $i$ such that $\pi_i \neq 0$. Then, the sparsity of $\pi$ will imply the desired scalability condition $|\Sigma| \ll N$. Now, setting the symbol probabilities $p(s) = \pi_s$ and component densities $p(\cdot | s) = K(\cdot, x_s)$ for $s \in \Sigma$ yields the desired mixture model (4.1).

4.2.2 Partitioning and Symbolization

A partition of the information space is determined by a symbolization rule $\mu: X \to \Sigma$, which assigns a unique symbol to each measurement sampled from $X$. A maximum likelihood (ML) rule $\mu^\text{ML}(\cdot)$ and a maximum a posteriori (MAP) rule $\mu^\text{MAP}(\cdot)$ has been considered for symbolization. These rules are defined by

$$
\mu^\text{ML}(x) = \arg \max_{s \in \Sigma} p(x | s),
$$

$$
\mu^\text{MAP}(x) = \arg \max_{s \in \Sigma} p(s)p(x | s)
$$

for $x \in X$. These rules lead to generalizations of Voronoi diagrams [5], which divide a space into regions that are closer to one of the given seed points using some distance function. In our approach, the seed points $x_s$, $s \in \Sigma$, are chosen by the optimization problem, and the distance function for generating partition regions depends upon the choice of the kernel used in the density estimation technique. With the ML rule, for example, the symmetric Laplace kernel in (4.3a) (resp. Gaussian kernel in (4.3b)) leads to a Voronoi partition of $X$ with the $\ell_1$ norm (resp. $\ell_2$ norm) defining the distance function. If the distance function can also have a separate bias term representing the weight of each seed point, it leads to an additively-weighted Voronoi diagram, which can be generated with the MAP rule. Examples of a few of these partitions are shown in Section 4.3. Note that the kernel-based mixture models have identical component densities, hence they lead to Voronoi partitions, which would not have been possible by other nonparametric approaches, such as orthogonal series expansions [103].

4.2.3 Markov Models of Time-series Data

The modeling method is built upon the concept of symbolic dynamic filtering [91] or symbolic time-series analysis (See Appendix B for more details), where the
symbol strings are used to generate a special class of statistical models called $D$-Markov machines [72]. These models have a deterministic algebraic structure and their states are represented by symbol words of length $D$ or less, which are identified from the symbol sequence. A $D$-Markov machine [72] is constructed in two steps; namely, (i) state splitting, where the states are split based on their information contents; and (ii) state merging, where two or more states (of possibly different lengths) are merged together to form a new state without any significant loss of the embedded information. The $D$-Markov machine is completely represented by the tuple $G = (\Sigma, Q, \delta, M)$, where $\Sigma$ is the alphabet set, $Q$ is the state set, $\delta : Q \times \Sigma \rightarrow Q$ is the state transition map, and $M : Q \times \Sigma \rightarrow [0, 1]$ is the morph map. The quantity $M(q, s)$ represents the probability of occurrence of symbol $s \in \Sigma$ when the current state is $q \in Q$. The matrix formed by these quantities can be used as a feature to represent the underlying time series for various classification and machine learning tasks. State splitting and merging steps [72] typically ensure that $|Q| \ll |\Sigma|^D$.

### 4.3 Results and Discussion

A forced nonlinear Duffing oscillator model is used for validation of the proposed multi-dimensional information-space partitioning technique. The simulation aims to demonstrate that the data-driven models, which are obtained from the generated symbol sequences, can be used on-line for anomaly/change detection [37] as well as for identifying slowly-changing health parameters of various systems.

**Data Generation**

The simulated model is given as follows:

$$\ddot{x} + b\dot{x} + k_1 x + k_2 x^3 = A \cos(\omega t).$$

The system parameters used in simulation were $A = 22$, $\omega = 1$, $k_1 = 0$, and $k_2 = 1$. Ten different values were chosen for the parameter $b$ in the range $[1, 3]$. A few sample phase trajectories are shown in Fig. 4.1; they demonstrate the small changes in the trajectories as $b$ varies. The system was simulated for 800 seconds for each value of $b$ and the obtained trajectory is divided into 30 equal runs.
Although periodic, each run will have a different initial state, and thus in total 300 runs were available in the training dataset. Note that, in order to demonstrate the utility of the proposed approach for data-driven modeling, the differential equation model is only used for simulation of phase trajectories. This two state system also enables easy visualization of the phase-space partitions. The framework has been applied to periodic trajectories in this work, however it can also be applied when the trajectories are chaotic.

**Partitioning, Model Construction, and Testing**

The training dataset has 30 measurement sequences for each of the 10 simulated values of parameter $b$. One representative time series from each of the ten cases was used for density estimation and partitioning. The kernels used for estimating the density are shown in (4.3), where the kernel shape parameter $\gamma$ controls the resolution of the partitioning process. A high value of $\gamma$ would result in coarse-grained partition and a small alphabet size, which are well-suited for the study of high-level system behavior, whereas a small value of $\gamma$ would give a fine partition, which is suitable for a detailed study of the underlying dynamics—see Table 4.1. Thus, the choice of $\gamma$ is dependent on the task at hand and it can be optimized with cross-validation. In this experiment, four different values of $\gamma$ were used for each type of kernel to see the effect of the shape parameter on density estimation.
Table 4.1: Density estimation, partitioning, and classification performance with the proposed technique and symbolization rules

| Kernel Type | $\gamma$ | $\sigma$ | $|\Sigma|$ | Classification Accuracy |
|-------------|----------|----------|------------|-------------------------|
|             |          | ML       | MAP        | ML | MAP | GMM |
| Laplace     | 0.5      | 0.017    | 31         | 23 | 1.000 | 0.990 | 0.993 |
|             | 1.0      | 0.036    | 11         | 7  | 0.970 | 0.910 | 0.890 |
|             | 1.5      | 0.056    | 7          | 3  | 0.960 | 0.927 | 0.793 |
|             | 2.0      | 0.072    | 4          | 2  | 0.843 | 0.417 | 0.373 |
| Gaussian    | 0.5      | 0.026    | 19         | 15 | 0.993 | 0.927 | 0.927 |
|             | 1.0      | 0.037    | 14         | 7  | 0.993 | 0.903 | 0.923 |
|             | 1.5      | 0.045    | 9          | 5  | 0.923 | 0.810 | 0.823 |
|             | 2.0      | 0.052    | 7          | 3  | 0.910 | 0.563 | 0.817 |

and partitioning. Since we fixed the parameter $\nu$ at 0.05, in order to maintain the same fraction of errors, the optimal value of the error margin $\sigma$ increased as the kernel parameter $\gamma$ was increased—see Table 4.1. A few of the resulting partitions from the analysis are shown in Fig. 4.2.

The density estimation process automatically chooses more seed points in dense areas. Since the ML rule ignores the weights associated with these points, it leads to a finer partition in denser regions of the information space. As verified by Figs. 4.2a and 4.2e, this aspect of ML rule–based partitioning is expected to enable the evolution models to capture small changes in system behavior. On the other hand, as the MAP rule considers the weight associated with each symbol, it enables automatic merging of the nearby partition regions of different symbols, thus denser regions of the information space lie in a single partition region, as seen in Figs. 4.2g and 4.2h. This allows the model to capture the high-level system behavior with small alphabet sizes. Hence, Table 4.1 shows that the effective alphabet size $|\Sigma|$ with the MAP rule is always smaller than that with the ML rule.

In support of the sparsity claim for the proposed technique, Fig. 4.3 shows the values of the top 40 coefficients in descending order of their magnitude. Since
Figure 4.2: Phase-space partitions obtained by the proposed technique. The red crosses indicate the seed point locations.
the length of the solution vector $\pi$ is 1004 and only less than 30 components of the vector have significant values in all cases, the sparsity claim is verified for the current application scenario.

In this study, the $D$-Markov PFSA models constructed from the symbol sequences have $D = 1$, thus the state set $Q = \Sigma$. The feature matrix from the morph map $M$ is reshaped into a column vector, and then principal component analysis (PCA) is used for dimensionality reduction to extract a 6-dimensional feature vector. The average classification performance for 10 classes with 5-fold cross-validation using a support vector machine classifier is presented in Table 4.1. The results show that classification accuracy for features extracted using the ML technique is better than that from the MAP technique. This was expected as the classification task was to distinguish nearby trajectories and the ML technique yields finer partitions. The type of kernel used in the analysis did not significantly affect the classification results.

The classification result was also compared with GMM-based partitioning, which requires the number of component densities be fixed at the outset. In this study, we tried to fit a GMM with the number of components ranging from 2 to
30, but a minimum value for Akaike information criterion (AIC) was not attained inside this interval. Nonetheless, we used the number of components obtained from our proposed technique to train the GMM with diagonal covariance matrices. The resulting model performed comparably to the one with the MAP technique. This shows that the proposed technique is better than GMM in the sense that, while GMM alone is not suitable for alphabet-size selection, the proposed density estimation technique solves a single optimization problem for alphabet-size selection and information-space partitioning.

4.4 Conclusion

In this work, the nonparametric density estimation was used to together solve the alphabet-size selection and information-space partitioning problem. The linear program for density estimation gives sparse solution, and hence the alphabet set is much smaller than the number of training points. The density estimation framework for partitioning gives flexibility to the user to choose the resolution and shape of the partition. It was shown that the kernel shape parameter gives control over the resolution of the partition, and that, in order to study detailed behavior of temporal data, smaller values for kernel bandwidth is required. Moreover, the choice of kernel allows to select a distance metric for creating Voronoi partitions, since Laplace kernel gives Voronoi diagrams with $\ell_1$ metric and Gaussian kernel gives that with $\ell_2$ metric. Simulation result for parameter classification showed that the proposed approach can do better than GMM-based partitioning, since it can choose the alphabet size as well as give better parameter classification. Thus, the problem of information modeling from multidimensional time-series data, which was posed in Chapter 1, can now be answered with density estimation for alphabet-size selection, partitioning, and symbolization, followed by STSA for model construction. This modeling approach is not dependent on the sensing modality and is applicable to any sensor irrespective of its dimensionality.
Chapter 5

Sequential Hypothesis Testing with Models of Time-series Data

5.1 Introduction

Recently there has been an unprecedented increase in the volume and speed of temporal data being generated by physical systems, due to the improvements in low-cost sensing and high speed communication. Typical machine learning approaches [12] used for monitoring of physical processes extract features from a fixed number of consecutive observations and pose a supervised learning problem for detection or classification using those features. Similarly, fixed-sample-size (FSS) tests from estimation theory literature [84] compute likelihood or posterior probability of an observed sequence of fixed length and select the class which maximizes this statistic. However, in order to perform well on all feasible sequences, the chosen sample length in these approaches has to be longer than needed for most of the easily separable cases. In time-critical online monitoring systems, such as combustion instability detection for aircraft gas-turbine engines, an early detection can help to minimize the structural damage to the engine by avoiding the thermo-acoustic resonance and even prevent fatal accidents due to engine shutdown. A dynamic data-driven approach [21] for sequential detection and classification is needed which can adapt the sequence length based on the observations without any extra computational burden. Sequential hypothesis tests offer one such efficient
and adaptive framework, which can allow online detection of faults, anomalies, and mode transitions in systems where multiple streams of temporal data are generated. In this chapter, a sequential hypothesis test is presented for a class of Markov models of temporal data inferred using symbolic time-series analysis.

A sequential detector is a statistical decision function which uses a random number of samples depending on the observation sequence to detect the underlying hypothesis. A sequential detector, on average, would need much smaller sequence length than FSS tests [84]. Sequential probability ratio test (SPRT) [129, 107] has been traditionally used for binary hypothesis testing and is known to be an optimal detector when the observation sequences are independent and identically distributed (IID) [14]. However, sequential data from physical systems are typically not independent as causal relations, due to physics, lead to statistical dependencies. Hidden Markov models (HMM) are temporal evolution models that are learned from data using iterative techniques to capture some of those dependencies [12]. In [35, 16, 31], the SPRT was extended for data generated from a HMM. In contrast, symbolic time-series analysis-based Markov modeling makes use of concepts from symbolic dynamics to represent a dynamical system as a discrete Markov model. This technique consists of two critical steps - discretization (or partitioning) of phase-space of the dynamical system [49, 126] and memory-size estimation [112]. The dynamical system is then approximated as a $D$-Markov model, which is a Markov chain, with finite memory (or order), over the discrete state-space of the system [91, 72]. In this chapter, a technique is presented to sequentially estimate the log-likelihood ratio for a binary hypothesis test, where each hypothesis is represented by a $D$-Markov model.

This chapter extends the results of classical SPRT for IID observations to SPRT for observations from $D$-Markov models of time-series data. The probability density of the Markov model given the observed sequence is represented as a product of categorical distributions and the parameters of this distribution come from a Dirichlet distribution. Using the Dirichlet distribution, as the conjugate prior of the categorical distribution, a sequential update rule is developed for likelihood ratio of Markov models to perform hypothesis test for binary classification problems. Expected increment of the log-likelihood ratios are explicitly provided under each hypothesis and it is shown that the sequential tests for the Markov
models will terminate in finite time with probability one. The efficacy of the proposed approach for online monitoring with streaming data is demonstrated with simulations as well as experimental data from a lean-premixed swirl-stabilized combustion process. The performance of the sequential test is compared with a FSS test (maximum likelihood classifier) to demonstrate that the proposed approach can make accurate decisions using fewer observations.

The chapter is organized as follows: Section 5.2 presents the problem formulation and explains the difference with respect to the SPRT for IID sequences. In Section 5.3, proposes an approach to address the sequential hypothesis testing problem and provides the theoretical justification for the technique. Section 5.4 presents the results and inferences from simulation studies. Section 5.5 shows the results from the combustion instability detection experiment along with a description of the test apparatus. Section 5.6 concludes the chapter along with recommendations for future research.

5.2 Problem Formulation

In literature, sequential detector for real-valued independent and identically distributed (IID) observations to infer which of the known probability densities is generating the data has been well-studied [84]. In this section, the sequential detection problem for discrete-valued IID observations is shown and then extended to the case in which the observation sequence is generated by a $D$-Markov model.

5.2.1 Sequential Detection: IID Case

The discrete-valued observations $\{Y_t; t = 1, 2, \ldots\}$ are independent and identically distributed according to

\[ H_0 : Y_t \sim P_0, \ t = 1, 2, \ldots \]

versus

\[ H_1 : Y_t \sim P_1, \ t = 1, 2, \ldots, \]
where $P_0$ and $P_1$ are two known probability distribution (or measures) on the measurable space $(\Sigma, 2^\Sigma)$. Here, $\Sigma$ is a finite set of symbols, each symbol $s \in \Sigma$ is a discrete observation, and $2^\Sigma$ denotes the power set of $\Sigma$. Recall that power set of a set consists all possible subsets of the set including the empty set.

A sequential decision rule is a pair of sequences $(\bar{\phi}, \bar{\mu})$, where $\bar{\phi} \triangleq \{\phi_j; j = 0, 1, 2, \ldots\}$ is called a stopping rule with $\phi_j : \Sigma^j \rightarrow \{0, 1\}$ and $\bar{\mu} \triangleq \{\mu_j; j = 0, 1, 2, \ldots\}$ is called a terminal decision rule with $\mu_j : \Sigma^j \rightarrow \mathcal{X}$, where $\mathcal{X}$ is the hypothesis set, which is $\{0, 1\}$ for binary hypothesis testing. The sequential detector works as follows: For an observed symbol sequence of finitely many symbols $S_t = \{s_i \in \Sigma; i = 1, 2, \ldots, t\}$, the stopping rule decides if one must stop sampling (i.e., $\phi_t(S_t) = 1$) and make a decision or continue sampling (i.e., $\phi_t(S_t) = 0$) to observe the next symbol $s_{t+1}$. If the decision is to stop sampling, then, the terminal decision rule selects the hypothesis that one must declare for the observed symbol sequence, i.e., if $\phi_t(S_t) = 1$, then the final decision is $x \in \mathcal{X}$, where $x = \mu_t(S_t)$.

### 5.2.2 Sequential Detection: D-Markov Case

In the hypothesis testing problem under consideration, the observations may not be IID as they are observations from a physical process. Therefore, a D-Markov model is given corresponding to each hypothesis, instead of a probability distribution. These D-Markov models are represented as $\mathcal{M}_k = (Q, \Sigma, \delta, M_k)$ for each hypothesis $k \in \mathcal{X}$, where $Q$ represents the set of states, $\Sigma$ is the alphabet set, $\delta : Q \times \Sigma \rightarrow Q$ is the transition function, and $M_k$ is the class-specific morph matrix, which consists of entries $[M_k]_{ij} = m_{ij}^k = p_k(s_j|q_i)$. Each row of matrix $M_k$ gives the probability of seeing a symbol $s_j \in \Sigma$ from a particular state $q_i \in Q$ in model $k \in \mathcal{X}$. Thus, each row of the morph matrix is a categorical distribution [79]. See Appendix B for more details on D-Markov models.

Note that all of these models have a similar algebraic structure and they differ only in the symbol emission probabilities. It is assumed that the training instances for nominal and anomalous conditions have been obtained as sufficiently long symbol sequences, which ensure the convergence of the morph matrix parameters in the training phase [67]. However, in the test phase, we need to make accurate decisions using a smaller number of observations from the streaming data.
Consider a Markov model over the state-symbol pair, which generates the symbol sequence, to be given as:

$$p(q_{t+1}, s_{t+1} \mid q_t, s_t) = p(s_{t+1} \mid q_t, s_t) p(q_{t+1} \mid q_t, s_t)$$

$$= p(s_{t+1} \mid q_{t+1}) p(q_{t+1} \mid q_t, s_t)$$

(5.1)

where $p(q_{t+1} \mid q_t, s_t)$ is obtained from the transition function $\delta$ as $p(q_{t+1} \mid q_t, s_t) = 1$, if $\delta(q_t, s_t) = q_{t+1}$, and $p(q_{t+1} \mid q_t, s_t) = 0$, otherwise. In (5.1), $p(s_{t+1} \mid q_{t+1})$ is obtained from the morph matrix $M$. Since, the state transition is deterministic (see Definition B.1 in Appendix B), the emission of the symbol sequence from hypothesis $k$ is governed only by the morph matrix $M_k = \{p_k(s_{t+1} = j \mid q_{t+1} = i)\}$. Thus, the hypothesis test is given as follows: The discrete-valued observation sequence $S_t = \{s_i \in \Sigma : i = 1, 2, \ldots, t\}$ is generated according to

$$H_0 : S_t \sim M_0, \ t = 1, 2, \ldots$$

versus

$$H_1 : S_t \sim M_1, \ t = 1, 2, \ldots,$$

where $M_k$ is the morph matrix of the PFSA model $M_k$ for $k = 0, 1$. It is noted that, in a $D$-Markov model each state is represented by a finite history of $D$ symbols, thus observing the first $D$ symbols, the initial state will be known. Hence, the sequential detector is initiated after $D$ symbols have been observed. The sequential detector in this case is also defined as the pair of sequences of stopping rule and terminal decision rule as discussed in Section 5.2.1.

### 5.2.3 Problem Statement

In this work, a sequential detector is constructed to identify the $D$-Markov model, which is generating a particular symbol sequence, as shown in the hypothesis test in Section 5.2.2. Given the models, the objectives of this work are:

1. To construct a sequential detector to guarantee a desired probability of detection $P_D(\tilde{\phi}, \tilde{\mu})$ as well as the probability of false alarm $P_{FA}(\tilde{\phi}, \tilde{\mu})$;

2. To prove that the sequential detector terminates with a decision in finite
time with probability one and to obtain an estimate of stopping time \( N = \min\{t : \phi_t(S_t) = 1\}; \\

3. To elucidate the developed sequential detector with simulation models; and \\

4. To validate the sequential test with experimental data for detection of combustion instabilities.

5.3 Proposed Approach

In this section, first the likelihood of a symbol sequence to be emitted from the PFSA model \( \mathcal{M}_k \) is computed. The sequential decision rule is then developed and some analysis is shown to obtain estimates of the performance bounds.

5.3.1 Likelihood for D-Markov Models

The \( D \)-Markov model for each hypothesis is represented by a morph matrix. Each row of the morph matrix of a \( D \)-Markov model is a discrete probability mass function denoting probability of emission of a symbol from a given state. Since the occurrence of states are statistically independent events, the rows represent a set of independent categorical distribution [79]. The Dirichlet distribution is known to be the conjugate prior of the categorical distribution [12]. In other words, if the prior density over parameters of the categorical distribution is represented as a Dirichlet distribution, then posterior density after a sequential update is also given by a Dirichlet distribution. Thus, for a given state, the density over parameters of a particular row of morph matrix is given by Dirichlet distribution. Let \( S_t \) for \( t \in \mathbb{N} \) denote the symbol sequence \((s_1, s_2, \ldots, s_t)\) of length \( t \). The likelihood of a given model \( \mathcal{M}_k \) conditioned on the symbol sequence \( S_t \) is computed as:

\[
P(\mathcal{M}_k \mid S_t) = \prod_{i=1}^{Q} \left( \frac{\prod_{j=1}^{\Sigma} (m_{ij})^{\alpha_{ij}^t - 1}}{B(\bar{\alpha}_i^t)} \right)
\]

where the hyper-parameter \( \alpha_{ij}^t \) is initialized at \( \alpha_{ij}^0 \), the value of \( \alpha_{ij}^t - \alpha_{ij}^0 \) is the count of occurrence of symbol \( s_j \) after state \( q_i \) in a sequence \( S_t \), \( \bar{\alpha}_i^t \) denotes the
vector \([\alpha_{t1}, \alpha_{t2}, \ldots, \alpha_{t|\Sigma|}]\) for all \(i \in \{1, 2, \ldots, |Q|\}\). \(B(\bar{\alpha}_t) = \prod_{|\Sigma|}^{\sum} \Gamma(\alpha_{tij})\), and \(\Gamma(\cdot)\) is the Gamma function.

In binary hypothesis testing, such as Bayes’ or Neyman-Pearson hypothesis testing, the log-likelihood ratio is computed and compared with a predetermined threshold \([84]\). In the proposed approach, the log-likelihood ratio is computed using (5.2) as follows:

\[
\Lambda_t = \log \left( \frac{P(M_1 | S_t)}{P(M_0 | S_t)} \right) = \log \left( \prod_{i=1}^{\sum} \prod_{j=1}^{\sum} \left( \frac{m_{1ij}}{m_{0ij}} \right)^{\alpha_{tij} - 1} \right) 
= \sum_{i=1}^{\sum} \sum_{j=1}^{\sum} (\alpha_{tij} - 1) \log \left( \frac{m_{1ij}}{m_{0ij}} \right), 
\]

\[
\Lambda_t = \sum_{i=1}^{\sum} \sum_{j=1}^{\sum} w_{ij}(\alpha_{tij} - 1), 
\]

where the constant \(w_{ij}\) denotes \(\log \left( \frac{m_{1ij}}{m_{0ij}} \right)\). A simple relation was obtained to compute the log-likelihood ratio in (5.3) and the sequential update rule for the log-likelihood ratio is developed ahead.

Proposition 5.1 (Sequential Update). Given that the log-likelihood ratio at time \(t\) is \(\Lambda_t\) and the state at time \(t+1\) is \(q_{t+1}\), if \(s_{t+1}\) is the emitted symbol, then the updated log-likelihood ratio \(\Lambda_{t+1}\) is given by:

\[
\Lambda_{t+1} = \Lambda_t + w_{q_{t+1}s_{t+1}}, 
\]

where \(w_{q_{t+1}s_{t+1}} = \{w_{ij} : q_{t+1} = i \text{ and } s_{t+1} = j\}\).

Proof. Recall that the hyper-parameter \(\alpha_{t+1}^{ij} - \alpha_{ij}^0\) is the count of occurrence of symbol \(s_j\) after the state \(q_i\) in the observed symbol sequence \(S_{i+1}\), then we get:

\[
\alpha_{t+1}^{ij} = \alpha_{ij}^t + \mathbb{1}_{\{ij\}}(q_{t+1})\mathbb{1}_{\{j\}}(s_{t+1}), 
\]

where \(\mathbb{1}_A(\cdot)\) is the indicator function with set \(A\), that is, \(\mathbb{1}_A(x) = 1\), if \(x\) belongs
to \( A \); otherwise, \( \mathbb{1}_A(x) = 0 \). Then, by using the log-likelihood ratio given in (5.3) for time \( t + 1 \), and (5.5), we obtain:

\[
\Lambda_{t+1} = \sum_{i=1}^{Q} \sum_{j=1}^{\Sigma} w_{ij}(\alpha_{ij}^t - 1) + w_{ij}\mathbb{1}_{\{i\}}(q_{t+1})\mathbb{1}_{\{j\}}(s_{t+1}) = \Lambda_t + w_{q_{t+1}s_{t+1}}. \tag{5.6c}
\]

The simplicity of sequential update of the log-likelihood ratio for \( D \)-Markov models enables online update of the statistic \( \Lambda_t(S_t) \). This factor would aid in real-time monitoring of physical systems. In the next part, we will use the computed log-likelihood ratio to obtain a sequential decision rule.

### 5.3.2 SPRT for D-Markov Models

As mentioned in Section 5.2.2, a sequential decision rule is a pair of sequences \((\bar{\phi}, \bar{\mu})\), where \( \bar{\phi} \triangleq \{\phi_j; j = 0, 1, 2, \ldots\} \) is called a stopping rule and \( \bar{\mu} \triangleq \{\mu_j; j = 0, 1, 2, \ldots\} \) is called a terminal decision rule. Let us choose two thresholds \( \gamma_0 \) and \( \gamma_1 \) with \( \gamma_0 < 0 < \gamma_1 \), then the sequential probability ratio test SPRT(\( \gamma_0, \gamma_1 \)) is defined by the sequence of stopping rule \( \phi_j \) and terminal decision rule \( \mu_j \) as follows:

\[
\phi_j(S_j) = \begin{cases} 
0; & \text{if } \gamma_0 < \Lambda_j(S_j) < \gamma_1, \\
1; & \text{otherwise.}
\end{cases} \tag{5.6a}
\]

\[
\mu_j(S_j) = \begin{cases} 
0; & \text{if } \Lambda_j(S_j) \leq \gamma_0, \\
1; & \text{if } \Lambda_j(S_j) \geq \gamma_1.
\end{cases} \tag{5.6b}
\]

Let \( p_d = P_D(\bar{\phi}, \bar{\mu}) \) be the probability of detection and \( p_{fa} = P_{FA}(\bar{\phi}, \bar{\mu}) \) be the probability of false alarm of the detector. In order to choose the thresholds, the following relations from Wald’s SPRT [84] are being used:

\[
\gamma_1 \leq \log \left( \frac{p_d}{p_{fa}} \right) \quad \text{and} \quad \gamma_0 \geq \log \left( \frac{1 - p_d}{1 - p_{fa}} \right). \tag{5.7}
\]
These relations hold even without the assumption of independence [35]. These inequalities not only relate the performance with the chosen thresholds but also help to choose the threshold to guarantee a particular desired performance. The specific proof to show that these relations also hold for our case is shown in Section 5.3.4. The thresholds are conservatively chosen to be $\gamma_1 \triangleq \log \left( \frac{p_d}{p_{fa}} \right)$ and $\gamma_0 \triangleq \log \left( \frac{1-p_d}{1-p_{fa}} \right)$. It is noted that, the thresholds and their respective performance guarantees are independent of the underlying models.

5.3.3 Analysis of the Sequential Detector

The sequential detector $\text{SPRT}(\gamma_0, \gamma_1)$ has been shown in (5.6). In this part, the stochastic evolution and asymptotic behavior of the log-likelihood ratio (LLR) statistic for $D$-Markov models will be explored as well as the proof that the sequential detector will terminate in finite time with probability one will be given.

Before discussing the analysis, recall Kullback-Leibler divergence (or K-L divergence) from information theory. The K-L divergence between two probability mass functions [17] is defined as:

$$d_{\text{KL}}(p(q) \| \bar{p}(q)) = \sum_{q \in Q} p(q) \log \left( \frac{p(q)}{\bar{p}(q)} \right).$$

In this work, the difference between two conditional distributions is obtained in terms of the conditional relative entropy [17] as:

$$d(p(s|q) \| \bar{p}(s|q)) = \sum_{q \in Q} p(q) \sum_{s \in \Sigma} p(s|q) \log \left( \frac{p(s|q)}{\bar{p}(s|q)} \right). \quad (5.8)$$

Conditional relative entropy is used as a measure of distance between two morph matrices. The stochastic evolution of the log-likelihood ratio is explored ahead.

**Expectation of log-likelihood ratio**

The sequential update rule for LLR in (5.4) suggests that its stochastic evolution has some resemblance with a random walk on the real line [79]. This stochastic evolution is characterized in the following Proposition.

**Proposition 5.2** (Random Walk on a DFSA). Given that DFSA $\mathcal{G}$ is the common
algebraic structure in both models $\mathcal{M}_0$ and $\mathcal{M}_1$, and $S_t$ is a (finite-length) symbol sequence over the alphabet $\Sigma$, then the stochastic evolution of the log-likelihood ratio $\Lambda_t = \log\left(\frac{P(M_1 | S_t)}{P(M_0 | S_t)}\right)$, is given by a weighted random walk on the DFSA $G$.

Proof. Using (5.4), we can obtain:

$$
\Lambda_{t+2} = \Lambda_{t+1} + w_{q_{t+2}s_{t+2}},
= \Lambda_t + w_{q_{t+1}s_{t+1}} + w_{q_{t+2}s_{t+2}}.
$$

Given $q_{t+1}$ and $s_{t+1}$, the next state $q_{t+2} = \delta(q_{t+1}, s_{t+1})$, thus, $q_{t+2}$ is not random and is governed by the algebraic structure of the DFSA. The symbol emission probabilities are obtained from the morph matrix of true model, given the current state. The step size $w_{q_is_i}$ at time step $t$ can take $|Q||\Sigma|$ different values depending on the state and emitted symbol pair. Thus, the evolution of LLR is a weighted random walk over a DFSA. \(\Box\)

Given the stochastic evolution of the LLR statistic from the Proposition 5.2, we now show the expected value of the LLR statistic in the following theorem.

**Theorem 5.1** (Expected value of log-likelihood ratio). Given that $p_k(q_1)$ is a row vector of initial state probabilities, $T_k$ is the state transition probability matrix and $M_k$ is the morph matrix for model $\mathcal{M}_k$, and $W = [w_{ij}] = [\log(m^1_{ij})]$, then the expectation (over all possible symbol sequences) of the log-likelihood ratio statistic after $t$ updates is given as:

$$
E[\Lambda_t | \mathcal{M}_k] = \sum_{l=1}^{t} p_k(q_1)T_{k,l-1}^{l}(M_k \odot W)1_{|\Sigma|},
$$

(5.9)

where $\odot$ is element-wise multiplication operator and $1_n$ is column vector of $n$ ones.

Proof. Using $\Lambda_{t+1} = \Lambda_t + w_{q_{t+1}s_{t+1}}$ from (5.4), we get:

$$
E[\Lambda_{t+1} - \Lambda_t | \mathcal{M}_k] = E[w_{q_{t+1}s_{t+1}} | \mathcal{M}_k]
= E\left[\sum_{j=1}^{Q} \sum_{i=1}^{\Sigma} w_{ij}1_{\{i\}}(q_{t+1})1_{\{j\}}(s_{t+1}) | \mathcal{M}_k\right]
$$
Using $p_k(q_{t+1}) = p_k(q_1)T_k^t$, we obtain:

$$E[\Lambda_{t+1} - \Lambda_t | \mathcal{M}_k] = p_k(q_1)T_k^t(M_k \odot W)1_{|\Sigma|}. \quad (5.10)$$

Thus, we derived the expression for the expected increment in log-likelihood ratio after $t$ updates. Finally, adding up the expected increments, we get the desired result for the expected log-likelihood ratio value as:

$$E[\Lambda_t | \mathcal{M}_k] = \sum_{l=1}^t p_k(q_1)T_k^{t-1}(M_k \odot W)1_{|\Sigma|}. \quad \square$$

The implication of Theorem 5.1 is explained by the following set of remarks.

**Remark 5.1 (Information-theoretic interpretation).** Using (5.10) and the relation for conditional relative entropy (5.8), we get:

$$E[\Lambda_{t+1} - \Lambda_t | \mathcal{M}_1] = p_1(q_{t+1})(M_1 \odot W)1_{|\Sigma|}$$

$$= \sum_{j=1}^{|Q|} \sum_{i=1}^{|\Sigma|} p_1(q_{t+1} = j)m_{ij}^1 \log\left(\frac{m_{ij}^1}{m_{ij}^0}\right)$$

$$= \mathbb{E}_{p_1(s_{t+1}, q_{t+1})}\left[ \log\left(\frac{p_1(s_{t+1} | q_{t+1})}{p_0(s_{t+1} | q_{t+1})}\right) \right]$$

$$= d(p_1(s_{t+1} | q_{t+1}) || p_0(s_{t+1} | q_{t+1}))$$

$$= d(M_1 || M_0).$$

Similarly, using non-negativity of conditional relative entropy, we get:

$$E[\Lambda_{t+1} - \Lambda_t | \mathcal{M}_0] = -d(M_0 || M_1).$$
Thus, we can infer that expected increment in log-likelihood ratio at time $t+1$ is the conditional relative entropy between the given models at time $t+1$, where time dependence is introduced because of state probability vector $p_t(q_{t+1})$. Thus, considering conditional relative entropy to be a distance function, the expected increment is directly related to the statistical distance between the two different models. Intuitively, this result implies that models which are closer will need a larger number of observations to terminate the SPRT for a prespecified desired performance.

**Remark 5.2** (Asymptotic behavior of log-likelihood ratio). Using the result from Theorem 5.1 and computing the limit as number of observations tend to infinity, we get:

$$\lim_{t \to \infty} \frac{E[\Delta_t | \mathcal{M}_k]}{t} = \tilde{\pi}_k (M_k \odot W)_{1|\Sigma}$$

(5.11)

where $\tilde{\pi}_k = \lim_{t \to \infty} (\frac{1}{t} \sum_{l=1}^{t} p_k(q_l) T_{k}^{l-1})$ is the Cesaro limit. The constant value on the right hand side of this result implies that the growth of log-likelihood ratio becomes linear asymptotically.

**Remark 5.3** (Case of stationary Markov model). Consider that both the Markov models $\mathcal{M}_0$ and $\mathcal{M}_1$ are stationary and the corresponding stationary distribution is given as $\pi_0$ and $\pi_1$, respectively. Thus, the state probability vector is same as the stationary distribution of the Markov model at all time epochs, that is, $p_k(q_{t+1}) = \pi_k$ for all $t \in \mathbb{N}$ and $k = 0, 1$. Using $\pi_k$ in (5.10), we get:

$$E[\Delta_{t+1} - \Delta_t | \mathcal{M}_k] = \pi_k (M_k \odot W)_{1|\Sigma},$$

(5.12)

which is a constant value. Hence, the expectation under model $\mathcal{M}_k$ is given by:

$$E[\Delta_t | \mathcal{M}_k] = t \pi_k (M_k \odot W)_{1|\Sigma}.$$

(5.13)

Since, it is known that the Cesaro limit is equal to the stationary distribution for irreducible aperiodic Markov models [67], this result matches the intuition from Remark 5.2 with $\tilde{\pi}_k = \pi_k$. Using the interpretation from Remark 5.1, we get:

$$E[\Delta_t | \mathcal{M}_k] = (-1)^{1-k} t d(M_k \parallel M_{1-k}),$$

(5.14)

for $k = 0, 1$. This relation for the stationary case shows direct relationship between
the expected value of log-likelihood ratio and the distance between the models.

5.3.4 Relation between Threshold and Performance

In the sequential detector for hypothesis testing with D-Markov models SPRT($\gamma_0, \gamma_1$) is given by (5.6) and the probability of false alarm and probability of detection is given by $p_{fa}$ and $p_d$ respectively. Then, it is claimed that the following relation holds between the thresholds and the performance:

$$\gamma_1 \leq \log \left( \frac{p_d}{p_{fa}} \right) \quad \text{and} \quad \gamma_0 \geq \log \left( \frac{1 - p_d}{1 - p_{fa}} \right).$$

Proof. Let the set of symbol strings $S_T$ with $T < \infty$, which terminate by satisfying $\Lambda_T(S_T) \geq \gamma_1$, be denoted by $\Gamma_1$, where $\Lambda_T(S_T)$ is the log-likelihood ratio for the sequence $S_T$. Then, we can write:

$$\Gamma_1 = \{ S \in \Sigma^* : \Lambda_T(S_T) \geq \gamma_1 \} = \bigcup_{t=1}^{\infty} \Theta_1^t,$$

where $\Theta_1^t = \{ S_t \in \Sigma^t : \Lambda_t(S_t) \geq \gamma_1 \}$. In the sequential detector, we declare the estimated hypothesis to be $\mathcal{H}_1$, when $\Lambda_t(S_t) \geq \gamma_1$, thus, we can obtain probability of false alarm ($p_{fa}$) and probability of detection ($p_d$) as:

$$p_{fa} = p(\Lambda_T(S_T) \geq \gamma_1 | \mathcal{H}_0) = \sum_{t=1}^{\infty} \sum_{S_t \in \Theta_1^t} p(S_t | \mathcal{H}_0),$$

$$p_d = p(\Lambda_T(S_T) \geq \gamma_1 | \mathcal{H}_1) = \sum_{t=1}^{\infty} \sum_{S_t \in \Theta_1^t} p(S_t | \mathcal{H}_1).$$

We know for $S_t \in \Theta_1^t$, $\log\left( \frac{p(S_t | \mathcal{H}_1)}{p(S_t | \mathcal{H}_0)} \right) \geq \gamma_1$, so we get $e^{-\gamma_1} p(S_t | \mathcal{H}_1) \geq p(S_t | \mathcal{H}_0)$ for all $t$. Thus,

$$p_{fa} \leq e^{-\gamma_1} \sum_{t=1}^{\infty} \sum_{S_t \in \Theta_1^t} p(S_t | \mathcal{H}_1) = e^{-\gamma_1} p_d,$$

implying that, $\gamma_1 \leq \log\left( \frac{p_d}{p_{fa}} \right)$. Similarly to prove the other inequality, let

$$\Gamma_0 = \{ S \in \Sigma^* : \Lambda_T(S_T) \leq \gamma_0 \} = \bigcup_{t=1}^{\infty} \Theta_0^t,$$
where $\Theta^0_t = \{S_t \in \Sigma^t : \Lambda_t(S_t) \leq \gamma_0\}$. Then, probability of misdetection can be written as:

$$1 - p_d = \sum_{t=1}^{\infty} \sum_{S_t \in \Theta^0_t} p(S_t | \mathcal{H}_1).$$

Since $e^{\gamma_0} p(S_t | \mathcal{H}_0) \geq p(S_t | \mathcal{H}_1)$ for all $S_t \in \Theta^0_t$ for all $t$, we get:

$$1 - p_d \leq \sum_{t=1}^{\infty} \sum_{S_t \in \Theta^0_t} e^{\gamma_0} p(S_t | \mathcal{H}_0),$$

$$1 - p_d \leq e^{\gamma_0} (1 - p_{fa}),$$

which implies that $\gamma_0 \geq \log \left( \frac{1 - p_{fa}}{1 - p_d} \right)$.

### 5.3.5 Completion of Test and Estimated Sequence Length

In this subsection, we will prove that the sequential probability ratio test for $D$-Markov models will terminate in finite time with probability one and also provide an estimate of the average observation sequence length. The notion of stopping time under each hypothesis for a symbol sequence $S_t$ is given as follows:

$$\tau_0(S_t) = \inf \{t \in \mathbb{N} : \Lambda_t(S_t) \leq \gamma_0\}, \quad (5.15a)$$

$$\tau_1(S_t) = \inf \{t \in \mathbb{N} : \Lambda_t(S_t) \geq \gamma_1\}. \quad (5.15b)$$

Then, the stopping time for the sequential test is given by $\tau(S_t) = \min\{\tau_0(S_t), \tau_1(S_t)\}$. The formal result of the guarantee of completion of the sequential test in finite time is given ahead in Theorem 5.2.

**Theorem 5.2.** (Completion of test) If the discrete finite-order Markov process generating the symbols is ergodic, then the sequential test with decision rule shown in Section 5.3.2 terminates in finite time almost surely under both hypothesis, i.e., $P(\{\tau(S_t) < +\infty\}|\mathcal{M}_k) = 1$, for $k = 0, 1$.

**Proof.** Ergodicity of the symbol sequence generated by a Markov model implies that the sequence of log-likelihood ratio (LLR) is also ergodic for the model, since the LLR is a deterministic function of the stochastic sequence. Thus, the sequence
of time averages of the LLR converges to the ensemble average (or expected value) of the LLR almost surely, that is,

$$\lim_{t \to \infty} \frac{E[\Lambda_t | M_k]}{t} = \lim_{t \to \infty} \frac{1}{t} \Lambda_t(S_t) \text{ a.s.}$$  \hspace{1cm} (5.16)

under hypothesis $k \in \mathcal{X}$. Using (5.11) for $k = 1$, we get:

$$\lim_{t \to \infty} \frac{1}{t} \Lambda_t(S_t) = \tilde{\pi}_1(M_1 \circ W)1_{|\Sigma|} \text{ a.s.}$$  \hspace{1cm} (5.17)

This result implies that $P(\{\lim_{t \to \infty} \Lambda_t(S_t) > \gamma_1\}|M_1) = 1$. Similarly, for $k = 0$ we can show $P(\{\lim_{t \to \infty} \Lambda_t(S_t) < \gamma_0\}|M_0) = 1$. Using (5.15), this means that $P(\{\tau_k(S_t) < +\infty\}|M_k) = 1$, for $k = 0, 1$. The result then follows by using that $\tau(S_t) = \min\{\tau_0(S_t), \tau_1(S_t)\}$.

From Theorem 5.2, we know that the sequential tests for $D$-Markov machines will terminate in finite time with probability one. We will now provide an estimate of this stopping time with respect to the expected behavior of the LLR sequence. For each hypothesis, this sequence length estimate is defined using $E[\Lambda_t | M_k]$ from (5.9) as follows:

$$\tilde{\tau}_1 \triangleq \inf_{t \in \mathbb{N}} \{t : \sum_{l=1}^{t} p_1(q_l)T_1^{l-1}(M_1 \circ W)1_{|\Sigma|} \geq \gamma_1\},$$  \hspace{1cm} (5.18a)

$$\tilde{\tau}_0 \triangleq \inf_{t \in \mathbb{N}} \{t : \sum_{l=1}^{t} p_0(q_l)T_0^{l-1}(M_0 \circ W)1_{|\Sigma|} \leq \gamma_0\}.$$  \hspace{1cm} (5.18b)

The utility of these estimates and other results from this section are elucidated by simulation and validated with experimental data in the next two sections.

### 5.4 Results from Simulation

A theoretical framework for sequential hypothesis testing with $D$-Markov models was developed in the previous sections. This section presents the results of numerical simulation to elucidate the underlying principles of the proposed sequential hypothesis testing procedure.
Table 5.1: D-Markov models used in simulation

| Case | $M_0$ | $M_1$ | $d(M_0||M_1)$ | $d(M_1||M_0)$ |
|------|-------|-------|---------------|---------------|
| 1    | 0.6 0.4 | 0.6 0.4 | 0.037         | 0.075         |
|      | 0.7 0.3 | 0.7 0.3 |               |               |
|      | 0.6 0.4 | 0.6 0.4 |               |               |
|      | 0.7 0.3 | 0.3 0.7  |               |               |
| 2    | 0.1 0.9 | 0.9 0.1 | 0.561         | 0.989         |
|      | 0.7 0.3 | 0.3 0.7  |               |               |
|      | 0.6 0.4 | 0.4 0.6  |               |               |
|      | 0.7 0.3 | 0.3 0.7  |               |               |

5.4.1 Description of the Simulation Setup

An anomaly detection scenario is simulated by two D-Markov models, having a common binary alphabet set (i.e., $\Sigma = \{0,1\}$) and the same depth $D = 2$. These models have been trained with sufficiently long training data. The first model, $M_0$, represents the nominal behavior and the second model, $M_1$, represents the anomalous behavior. The state set $Q$ for each model is given by the words $\{00, 01, 10, 11\}$. We perform simulation for two different cases: Case 1, where models $M_0$ and $M_1$ are largely similar; and Case 2, where models $M_0$ and $M_1$ are reasonably different. The morph matrix for the models from both the cases is shown in Table 5.1. The distance between the models, in terms of conditional relative entropy, using the stationary distribution for the state probabilities, is also given in Table 5.1 and we can verify that Case 1 models are closer than the models in Case 2.

We have simulated 2500 symbol sequences of length 1000 from each model and used the developed sequential detector from Section 5.3.2 to make decisions. The results from this analysis are discussed ahead.

5.4.2 Simulation Results

Simulated symbol sequences are used to sequentially estimate the log-likelihood ratio (LLR) statistic. In order to visualize the behavior of the LLR statistic under different hypothesis, we show 500 LLR trajectories under each hypothesis for both cases in Figure 5.1. The expected value of the LLR is computed using the derived
equation (5.9) and it is also shown in Figure 5.1. We can qualitatively verify that the LLR trajectories under the two hypothesis begin to differ as we use more observations in both cases. Moreover, if the models are further apart (as in Case 2), then the distinction between the LLR trajectories is discernible using fewer observations. The models in Case 1 have identical emission probabilities for 3 out of 4 states (see Table 5.1), therefore the discriminatory information is only available when the model is in state 4. Hence, in Figure 5.1a, we can observe that the LLR value remains constant for several consecutive observations.

In Remark 5.2, we had derived the asymptotic value of the average increment of expected value LLR under each hypothesis. In Figure 5.2, we qualitatively verify the convergence of the time average of increment in expected LLR to the limit computed in Remark 5.2 under both hypotheses for Case 2.

We have given an estimate of the sequence length in (5.18). We show that it is a reasonable estimate of the average sample length in Figure 5.3 by comparing the estimated length with the average sample length by repeating the simulation for 40 different choices of probability of detection between 0.8 and 0.999, while keeping probability of false alarm \( p_{fa} \) fixed at 0.001. This figure also verifies the intuition that we will need more observations to get better performance.

In Figure 5.4, the histograms of stopping length under \( H_0 \) being true and \( H_1 \) being true for Case 1 of our simulation models are shown. From Remark 5.1, we know that magnitude of expected increment in LLR, under a given hypothesis, say \( H_k \), is equal to the distance of the true model from the alternate model, i.e., \( d(M_k \| M_{1-k}) \). Thus, if \( d(M_1 \| M_0) > d(M_0 \| M_1) \) (see Table 5.1 for Case 1), then on average we expect the sequential test to terminate with fewer observations under hypothesis \( H_1 \). This can be verified in Figure 5.4, where the distribution for \( H_1 \) is to the left of \( H_0 \). It is noted that the empirical distribution is not symmetric, thus, the characterization of the stopping time for the test requires further investigation by estimating the higher moments of the respective distributions [35].

The efficacy of a sequential detector is characterized by the average sample length and the detection performance. The desired performance in terms of probability of detection and false alarm is used to design the sequential detector, as shown in Section 5.3.2. In Figure 5.5, we can verify that the probability of detection of the sequential detector is better than the desired \( p_d \) and the probability of
false alarm was also lesser than 0.001. Thus, the claim that sequential detector with $D$-Markov models achieves the desired performance has been verified.

The performance of the proposed sequential approach is compared with the

![Log-likelihood ratio trajectories](image)

(a) *Case 1*: Models $M_0$ and $M_1$ being largely similar

(b) *Case 2*: Models $M_0$ and $M_1$ being different

Figure 5.1: Log-likelihood ratio trajectories, when $H_0$ is true (shown in red) and when $H_1$ is true (shown in blue), for (a) *Case 1* and (b) *Case 2*. The expected value of log-likelihood ratio in each case is shown for $H_0$ (with −· line) and $H_1$ (with − − line)
Figure 5.2: Validation of convergence of average increment of expected log-likelihood ratio to the computed limit under each hypothesis.

(a) Under $\mathcal{H}_0$

(b) Under $\mathcal{H}_1$

Figure 5.3: Average sample length and estimated length for different values of $p_d$ and constant $p_{fa} = 0.001$

maximum likelihood classifier, which is a fixed-sample-size (FSS) test. The classification rule for a symbol sequence $S_N$ of length $N$ is given in terms of the likelihood as follows: $x^*(S_N) = \arg \max_{x \in \mathcal{X}} p(S_N | \mathcal{M}_x)$. The comparison of performance of
Figure 5.4: Histograms of sample length under each hypothesis with $p_d = 0.999$ and constant $p_{fa} = 0.001$

Figure 5.5: Observed probability of detection for different values of $p_d$ and constant $p_{fa} = 0.001$

the classifier with the developed approach for different probability of detection ($p_d$) is shown in Table 5.2. We can infer that for the same $p_d$, on average the FSS test needs several more observations as compared to the proposed sequential detector. Note that, the desired $p_{fa}$ was chosen to be 0.001 for the SPRT and the observed
Table 5.2: Comparison of the performance ($p_{fa}$) and average sample length (ASL) of the proposed SPRT and maximum likelihood (ML) classifier (fixed length ($N$)) for same $p_d$.

<table>
<thead>
<tr>
<th>$p_d$</th>
<th>SPRT</th>
<th>ML</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$p_{fa}$</td>
<td>ASL</td>
</tr>
<tr>
<td>0.960</td>
<td>0.0004</td>
<td>101.5</td>
</tr>
<tr>
<td>0.980</td>
<td>0.0004</td>
<td>114.4</td>
</tr>
<tr>
<td>0.995</td>
<td>0.0004</td>
<td>138.4</td>
</tr>
<tr>
<td>0.999</td>
<td>0.0004</td>
<td>162.0</td>
</tr>
</tbody>
</table>

$p_{fa}$ is 0.0004, which is lower than all cases of ML classifier. Thus, we obtain better performance with lesser number of observations (on average) with the proposed sequential detector.

In summary, the inferences from the simulation experiments are as follows:

- The LLR trajectories differ under different hypothesis as more observations are used and the distance between the models govern the expected LLR increment.
- The average increment of expected LLR from the simulated models converges to the limit derived in (5.11).
- The estimate of the sequence length provided in (5.18) is a reasonable estimate of the average sample length.
- The models that are further apart need fewer observations for termination of the sequential detector.
- The thresholds in the sequential detector can be chosen based on any desired performance and that performance is guaranteed by the detector at the expense of sampling more observations.
- The sequential detector gives better performance with fewer observations (on average) as compared to fixed-sample-size tests in this study.
5.5 Results from Combustion Experiments

In this section, the analysis for and the results from detection of unstable behavior in lean pre-mixed combustion have been described. Combustion instability problem has been explained in Appendix A. In this validation, a total of 780 cases for pressure data have been used (each sequence is 65,536 long); however, the label for stable/unstable case is not available. The labels are created by clustering the Markov models of pressure data and using some domain expertise. These details are being skipped here. Finally, 125 stable and 125 unstable cases are used for analysis presented in this work.

The time-series data set is first normalized by subtracting the mean and then dividing by the standard deviation of its elements; this step corresponds to bias removal and variance normalization. Data from engineering systems are typically over-sampled to ensure that the underlying dynamics can be captured. Due to coarse-graining in the symbolization process, an over-sampled time-series may mask the true nature of the system dynamics in the symbolic domain (e.g., occurrence of self loops and irrelevant spurious transitions in the Markov chain). A time-series is first down-sampled to find the next crucial observation. The first minimum of auto-correlation function generated from the observed time-series is obtained to find the uncorrelated samples in time. The data sets are then down-sampled by this lag (it is noted that different sequence will have different lags). To avoid discarding significant amount of data due to down-sampling, the down-sampled data using different initial conditions are concatenated. Further details of this pre-processing are reported in [112, 44].

The measurement space of the continuous time-series is then partitioned using maximum entropy partitioning (MEP) [86], where the information rich regions of the measurement space are partitioned finer and those with sparse information are partitioned coarser. In essence, each cell in the partitioned set contains (approximately) equal number of data points under MEP. A ternary alphabet with \( \Sigma = \{ 0, 1, 2 \} \) has been used to symbolize the data. The datasets from two different modes of combustion, stable and unstable, have been analyzed with the aim of detecting the unstable modes.

To perform the sequential ratio tests using the combustion data, a single-depth
Figure 5.6: Average sample length and estimated length for different values of $p_d$ and constant $p_{fa} = 0.01$

Figure 5.7: Observed probability of detection for different values of $p_d$ and constant $p_{fa} = 0.01$

Markov model for stable and unstable cases on the discretized sequence is trained. A single symbol sequence of length $\sim 65,000$ is used to train Markov models for the two hypotheses – stable and unstable. This model is then used to detect the unstable modes from a test data set with 250 discrete sequences with equal pro-
Table 5.3: Comparison of the performance ($p_d$) and average sample length (ASL) of the proposed SPRT and the maximum likelihood (ML) classifier (fixed length $N$) for the same $p_{fa}$

<table>
<thead>
<tr>
<th>$p_{fa}$</th>
<th>SPRT</th>
<th>ML</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$p_d$</td>
<td>ASL</td>
</tr>
<tr>
<td>0.016</td>
<td>0.952</td>
<td>60.64</td>
</tr>
<tr>
<td>0.032</td>
<td>0.952</td>
<td>52.55</td>
</tr>
<tr>
<td>0.080</td>
<td>0.960</td>
<td>41.88</td>
</tr>
<tr>
<td>0.120</td>
<td>0.968</td>
<td>36.56</td>
</tr>
</tbody>
</table>

portions of stable and unstable cases. Results of sequential tests are summarized in Figure 5.6 and 5.7. Figure 5.6 compares the theoretical estimate of stopping time for the sequential tests with the average sample length obtained over the test data set corresponding to different operating conditions specified by the desired probability of detection ($p_d$) and a constant false alarm rate, $p_{fa} = 0.01$. Figure 5.7 shows the probability of detection achieved by the sequential tests for different desired detection rates. Observed probability of false alarm is 0.016 for all test cases, which is slightly higher than the desired value of $p_{fa} = 0.01$. Out of 125 stable and 125 unstable cases, the best performance has 5 misdetections and 2 false alarms. It is noted that the sequential tests for Markov models are able to achieve desired performance for most cases (except for cases when desired $p_d > 0.95$).

The result from sequential approach using experimental data is also compared with a fixed-sample-size (FSS) maximum likelihood classifier. The desired $p_d$ was chosen to be 0.95 for the SPRT, the observed $p_d$ is greater than 0.95 and it is also higher than all cases of ML classifier (See Table 5.3). Thus, the result shown in Table 5.3 experimentally verifies that sequential tests need fewer observations than FSS tests on average to give the same (or even better) level of performance.

In contrast to the inferences from the simulation examples in Section 5.4, we can observe that the sequential tests could not always guarantee desired performance. This observation can be attributed to the fact that the test set contains data from a wide range of operating conditions. Thus, the test samples cannot be represented accurately by a single $D$-Markov model used for training.
5.6 Conclusion

Symbolic analysis-based Markov modeling provides very simple inference and learning technique which can be used for on-line monitoring of critical physical systems. Sequential hypothesis tests are useful for fast and in-situ detection of anomalies and events in systems where measurements are available in a streaming fashion. In this work, results for sequential hypothesis tests for symbolic analysis-based Markov models of time-series data were presented. First, a likelihood model for $D$-Markov models was presented using Dirichlet and categorical distributions; it was then used to estimate the log-likelihood ratio (LLR) for models representing the hypothesis to be tested. The LLR was then used to develop a sequential probability ratio test (SPRT) for the $D$-Markov model. The stochastic evolution of the LLR statistic was studied and it was proven that the sequential probability ratio test will terminate in finite time with probability one. The underlying concept was illustrated using two cases of $D$-Markov models with a binary symbol alphabet and depth $D = 2$. Finally, the proposed sequential test was used to detect occurrence of unstable combustion on experimental data of pressure time-series collected from a laboratory apparatus of swirl-stabilized combustion. The sequential tests are able to achieve 95% detection accuracy at 1.6% false alarm for detecting instability with small data-length over a wide range of operating conditions. The performance of the proposed sequential test for $D$-Markov models was compared with fixed-sample-size test (maximum likelihood). We verified that sequential tests need fewer observations than FSS tests on average to give the same level of performance, which is important for real-time monitoring of physical processes.

Although, the Dirichlet Multinomial framework was leveraged to obtain the likelihood of the model, the likelihood ratio could have been obtained even without using the Dirichlet framework. In this initial study, the posterior density given the symbol sequence was not used to address problems of model uncertainty, measurement noise, and/or process variability, which will be explored in future. The sequential probability ratio test for detection of simple hypothesis with IID observations is known to be optimal in the sense that it minimizes the expected sample size among all sequential tests that do not have a larger probability of error. The
analysis to check whether this optimality result holds for $D$-Markov observations is a part of future work. Extension to multi-hypothesis testing [8] and truncated SPRT [115] for current class of Markov models are also recommended as possible topics of future work. The proposed sequential detector was demonstrated to successfully detect thermo-acoustic combustion instabilities. The experimental validation on other types of experimental apparatuses and real-life physical processes will also be conducted in future.
Chapter 6

Context-aware Dynamic Sensor Selection

6.1 Introduction

Intelligent physical systems, under the paradigm of dynamic data-driven application systems (DDDAS), have to make decisions at various time scales not only for situation awareness of the system but also for adapting the measurement system to get more relevant and useful data [21]. In this process of monitoring and decision-making, the statistical behavior of interest, which is known as the state, is to be inferred from the sensor data. However, the operational conditions [46], which consist of various man-made or natural factors, affect the sensor data or its interpretation in the observed system. These factors apart from the statistical behavior of interest are collectively called as context in the literature [81, 125]. Considering the importance of correct interpretation of data in the decision-making process, ignoring context can lead to poor performance and can even have severe consequences. For example, in a border surveillance system, the measurements from geophones, which are used for intruder detection, are severely affected by soil moisture [66]; neglecting the effect of precipitation can cause frequent false-alarms or mis-detections leading to loss of trust in the surveillance system. This chapter presents a unified framework for optimal decision-making in the presence of context, which addresses the problems of context modeling, context-aware dynamic
robust sensor selection as well as state estimation.

In this work, we consider two different effects of context on a data-driven system; namely, intrinsic and extrinsic effects. Intrinsic context includes those factors which affect the measurement of the system. These typically include time-varying factors, such as environmental conditions, health of sensor, etc. For example, soil moisture, porosity, and temperature are intrinsic contexts for seismic sensors, whereas wind gusts and air temperature are those for acoustic sensors. Unlike intrinsic context, the factors which do not affect the sensor measurements for any state, but affects the interpretation of those observations, are called the extrinsic context. For example, if the border patrol agents have information about a possible drug smuggling in a particular region of the border, the penalty for missing detections of human or vehicle targets can be increased for the unattended ground sensors in that area. The interpretation of the sensor data may change due to shift in decision boundary, since the cost structure was changed. In order to develop an intuition for the role of these factors, consider the hypothesis testing decision rules (Chap. 2 in [84]) in which the likelihood ratio is compared with a chosen threshold to declare the state estimate. Intuitively, the intrinsic factors affect the likelihood ratio, whereas the extrinsic factors influence the optimal threshold. In other words, the decision threshold which controls the interpretation of measurements is affected by the extrinsic context. In the Bayesian hypothesis testing problem, the optimal threshold depends on the prior probabilities and the cost associated with declaring a state estimate, whereas, in the Neyman-Pearson hypothesis testing problem, the threshold depends on the acceptable level of false-alarm. These examples suggest that extrinsic context consists of factors or scenarios which lead to changes in state prior probabilities, cost of state estimation, and acceptable level of false-alarms, which in turn affect the decision thresholds and thus affect the interpretation of available sensor data in the hypothesis testing problem. More examples and formal definitions of context have been given in Chapter 3.

The set of extrinsic contexts cannot be extracted from sensor data, thus they must be provided by the user. A mathematical formalization of context was developed in [81] and was discussed in Chapter 3, which allows machines to extract, learn, and use intrinsic context from the data. Two different classes of techniques to learn context from data were proposed in [81] for cases in which the factors
affecting the sensing modality are known (and can be simulated) or otherwise. In some application areas, all relevant factors which significantly affect the data are known and these factors can be individually applied to the system (possibly) in a controlled environment or by using a few high fidelity simulation models. In supervised learning of context, the knowledge of these factors is used in training. On the other hand, in some application areas, this knowledge is not available \textit{a priori}, thus such systems need to learn the relevant factors during in-situ training by using unsupervised learning algorithms. These unsupervised learning techniques have already been reported in the literature, using modularity-based clustering in [81], using Gaussian mixture models in [89], and using non-parametric density estimation in [125]. This work extends the theory from the first-of-its-kind work on supervised modeling of context [81] to be used for context-aware decision-making.

The decision problem of interest in this work is of dynamic sensor selection for sequential hypothesis testing. This problem is a special type of optimal stopping problem (Chap. 4 in [11]), in which we can choose to stop by accepting one of the hypothesis and incur the penalty if it is incorrect or we can delay the decision by choosing to sample a new measurement from a particular sensor to learn more about the true hypothesis by paying the activation/sampling cost of that sensor. Sensor selection problems have been addressed in the literature over the last two decades. In [43], a dynamic programming (DP) approach led to a sensor selection scheme that takes into account computational cost and known confidence levels of various sensors. A partially observable Markov decision process was used in [110] for dynamic sensor selection in camera networks. Other sensor selection formulations for sensor scheduling in a tracking problem [36], subset selection using convex optimization [45], and information-theoretic sensor network management [133], have also been presented in the literature. Different formulations combining sensor selection and hypothesis testing problems were studied in [7, 113, 74]. Recently, a DP-based technique for optimal decision-making, which seamlessly combines sensor selection, sequential hypothesis testing, and context awareness, was developed in [124]; this was the first formulation in the literature to include the effect of intrinsic context. However, the presented framework had three restricting assumptions, namely, that the context set is known \textit{a priori}, that the context set is not dependent on the state, and that the unknown intrinsic context is static. The
current work aims to relax these assumptions by deriving a context set from data for each state and by using a Markov chain model for the spatio-temporal evolution of the random variable representing the intrinsic context. Additionally, this work introduces time-varying extrinsic context, which in this case would focus on factors that cause changes in the costs of activating and operating sensors, and in the penalties for making incorrect terminal decisions. Since the evolution of extrinsic context is unknown, it is modeled as an uncertain non-random switching parameter controlled by nature. A minimax game formulation is developed using the DP framework to address this robust dynamic sensor selection problem with uncertainty in switching of the cost structure. Thus, the problem of context-aware robust decision-making with a network of information sources is addressed in this work.

The developed sensor selection framework is validated for control of a simulated border surveillance sensor network consisting of six sensors for a binary location testing problem. The proposed method is compared with a DP-based approach which does not use context awareness in decision-making, so as to highlight the benefit of context awareness in practical problems. The comparison of dynamic sensor selection with all fixed sensor teams is also given in this chapter.

This chapter comprises of four more sections. Section 6.2 mathematically formulates a game for the problem of context-aware sequential hypothesis testing over a sensor network. In Section 6.3, the solution approach for graph-theoretic supervised learning of context as well as DP with minimax optimization for robust sensor selection are presented. Section 6.5 presents experiment details, machine learning schemes to improve tractability, and numerical results with the proposed approach. Lastly, in Section 6.6, some concluding remarks, avenues for future research, and other application domains are discussed.

## 6.2 Problem Formulation

The sensor network operation for the task of sequential hypothesis testing is described at first. Let $\mathcal{X} = \{1, 2, \ldots, M\}$ be the set of hypotheses on the value of the random state $X$ of nature, which one wishes to estimate through sequential hypothesis testing. The hypothesis set $\mathcal{X}$ is defined by the user, and in this case
we have a classification problem, where the state is to be assigned to one of \( M \) possible classes. Let \( \mathcal{S} = \{1, 2, \ldots, N\} \) be the set of all sensors, and let \( Y_i \) be the random observation of the state \( X \) from sensor \( i \in \mathcal{S} \). The sensors form a network whose connectivity is defined by a directed graph \( G = (\mathcal{S}, \mathcal{E}) \) for some \( \mathcal{E} \subset \mathcal{S}^2 \), so that activating sensor \( i \in \mathcal{S} \) and sampling observation \( Y_i \) makes all sensors \( j \) satisfying \((i, j) \in \mathcal{E}\) available for subsequent activation.

Define the set of out-neighbors of sensor \( i \) as

\[
\mathcal{N}_i^{\text{out}} = \{ j \in \mathcal{S} : (i, j) \in \mathcal{E}^2 \}
\]

for \( i \in \mathcal{S} \). Let \( \mathcal{S}_\emptyset \subset \mathcal{S} \) be the set of sensors that can initially be activated when none of the sensor measurements has been sampled. Let

\[
\mathcal{U}_\emptyset = (\{0\} \times \mathcal{S}_\emptyset) \cup (\mathcal{X} \times \{0\})
\]

be the set of actions \( u_\emptyset \) one can initially take. Here, \( u_\emptyset = (0, j) \) for some \( j \in \mathcal{S}_\emptyset \) means that one decides to activate sensor \( j \) and sample the observation \( Y_j \) from it. On the other hand, \( u_\emptyset = (\hat{x}, 0) \) for some \( \hat{x} \in \mathcal{X} \) means that one decides to estimate the state \( X \) as \( \hat{x} \) based solely on the prior information without taking any measurement of the state.

Suppose that a set \( \mathcal{T} \subsetneq \mathcal{S} \) of sensors has been selected and the observations \( Y_i \) have been sampled for all \( i \in \mathcal{T} \). Suppose that \( \mathcal{S}_\mathcal{T} \) denotes the set of additional sensors that can be activated. Then, at this stage, the set of actions \( u_\mathcal{T} \) that one can take is given as

\[
\mathcal{U}_\mathcal{T} = (\{0\} \times \mathcal{S}_\mathcal{T}) \cup (\mathcal{X} \times \{0\}),
\]

where \( u_\mathcal{T} = (0, j) \) for some \( j \in \mathcal{S}_\mathcal{T} \) means that one decides to add sensor \( j \) to the existing set \( \mathcal{T} \) of activated sensors and sample the observation \( Y_j \) before estimating the state \( X \), and where \( u_\mathcal{T} = (\hat{x}, 0) \) for some \( \hat{x} \in \mathcal{X} \) means that one decides to make a terminal decision and generates an estimate \( \hat{x} \) of \( X \) based solely on the observations from the sensors in \( \mathcal{T} \). Then, the sets \( \mathcal{S}_\mathcal{T} \) and \( \mathcal{U}_\mathcal{T} \) are updated as

\[
\mathcal{S}_{\mathcal{T} \cup \{j\}} = \begin{cases} 
(S \setminus (\mathcal{T} \cup \{j\})) \cap \mathcal{N}_j^{\text{out}} & \text{if } u_\mathcal{T} = (0, j); \\
\emptyset & \text{otherwise},
\end{cases} \quad (6.1)
\]
and
\[ U_{\mathcal{T} \cup \{j\}} = (\{0\} \times \mathcal{S}_{\mathcal{T} \cup \{j\}}) \cup (\mathcal{X} \times \{0\}) \] (6.2)

for all \( j \in \mathcal{S} \setminus \mathcal{T} \) (and hence for all \( j \in \mathcal{S}_{\mathcal{T}} \)).

The update rule (6.1) dictates that, in order to make a terminal decision on the state value based on the observations from a sensor set \( \mathcal{T} \subset \mathcal{S} \), one must follow an acyclical path \((i_0, \ldots, i_T)\) on the graph \( G \) in sequentially activating sensors, so that \((i_t, i_{t+1}) \in \mathcal{E} \) and \( u_{(i_0, \ldots, i_t)} = (0, i_{t+1}) \) for \( t = 0, \ldots, T - 1 \), and that \( \mathcal{T} = \{i_0, \ldots, i_T\} \) and \( u_\mathcal{T} = (\hat{x}, 0) \) for some \( \hat{x} \in \mathcal{X} \). Moreover, for any sensor set \( \{j_0, \ldots, j_t\} \) of cardinality \( t \), the update rule puts \( \mathcal{S}_{(j_0, \ldots, j_t)} = \emptyset \) and \( U_{(j_0, \ldots, j_t)} = \mathcal{X} \times \{0\} \) whenever either \( t \geq T \) holds or both \( \{j_0, \ldots, j_t\} \neq \{i_0, \ldots, i_t\} \) and \( t < T \) hold.

6.2.1 Context-aware Measurement Model

In most practical situations, the observations \( Y_1, \ldots, Y_N \) are not statistically independent even if they are conditioned on \( X \), unless the state \( X \) captures all factors affecting the sensor measurements. A context-aware measurement model introduces a random variable, called the context, such that the observations are conditionally independent given the state-context pair. In this section, we define the notion of context and present a Markov-chain model of context evolution. Refer to Chapter 3 for more details on the notion of context and learning of context from data. For simplicity, both probability densities (of continuous variables) and probability masses (of discrete variables) will be referred to as densities.

**Definition 6.1.** Let \( L \) be a random variable that takes values in a finite set \( \mathcal{L} \) and satisfies
\[ p(Y_1, \ldots, Y_N \mid X, L) = \prod_{i=1}^{N} p(Y_i \mid X, L), \] (6.3)

where the left-hand side (resp. right-hand side) of (6.3) denotes the joint density (resp. product of marginal densities) of the observations conditioned on the pair \((X, L)\). Then \( L \) is called the context element.

The context element \( L \) describes all natural and man-made physical phenomena that, along with the state \( X \), completely condition the observations \( Y_i, i \in \mathcal{S} \). For practical purposes, it is reasonable to suppose that the context element set \( \mathcal{L} \)
exists, and that one can partition $\mathcal{L}$ into a finite, state-dependent collection $\mathcal{C}(X)$ of equivalence classes.

**Definition 6.2.** Let $L$ be the context element taking values in $\mathcal{L}$. Let each $\mathcal{C}(x)$, $x \in \mathcal{X}$, be a finite partition of $\mathcal{L}$ such that

$$p(Y_i \mid X = x, L = l) = p(Y_i \mid X = x, L = m)$$  \hspace{1cm} (6.4)

for all $i \in S$, for all $x \in \mathcal{X}$, and for all $l, m \in \mathcal{L}$ belonging to a common member of $\mathcal{C}(x)$. A random variable $C$ that takes values in $\mathcal{C}(X)$ conditioned on $X$, so that

$$p(Y_1, \ldots, Y_N \mid X, C) = \prod_{i=1}^{N} p_i(Y_i \mid X, C)$$  \hspace{1cm} (6.5)

holds, where $p_i(Y_i \mid X, C)$ is the conditional density of the observation $Y_i$ given the state-context pair $(X, C)$, is called the (intrinsic) context.

The distribution of a context variable $C$ is induced from that of the context element $L$. That is, the conditional probability that $C = [l]$ (i.e., the equivalence class that a context element $l$ belongs to) given $X = x$ is equal to $\sum_{m \in [l]} p(m \mid x)$ for $x \in \mathcal{X}$ and $l \in \mathcal{L}$. Here, $p(L \mid X)$ denotes the conditional density of $L$ given $X$, which reflects one’s prior belief on the value of $L$ conditioned on $X$. With this understood, we can replace the members of the context sets $\mathcal{C}(x)$, $x \in \mathcal{X}$, with their labels and write $\mathcal{C}(x) = \{1, \ldots, \ell(x)\}$ for $x \in \mathcal{X}$, where $\ell(x)$ is the cardinality of $\mathcal{C}(x)$. The members of $\mathcal{L}$ and $\mathcal{C}(X)$ are referred to as context elements and contexts, respectively.

The presented definition of contexts can be generalized to a Markov-chain model of context evolution in the setting of sequential hypothesis testing. For each $x \in \mathcal{X}$, let $\mathbf{P}(x) = [p_{ij}(x)] \in \mathbb{R}^{\ell(x) \times \ell(x)}$ be a stochastic matrix whose unique invariant distribution is equal to the distribution of the context $C$ conditioned on $X = x$. (In the Bayesian view, one can deduce $\mathbf{P}(X)$ from any transition probability matrix on $\mathcal{L}$ whose unique stationary distribution reflects one’s belief on the value of $L$ conditioned on $X$.) Let $\mathcal{C}_{\emptyset} = C$.

Suppose that a set $\mathcal{T} \subset S$ of sensors has been selected and the observations $Y_i$ have been sampled for all $i \in T$. Then, given a $\mathcal{C}(X)$-valued random variable $C_T$,
define the random variable $C_{T \cup \{j\}}$ by the transition probabilities

$$p_{c_1, c_2}(x) = P\{C_{T \cup \{j\}} = c_2 \mid X = x, C_T = c_1\}$$

for all $j \in S \setminus T$, $x \in \mathcal{X}$ and $c_1, c_2 \in \mathcal{C}(x)$, where $P\{C_{T \cup \{j\}} \mid X, C_T\}$ denotes the conditional distribution of $C_{T \cup \{j\}}$ given $(X, C_T)$.

If one follows an acyclical path $(i_0, i_1, \ldots)$ on the graph $G$ in sequentially activating sensors, then the sequence $(C_{\emptyset}, C_{\{i_0\}}, C_{\{i_0,i_1\}}, \ldots)$, conditioned on the state $X$, defines a Markov chain in steady state, where the conditional distribution of each member of this sequence given $X$ is equal to that of $C$ given $X$. Also, the conditional independence property (6.5), with $C$ replaced by any member of the sequence, remains valid. Thus, the sequence $(C_{\emptyset}, C_{\{i_0\}}, C_{\{i_0,i_1\}}, \ldots)$ shall be called a Markov chain of context variables.

### 6.2.2 Cost Structure

While the context variable $C$ defined in Section 6.2.1 is capable of describing the effects of intrinsic contexts that directly affect sensor measurements, it does not model the effects of extrinsic contexts whose variation can indirectly cause changes in the interpretation of sensor measurements. We will focus on some of these extrinsic contextual effects that cause changes in the costs of activating and operating sensors, and in the penalties for making incorrect terminal decisions. In order to describe these changes, we introduce an unknown, possibly time-varying, nonrandom parameter that determines the cost structure, and formulate the problem of sequential hypothesis testing with uncertainty in the value of this parameter.

Recall that, given a set $T \subset S$ of activated sensors, the agent’s action $u_T$ is constrained to be within the action set $U_T$, which is updated according to (6.1) and (6.2). Let $X_{\emptyset} = X$ and define

$$X_{T \cup \{j\}} = \begin{cases} X_T & \text{if } u_T = (0, j); \\ 0 & \text{otherwise} \end{cases}$$

for $T \subset S$ and for $j \in S \setminus T$, so that $X_T = X$ if a terminal decision on the value of the state $X$ has not been made, and so that $X_T = 0$ otherwise. Similarly,
let $Y_\emptyset$ be a null observation (which represents the prior information and thus has no information content), and define

$$Y_{T \cup \{j\}} = \begin{cases} Y_j & \text{if } u_T = (0, j); \\ Y_\emptyset & \text{otherwise} \end{cases}$$

for $T \subsetneq S$ and for $j \in S \setminus T$, so that $Y_T = Y_{i_T}$ if one has followed an acyclical path $(i_0, \ldots, i_T)$ of the connectivity graph $G$ and activated a subset $T = \{i_0, \ldots, i_T\}$ of sensors, and so that $Y_T = Y_\emptyset$ otherwise.

For $T \subset S$, let $\theta_T$ be an unknown nonrandom parameter that takes values in a finite parameter set $\Theta_T$ and determines the cost structure at the decision stage where the subset $T$ of sensors have been activated. More specifically, we assume that, when the state $X_T = x$ and the parameter $\theta_T = \theta$, the agent’s action $u_T = u$ incurs a stage cost given by

$$g_T(x, u, \theta) = \begin{cases} \delta_T(x, \hat{x}, \theta) & \text{if } x \neq 0 \text{ and } u = (\hat{x}, 0) \text{ for some } \hat{x} \in \mathcal{X}; \\ \alpha_j(\theta) & \text{if } x \neq 0 \text{ and } u = (0, j) \text{ for some } j \in \mathcal{S}_T; \\ 0 & \text{if } x = 0 \end{cases}$$

for $(x, u, \theta) \in (\mathcal{X} \cup \{0\}) \times U_T \times \Theta_T$, where $\delta_T(x, \hat{x}, \theta) \geq 0$ is the cost associated with declaring the state estimate as $\hat{x}$ when the true state is $x$, and $\alpha_j(\theta) \geq 0$ is the cost of activating and operating sensor $j$. These costs are parameterized by $\theta$.

Note that the stage cost is zero if a terminal decision has already been made. The overall cost for the decision maker, or the agent, to minimize is the sum of stage costs over all $T \subset S$: $\sum_{T \subset S} g_T(X_T, u_T, \theta_T)$.

### 6.2.3 Information Structure

The parameter $\theta_T$, which models the uncertain extrinsic context, is controlled by a fictitious player, or nature, with no known objective or strategy. In this case, it is not unreasonable to base the agent’s decision on the assumption that nature wants to maximize the agent’s loss. This pessimistic view is known as Wald’s criterion [129], which is a standard approach to robust decision-making under uncertainty [69]. Our problem of sequential hypothesis testing is formulated as a
minimax game played by the agent and nature. While the agent takes the role of
the minimizing player who aims to minimize the cost associated with estimating
the state of nature and selectively collecting observations from a sensor network,
the overall cost structure is controlled by nature, the maximizing player.

At the initial stage where no observation has been sampled from the sensors,
the maximizing player chooses the initial parameter \( \theta_\emptyset \in \Theta_\emptyset \) based solely on the
prior information \( J_\emptyset = Y_\emptyset \). Then, the agent takes its action based on the initial
cost structure chosen by the maximizing player. That is, the information available
to the minimizing player at the initial stage (after the maximizing player has taken
an initial action) is \( I_\emptyset = (Y_\emptyset, \theta_\emptyset) \). At the stage where a subset \( \mathcal{T} \subset \mathcal{S} \) of sensors
have been activated and their observations have been sampled, if the information
vectors for the minimizing and maximizing players are \( I_\mathcal{T} \) and \( J_\mathcal{T} \), respectively,
then they are updated at the next stage as

\[
J_{\mathcal{T} \cup \{j\}} = (I_\mathcal{T}, u_\mathcal{T}), \quad (6.6a)
\]
\[
I_{\mathcal{T} \cup \{j\}} = (J_{\mathcal{T} \cup \{j\}}, Y_{\mathcal{T} \cup \{j\}}, \theta_{\mathcal{T} \cup \{j\}}), \quad (6.6b)
\]

respectively, for \( j \in \mathcal{S} \setminus \mathcal{T} \). This nested information structure indicates that the
players have perfect recall of past observations and actions, and that, at each
stage, each player can make a decision using all the information the opponent has
gathered till that stage.

The sequential decision rules for the minimizing and maximizing players, de-
noted by \( \mu = (\mu_\mathcal{T})_{\mathcal{T} \subset \mathcal{S}} \) and \( \gamma = (\gamma_\mathcal{T})_{\mathcal{T} \subset \mathcal{S}} \), respectively, are such that the action
sequences are generated according to

\[
u_\mathcal{T} = \mu_\mathcal{T}(I_\mathcal{T}) \in \mathcal{U}_\mathcal{T},
\]
\[
\theta_\mathcal{T} = \gamma_\mathcal{T}(J_\mathcal{T}) \in \Theta_\mathcal{T}
\]

for \( \mathcal{T} \subset \mathcal{S} \). The expected overall cost incurred by the agent using a decision rule \( \mu \)
in the worst-case scenario of parameter variation is

\[
J(\mu) = \max_{\gamma} \mathbb{E} \left[ \sum_{\mathcal{T} \subset \mathcal{S}} g_\mathcal{T}(X_\mathcal{T}, \mu_\mathcal{T}(I_\mathcal{T}), \gamma_\mathcal{T}(J_\mathcal{T})) \mid I_\emptyset \right],
\]
where $E[\cdot]$ denotes expectation. The objective of the agent player is to find a decision rule $\mu^*$ such that $J(\mu^*) \leq J(\mu)$ for all $\mu$.

6.3 Problem Solution

6.3.1 Bayesian Update of Posterior

Let the probability that $(X_\emptyset, C_\emptyset) = (x, c)$ at the initial stage, where no observation has been sampled, be

$$p_{x,c}(I_\emptyset) = P\{X_\emptyset = x, C_\emptyset = c\}$$

for $x \in X$ and $c \in C(x)$. Write the contextual observation densities as

$$p_j(Y_j \mid x, c) = p(Y_j \mid X_T = x, C_T = c)$$

for $j \in S, x \in X$, and $c \in C(x)$. Suppose that, at the stage where a subset $T \subseteq S$ of sensors have been activated and their observations sampled, the probabilities that the state $X_T = x$ and context $C_T = c$ conditioned on the agent’s information $I_T$ are

$$p_{x,c}(I_T) = P\{X_T = x, C_T = c \mid I_T\}$$

for $x \in X$ and $c \in C(x)$. Then, depending on the agent’s action $u_T$ and observation $Y_T$, these “prior” probabilities are updated to the “posterior” probabilities that $(X_{T \cup \{j\}}, C_{T \cup \{j\}}) = (x, c)$, which are given by

$$p_{x,c}(I_{T \cup \{j\}}) = P\{X_{T \cup \{j\}} = x, C_{T \cup \{j\}} = c \mid I_{T \cup \{j\}}\}$$

$$= \begin{cases} 
\sum_{c \in C(x)} p_j(Y_j \mid x, c)p_{\tilde{c}c}(x)p_{x,\tilde{c}}(I_T) \\
\sum_{x \in X} \sum_{c \in C(x)} \sum_{\tilde{c} \in C(x)} p_j(Y_j \mid x, c)p_{\tilde{c}c}(x)p_{x,\tilde{c}}(I_T) \\
p_{x,c}(I_T) 
\end{cases}$$

for $u_T = (0, j)$;

otherwise

for $j \in S \setminus T, x \in X$ and $c \in C(x)$. Suppose that the agent has activated the sensors in $T = \{i_0, \ldots, i_T\}$ and sampled their observations by following an acyclical path $(i_0, \ldots, i_T)$ on the graph $G$. Then, according to the principles of stochastic
control [52] and switched control [55], the agent’s information vector $I_T$ can be replaced, without loss of optimality, with the posterior probabilities $p_{x,c}(I_T)$ for $x \in X$ and $c \in C(x)$, and the past parameter sequence $(\theta_{\emptyset}, \theta_{\{i_0\}}, \theta_{\{i_0, i_1\}}, \ldots, \theta_T)$.

### 6.3.2 Minimax Dynamic Programming

We will do a backward induction. First, suppose that the process of sequential hypothesis testing with dynamic sensor selection is at the final stage, where a subset $T \subset S$ of sensors have been activated and their observations sampled, and where the set of available sensors for activation is empty (i.e., $S_T = \emptyset$). At this stage, it follows from the information structure defined by (6.6) that the value of the parameter $\theta_T$ is available to the agent (i.e., $\theta_T$ is a part of $I_T$), and so there is no uncertainty about the cost structure. Define the cost of state estimation as

$$J_{SE}(I_T) = \min_{\hat{x} \in X} \sum_{x \in X} \left( \delta_T(x, \hat{x}, \theta_T) \sum_{c \in C(x)} p_{x,c}(I_T) \right).$$

Then, the optimal cost-to-go at the stage with $S_T = \emptyset$ is

$$V_T(I_T) = \min_{u_T \in U_T} \mathbb{E} \left[ g_T(X_T, u_T, \theta_T) \mid I_T \right] = J_{SE}(I_T),$$

and the optimal decision rule at this stage is such that $\mu_T(I_T) = (\hat{x}_T, 0)$, where the optimal state estimate is equal to

$$\hat{x}_T = \arg\min_{\hat{x} \in X} \sum_{x \in X} \left( \delta_T(x, \hat{x}, \theta_T) \sum_{c \in C(x)} p_{x,c}(I_T) \right).$$

(6.7)

Next, suppose that the process of sequential hypothesis testing with dynamic sensor selection is at an intermediate stage, where a subset $T \subset S$ of sensors have been activated and their observations sampled, and where additional sensors are available for activation (i.e., $S_T \neq \emptyset$). At this stage, the agent needs to decide whether to activate and operate an additional sensor from the set $S_T$ or to make a terminal decision on the value of the state $X$ without sampling another observation.
Define the cost of sensor selection as

\[
J_{SS}(I_T) = \min_{j \in S_T} \max_{\theta \in \Theta_{T \cup \{j\}}} \left( \alpha_j(\theta) + E_\theta \left[ V_{T \cup \{j\}}(I_{T \cup \{j\}}) | I_T \right] \right),
\]

where

\[
E_\theta \left[ V_{T \cup \{j\}}(I_{T \cup \{j\}}) | I_T \right] = \sum_{x \in \mathcal{X}} \sum_{c \in \mathcal{C}(x)} \int V_{T \cup \{j\}}(I_{T \cup \{j\}}, (0, j), y, \theta) \, p_j(y | x, c) \, p_{\tilde{x}c}(x) \, p_{x, \tilde{c}}(I_T) \, dy.
\] (6.8)

Then, invoking the sequential information structure defined by (6.6), the optimal cost-to-go at a stage with \( S_T \neq \emptyset \) is deduced as

\[
V_T(I_T) = \min_{T \subset R \subset S} \max_{\mu \in \mathcal{R}} \mathbb{E} \left[ \sum_{T \subset R \subset \mathcal{S}} g_R(X_R, \mu_R(I_R), \gamma_R(J_R)) | I_T \right]
\]

\[
= \min \{ J_{SE}(I_T), J_{SS}(I_T) \}.
\]

If the inequality \( J_{SE}(I_T) \leq J_{SS}(I_T) \) holds true, then \( u_T = \mu_T(I_T) = (\hat{x}_T, 0) \), where \( \hat{x}_T \) is as in (6.7), is an optimal action. Otherwise, the optimal action is to add some sensor \( j \in S_T \) to the set \( T \) and sample its measurement. That is, the optimal decision rule at an intermediate stage with \( J_{SE}(I_T) > J_{SS}(I_T) \) is such that \( u_T = \mu_T(I_T) = (0, j_T) \), where

\[
j_T = \arg \min_{j \in S_T} \max_{\theta \in \Theta_{T \cup \{j\}}} \left( \alpha_j(\theta) + E_\theta \left[ V_{T \cup \{j\}}(I_{T \cup \{j\}}) | I_T \right] \right).
\]

Lastly, at the initial stage where none of the sensors are active (i.e., when \( T = \emptyset \)), the optimal cost-to-go gives the value of the game as follows:

\[
V_\emptyset(I_\emptyset) = \min_{\mu} J(\mu) = \min \{ J_{SE}(I_\emptyset), J_{SS}(I_\emptyset) \}.
\]

If the inequality \( J_{SE}(I_\emptyset) \leq J_{SS}(I_\emptyset) \) holds true, then it is optimal for the agent to declare the state estimate \( \hat{x}_\emptyset \) without collecting any sensor data. Otherwise, it is optimal for the agent to activate sensor \( j_\emptyset \) and sample its measurement before making a terminal decision on the state value. While the very first sensor is selected
based solely on the prior over \((X_\phi, C_\phi)\) and the initial parameter \(\theta_\phi\), all subsequent sensors are selected by the agent based on the posterior updated by measurements and the past and present extrinsic effects of context.

### 6.3.3 Contextual Value of Information

The expected value of information associated with each sensor, if quantified, would enable us to determine how much cost we should be willing to pay before making a decision to activate that sensor. In the presence of context, the value of information must capture whether a sensor can provide us with contextually relevant data, which will likely be beneficial for the state estimation task. In particular, in sequential hypothesis testing, the value of information from a new observation must account for the information already accumulated from the observations sampled in the past, so that sensors that can further disambiguate existing hypotheses are valued higher than those that just support available information. In view of the dynamic programming approach presented in Section 6.3.2, a definition for the expected value of information is proposed as follows.

**Definition 6.3.** Let \(\mathcal{T} \subseteq \mathcal{S}, j \in \mathcal{S}_\mathcal{T}, \text{ and } \theta \in \Theta_{\mathcal{T} \cup \{j\}}\). If the information available to the agent is \(I_\mathcal{T}\), and if the extrinsic contextual parameter \(\theta_{\mathcal{T} \cup \{j\}} = \theta\), then the (expected contextual) value of information associated with sensor \(j\) is defined as

\[
V(j, \theta, I_\mathcal{T}) = J_{SE}(I_\mathcal{T}) - E_\theta\left[V_{\mathcal{T} \cup \{j\}}(I_{\mathcal{T} \cup \{j\}}) \mid I_\mathcal{T}\right].
\]  

(6.9)

In this definition, the cost of state estimation with the current set of active sensors is contrasted against the expected optimal cost-to-go after adding sensor \(j\) to the set of active sensors. If \(V(j, \theta, I_\mathcal{T}) > 0\), then sensor \(j\) is a contextually relevant source of information. Moreover, if \(V(j, \theta, I_\mathcal{T}) \geq \alpha_j(\theta)\), then the value of information from sensor \(j\) is expected to be higher than the cost of activating and operating sensor \(j\) under the cost structure given by \(\theta\); that is, adding sensor \(j\) to the active sensor set \(\mathcal{T}\) will likely reduce the overall expected cost. This value-of-information notion is consistent with the minimax dynamic programming approach, which assumes that the unknown parameter \(\theta\) gives the worst-case cost structure for sensor \(j\).
6.4 Tractable Implementation of Minimax Dynamic Programming

Although the action set of state estimation and sensor selection is a finite set, the information space of sensor measurements and the belief space of probability over state-context pairs are both multi-dimensional and continuous. Once the information model/feature is extracted from time-series data, the feature space is partitioned using maximum likelihood rule in such a way that each state-context pair has a disjoint region. This partitioning uses tools of dimensionality reduction, density estimation, and machine learning, to create a finite discrete measurement space. For example, as shown schematically in Figure 6.1a, the measurement space for a seismic sensor is obtained as \( \mathcal{Y}_1 = \{y_1, y_2, y_3, y_4\} \) since \(|\mathcal{X}| = 2\) and \(|\mathcal{C}| = 2\).

The belief space over state-context pairs is a \((|\mathcal{X}||\mathcal{C}| - 1)\)-simplex plane in a \(|\mathcal{X}||\mathcal{C}|\)-dimensional space. The simplex plane is partitioned by dividing the region \(\sum_{i=1}^{(|\mathcal{X}||\mathcal{C}|-1)} p_i \leq 1\) by hyper-cubes of side \(1/K\). The proposed approximation creates a triangular-based pyramid of the hyper-cubes. The obtained discrete space consists

\[ Y_1 = \{y_1, y_2, y_3, y_4\} \]

(a) Discrete measurement space obtained by feature extraction and partitioning

(b) Belief space discretization by regular hypercubes to obtain lookup tables

Figure 6.1: Information space and belief space discretization
of \((K + \frac{|X||C| - 2}{|X||C| - 1})\) disjoint regions, and thus a discretization of the belief space was obtained.

The state set (belief space), measurement set, and action set being finite allows us to use backward induction to create a feedback policy, which can be stored as a lookup table on the embedded system of sensors, allowing online control of the sensor team formation.

6.5 Simulation Results and Discussion

The dynamic sensor team formation and state estimation approach presented in this chapter is validated in this section using a few simulation experiments. The objective of the simulations is to test the following hypotheses:

1. For fixed extrinsic context, context-aware dynamic sensor selection and state estimation performs better than all fixed teams in terms of cost of decision-making.
2. Sensors form bigger teams as misclassification cost increases.
3. For context-aware dynamic sensor selection, the decision sequence changes with change in cost structure for the same measurement sequence.
4. For fixed cost structure, context-aware dynamic sensor selection performs better than dynamic sensor selection without context awareness.

The simulation setup and the results are discussed ahead in this section.

The simulated sensor network includes 6 sensors with a communication topology as shown in Figure 6.2. This topology represents the order in which the sensors
can be activated and the directed edge from $i$ to $j$ denotes that sensor $i$ can invite sensor $j$ to join the team. The node $\phi$ denotes the initial step, when none of the sensors are active, and sensor 1 or 4 can be activated. The node Stop denotes the final stage, where none of the sensors can be added to the team and the state estimate has to be declared. Note that the formulation allows to go to the Stop node from any node in the graph, thus it is equivalent to the input of $(\hat{x}, 0)$. There are several simulation tests, which can be done with the sensor network to test/validate the list of hypotheses given above. Four different tests were conducted on the simulation testbed to illustrate the capabilities of the proposed sensor selection framework and also test the list of hypotheses given above. The description of the tests, their results, and the related discussions are given ahead. For all the analysis, the dataset was randomly partitioned into training set (70%) and test set (30%) and the analysis was repeated 10 times to get the average error. The tests are as follows:

1. **Fixed versus dynamic teams**: In the sensor network shown in Figure 6.2, there are in total 61 paths in the directed graph to go from $\phi$ to Stop node, including the edge directly connecting $\phi$ to Stop node. If each of these 60 paths is assumed to be a fixed team, then the state estimation performance of these teams is compared with the dynamic team formed by the proposed sensor selection approach. The average cost of decision-making over the test data is given in Figure 6.3a. It shows that the dynamic sensor selection is able to perform better than all fixed teams. The classification accuracy with the dynamic approach is slightly lower than the best team, as shown in Figure 6.3b, whereas the average team size is smaller than most fixed teams, as shown in Figure 6.3c. Over the test data set, the team size distribution of the dynamic sensor selection approach is shown in Figure 6.3d, which shows that mostly smaller-sized teams are formed. Thus, this simulation supports the hypothesis that context-aware dynamic sensor selection and state estimation performs better than all fixed teams in terms of cost of decision-making.

2. **Effect of misclassification cost**: The sensor team formation depends not only upon the confidence in decision-making of the existing team, but also on the cost of adding new sensors and the cost of misclassification. If the cost of misclassification is higher, then the sensor team must be more confident before it
declares the state, thus the team will add more sensors. This intuition holds true in this simulation test. In Case 1, the cost of misclassification is 100, whereas in Case 2 that cost is 1000, thus bigger teams are formed in Case 2. As shown in Figure 6.4, the team size distribution has shifted more towards the right. Moreover, the classification accuracy also improves in Case 2 as expected, as shown in Table 6.1. Thus, this test supports the hypothesis that sensors form bigger teams as the misclassification cost increases.

Table 6.1: Dynamic sensor team performance for different misclassification costs

<table>
<thead>
<tr>
<th>Case</th>
<th>Misclassification Cost</th>
<th>Accuracy</th>
<th>Team Size</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>100</td>
<td>98.49%</td>
<td>1.71</td>
<td>3.22</td>
</tr>
<tr>
<td>2</td>
<td>1000</td>
<td>99.31%</td>
<td>2.30</td>
<td>9.19</td>
</tr>
</tbody>
</table>
Table 6.2: Dynamic sensor team performance for different sequences of extrinsic context

<table>
<thead>
<tr>
<th>Case</th>
<th>Extrinsic Context Sequence</th>
<th>Accuracy</th>
<th>Team Size</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>[1, 1, 1, 1, 1, 1]</td>
<td>97.11%</td>
<td>1.00</td>
<td>3.89</td>
</tr>
<tr>
<td>2</td>
<td>[1, 2, 1, 1, 1, 1]</td>
<td>98.44%</td>
<td>1.78</td>
<td>5.34</td>
</tr>
</tbody>
</table>

3. Effect of time-varying extrinsic context: In this simulation test, the extrinsic context ($\theta$) is modeled to influence the misclassification cost. If $\theta = 1$, then uniform misclassification cost is 100, and for $\theta = 2$, the cost is 1000, i.e., $\delta_T(1, 2, 1) = \delta_T(2, 1, 1) = 100$ and $\delta_T(1, 2, 2) = \delta_T(2, 1, 2) = 1000$ for all $T \subseteq S$. Since the switching between the extrinsic context parameter values is uncertain
Table 6.3: Team formation examples for two different sequences of extrinsic context

<table>
<thead>
<tr>
<th>Observation</th>
<th>Case</th>
<th>Team</th>
<th>Cost</th>
<th>X</th>
<th>Ÿ</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>{4}</td>
<td>101</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>{4, 1, 3}</td>
<td>3</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>{4}</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>{4, 1}</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

and only the current extrinsic context is known, the sensor network controller minimizes the worst case expected loss. In Case 1, although there is no switching in extrinsic context, the state estimate is declared with only one sensor in the team. It seems that the controller is hoping to avoid the scenario in which extrinsic context switches to 2, since that has much higher misclassification penalty. Note that Case 1 in Table 6.2 had misclassification cost of 100 throughout, which is the same as the Case 1 in Table 6.1; however, due to the possibility of uncertain switching, the controller was more conservative and the performance was poorer. Here, in Case 2 before the controller could declare the state, the extrinsic context shifted to $\theta = 2$. Making decisions with small confidence would be considered a risky strategy and will have higher expected cost due to higher probability of misclassification, thus it seems that the robust controller decides to add more sensors before declaring the state estimate, as seen in Figure 6.5. From Table 6.2, we can see that in Case 2 the teams are bigger and the performance is better too.

In order to understand the effect of switching of extrinsic context, consider the specific team formation examples given in Table 6.3. Although sensor 4 received same observation in both cases, the controller decided to stop and declare the state and got incorrect output in Case 1, whereas more sensors were added to the team and correct state estimate was obtained in Case 2. In another example, sensor 4 obtained correct state estimate in Case 1, but it was not confident enough, hence sensor 1 was added to the team in Case 2, as misclassification cost was higher due to the switching of extrinsic context. Hence, this simulation test demonstrates that the robust controller is able to respond to uncertain switching of extrinsic context and the same observations might lead to different behaviors under disparate extrinsic contexts.
Table 6.4: Dynamic sensor team performance with and without context awareness

<table>
<thead>
<tr>
<th>Case</th>
<th>Mean Accuracy</th>
<th>Mean Team Size</th>
<th>Mean Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>With Context</td>
<td>95.11 ± 0.008%</td>
<td>3.68 ± 0.11</td>
<td>52.57 ± 7.75</td>
</tr>
<tr>
<td>Without Context</td>
<td>91.84 ± 0.011%</td>
<td>4.36 ± 0.24</td>
<td>85.91 ± 10.86</td>
</tr>
</tbody>
</table>

4. Impact of context awareness: In this simulation test, the proposed context-aware dynamic sensor selection framework is compared with the framework that does not use the notion of context in sensor selection in terms of accuracy, number of sensors used, and cost of decision-making. A binary classification problem was simulated, i.e., $|X| = 2$. The knowledge of context was available during the training phase and the context elements were binary too, i.e., $|C(X = 1)| = |C(X = 2)| = 2$. These two context labels along with the two states were used to partition the feature space into 4 parts using supervised learning approach to obtain a discrete measurement, which will be used in sensor selection. In the alternative approach, the state label was used to partition the measurement space into two parts. In this test, the extrinsic context was fixed and known, whereas the intrinsic context was fixed but unknown. The cost structure included uniform misclassification cost of 1000 and uniform sensor activation cost of 1. The results in Table 6.4 show the mean performance (along with ±1 standard deviation) after repeating
the test 10 times and the context-aware approach has better performance in terms of cost of decision-making, sensor team size, and classification accuracy. In other words, context awareness enables the framework to use fewer sensors, as shown in Figure 6.6, and yet make more accurate estimation. Note that the standard deviation of the different performance metrics is also lower with context awareness. Thus, this simulation illustrates the benefit of using context awareness in a sensor selection framework and it supports the hypothesis that for a fixed cost structure, context-aware dynamic sensor selection performs better than dynamic sensor selection, which does not use context awareness.

It is important to point out that the measurement model used to simulate the data in the first 3 tests was identical for all sensors, whereas, to accommodate different contextual sensitivities of different modalities, distinct models were used for some of the sensors in the last test.

### 6.6 Conclusion and Future Work

A dynamic data-driven context-aware robust controller is developed in this chapter to control a sensor network for target classification. The main contributions presented in this chapter are summarized as follows:

- Development of a graph-theoretic approach for supervised learning of context from data;
- Minimax dynamic programming is formulated as a game against nature to address the dynamic robust sensor selection for sequential hypothesis testing with:
  - Probabilistic temporal evolution of intrinsic context;
  - Uncertainty in switching of extrinsic context;
- Formalization of a definition for expected contextual value of information of a sensor to identify contextually relevant data in a particular operating condition;
- Machine learning-based scheme to extract discrete measurements from sensor data to enable tractable implementation of the DP framework in sensor networks;
- Validation of the proposed approach with simulation of a border surveillance system;
• Comparison of the developed framework with one which does not use context awareness in the formulation.

The simulation experiments verified the following hypotheses: (1) dynamic sensor selection performs better than all fixed sensor teams; (2) increases in the cost of misclassification leads to bigger teams in homogeneous networks; (3) switching of extrinsic context affects the team formation for the same measurement sequence; (4) Context-awareness improves performance.

In the future, the contextual value of information formulation will be used to identify sensors that are useful and those that can be pruned out from the network. The context-aware sensor selection process will be validated using field experiments with the Border Control testbed (Appendix A) and also using problems on machine monitoring. The context-aware sensor selection problem will be modified to include human operators as information interpreters, which can be invited to join the team and shown relevant data only if needed, thereby reducing cognitive burden on human operators.
Chapter 7

Sequential Learning of Measurement Models from Imperfect Instructors

7.1 Introduction

Sensors are used for monitoring of complex systems to assess the underlying state of the system, such as, health monitoring in physical systems or target detection/classification in surveillance systems. Some of these sensors (e.g., cameras) are easy to train with supervised learning approaches due to abundance of labeled training data. However, for other sensing modalities (e.g., geophones), which do not have adequate training data or which are more sensitive to the operating conditions of the environment, we need to collect and label in-situ training data to obtain the corresponding measurement models. In this work, an alternative approach to train generative models for new sensing agents is presented by using labels generated by existing sensors in the sensor network. The sensors which contribute (possibly incorrect) labels to educate the new sensor are called imperfect instructors. The objective of this work is to enable the sensors to learn true measurement models even after being trained with noisy labels from multiple sensors.

The relationship between observations/features and states/targets can be represented as the conditional probability \( p(Y|X) \), where \( Y \) is the measurement/feature from the sensor when the system state is \( X \). This conditional probability is referred to be the measurement model of the sensor and it does not assume any specific structure of the relationship and/or noise characteristics. This chapter presents a
sequential online learning framework to estimate the true measurement model for a new sensor by using estimated labels of discrete states from multiple (possibly heterogeneous) sensors in a network. The concepts of probabilistic graphical models and conditional independence given the state are used to develop a linear relationship between observation densities conditioned on the labels from instructors and the (true) measurement model. If the underlying linear operator is invertible, then it is implied that the existing sensor is qualified to be an instructor and that the (true) measurement model can be inferred eventually. This work builds upon the theory of recursive density estimation [130, 64] to develop a sequential estimator of the conditional observation density. This estimator uses the observed feature and estimated labels from several sensors acting as instructors.

Let us consider an example of a border surveillance network with cameras, geophones, and microphones. If we want to add more geophones to minimize false alarms and eventually reduce the computation and power used by cameras, then we need to train the target models for geophones. However, geophones are known to be sensitive to various soil conditions [66] and thus perform better with in-situ training data [128]. Due to high sampling rate and persistent usage, abundant in-situ data can be acquired; however, providing training labels to those samples manually may not be feasible. Thus, in a DDDAS paradigm [21], where data are often used during operation to improve models, it is of interest to explore if the existing sensors can be used to train the target models for the new sensors. Moreover, the high fidelity sensors might be used till the new sensors are calibrated to the environment, which should reduce long-term operating cost of the network. This concept is also applicable to health monitoring of engineering systems.

Frénay and Verleysen [30] have discussed several sources of label noise and have also reviewed several learning methods that are robust to label noise. The source of label noise in the current framework is due to imperfect classification performance of the sensors that assume the role of instructors for the new sensors. The problem of learning with label noise has been addressed in literature for one-step classification problems using batch processing approaches [75], where the noise-corrupted training set is available at the outset. The presented approach provides a sequential probabilistic approach to update the measurement models while removing the effect of label noise. Scott et al. [105] have addressed a clas-
sification problem in which the class-dependent label noise is estimated from the data and measurement models are recovered by maximal denoising. Unlike [105], the class-dependent label noise statistics are assumed to be known in the present work, because the measurement models and classifiers of instructors are assumed to be known a priori. Since multiple heterogeneous sensors act as instructors, the label noise statistics are modeled to be sensor-dependent and hence instructor-dependent. In order to develop a sequential estimation approach, we have used the classical work in statistics on recursive density estimation from [130, 64, 63], which extends the kernel density estimators developed by Rosenblatt [92] and Parzen [80] for recursive estimation. The consistency and convergence results in those works have been leveraged in the analysis of the proposed approach.

The current work considers the case in which an expert may not be able to provide adequate labels to observations and therefore one must rely on other sensors to aid in learning the measurement model and classifier for a new sensor. This perspective is fundamentally different from most of the work reported in literature and has not yet been explored to the authors’ knowledge. Specifically, this work derives a recursive density estimation approach to sequentially learn the true measurement model by using labels, estimated from several sensors and their corresponding state-dependent and sensor-dependent label noise statistics. In this chapter, the consistency of the derived estimator is shown and the necessary and sufficient conditions for the sensors to act as instructors is presented, and the convergence rate is derived as a function of the number of updates and instructor-dependent label noise statistics. In addition, the proposed learning method for intelligent student is compared with two extreme cases – blessed student, where the correct labels are provided by an oracle and naïve student, where the new sensor may incorrectly assume that it always obtains the correct labels. The performance evaluation is shown by means of simulations with a synthetic data set in order to compare the error from the ground truth.

This section introduced the problem of learning from other sensors. Section 7.2 mathematically formalizes the problem using probabilistic graphical models. Section 7.3 provides a concrete approach to solve the problem using recursive density estimation and linear algebra. The developed framework is validated using simulation results in Section 7.4. In Section 7.5, some concluding remarks are given.
7.2 Problem Formulation

This section formulates the sequential learning problem. Let $\mathcal{X} = \{1, 2, \ldots, L\}$ be the finite set of hypotheses on the random state $X$ of the system, which one wishes to estimate through observations. The hypothesis set $\mathcal{X}$ is known a priori. We have a classification problem in the present study, where the state is to be assigned to one of the $L$ possible classes. Let $\mathcal{S} = \{1, 2, \ldots, M\}$ be the finite set of all existing sensors, and let $Y_s$ be the random observation of the state $X$ from sensor $s \in \mathcal{S}$ and $\hat{X}_s$ be the estimated state from sensor $s \in \mathcal{S}$ using a deterministic mapping $h_s$ from the measurement space $Y_s$ to the state set $\mathcal{X}$. The measurement model $p(Y_s|X)$ and classifier $h_s$ are known for all sensors $s \in \mathcal{S}$, thus the conditional probability $p(\hat{X}_s|X) = p(h_s(Y_s)|X)$ characterizing the performance of the sensor $s$ is also known.

Let $Y_0$ be the random observation from the new sensor which takes the value $y_0^t \in Y_0$ at the sampling instant $t \in \mathbb{N}$. At every sampling instant, the new sensor has access to $z^t = (y_0^t, \{\hat{x}_1^t, \hat{x}_2^t, \ldots, \hat{x}_M^t\})$, which includes the measurement $y_0^t$ and the estimated labels from all other sensors. It is noted that $\hat{x}_s^t \in \mathcal{X} \cup \{\phi\}$, where $\phi$ is the null value implying that the label from sensor $s$ was not available at the instant $t$; the provision of null value allows the existing sensors to say “I don’t know” and also allows the sensors to have different sampling frequencies. The objective of this work is to use the sequence $\{z^t\}$ to obtain a consistent estimate $\hat{p}(Y_0|X = x)$.
of the measurement model of the new sensor for all states \( x \in \mathcal{X} \). It is assumed that measurements are independent conditioned on the state, and the sequence of measurements from each sensor are also independent. Furthermore, the sensors are assumed to be collocated in the sense that the same state affects all the measurements. The probabilistic graphical model in Figure 7.1 exhibits dependencies among the random variables of state, observations, and state estimates.

### 7.3 Technical Approach

This section first considers a single instructor (i.e., \( M = 1 \)) to obtain the relationship between conditional densities for a given estimated label, and then derives the sequential estimator of the measurement model. This framework of a single instructor is extended to allow multiple instructors.

#### 7.3.1 Relation between Conditional Densities

The relation between the observation densities conditioned on the estimated label and the true observation density is obtained by factorization of the model in Figure 7.1. Using marginalization, chain rule of probability, and conditional independence of \( Y_0 \) and \( \hat{X}_1 \) given \( X \), the following relation is obtained:

\[
p(Y_0|\hat{X}_1 = i) = \sum_{j=1}^{L} p(Y_0, X = j | \hat{X}_1 = i) \\
= \sum_{j=1}^{L} p(Y_0 | X = j, \hat{X}_1 = i)p(X = j | \hat{X}_1 = i) \\
= \sum_{j=1}^{L} p(Y_0 | X = j)p(X = j | \hat{X}_1 = i) \\
= \sum_{j=1}^{L} \alpha_{ij}^1 p(Y_0 | X = j),
\]

(7.1)

where the constants \( \alpha_{ij}^1 \) are obtained by using Bayes’ rule as:

\[
\alpha_{ij}^1 = p(X = j|\hat{X}_1 = i) = \frac{c_{ij}^1w_j}{\sum_{k=1}^{L} c_{ik}^1w_k}.
\]
These constants depend on prior probabilities \( \{w_j = p(X = j)\} \) and classification performance \( \{c^1_{ij} = p(\tilde{X}_1 = i|X = j)\} \) of sensor 1. It is noted that, in (7.1), the observation density conditioned on an estimated label is a mixture model of true measurement densities with known mixture weights but unknown components. The objective here is to identify the components of the mixture model.

Let us denote \( p(y|X = i) \) by \( p_i(y) \) and \( p(y|\tilde{X}_1 = i) \) by \( \tilde{p}_i(y; 1) \) for brevity. Then, (7.1) is rewritten for all \( i \in \mathcal{X} \) and for each \( y \in \mathcal{Y}_0 \) in a matrix form as:

\[
\begin{bmatrix}
\tilde{p}_1(y; 1) \\
\vdots \\
\tilde{p}_L(y; 1)
\end{bmatrix}
= \begin{bmatrix}
\alpha^1_{11} & \cdots & \alpha^1_{1L} \\
\vdots & \ddots & \vdots \\
\alpha^1_{L1} & \cdots & \alpha^1_{LL}
\end{bmatrix}
\begin{bmatrix}
p_1(y) \\
\vdots \\
p_L(y)
\end{bmatrix},
\]

\[
\tilde{P} = A_1 P.
\]

If \( A_1 \) is invertible, then

\[
P = A_1^{-1} \tilde{P} = B_1 \tilde{P}.
\] (7.3)

Matrix \( A_1 \) is depends on prior state probabilities and classification performance of sensor 1, thus if \( A_1 \) is invertible, then sensor 1 can be an instructor for the new sensor. The implication of the invertibility condition will be discussed further in Section 7.3.3. Using (7.3), the measurement model can be obtained as a linear combination of the probability densities conditioned on the estimated labels as:

\[
p_i(y) = \sum_{j=1}^L \beta^1_{ij} \tilde{p}_j(y; 1),
\] (7.4)

where constant \( \beta^1_{ij} \) is an element of the matrix \( B_1 \) and the equality holds for each \( y \in \mathcal{Y}_0 \). Thus, if we can estimate \( \tilde{p}_j(y; 1) \) for all \( j \in \mathcal{X} \) using the sequence \( z^t = (y^t, \tilde{x}^t_1) \), then we can use (7.4) to estimate the true measurement model for each state \( i \in \mathcal{X} \). In the following section, we will derive a sequential estimator for the measurement models.
7.3.2 Recursive Density Estimation

In this part, a recursive density estimator is used to sequentially estimate $\tilde{p}_j(y;1)$ for all $j \in \mathcal{X}$ and then an estimator for $p_i(y)$ is derived using (7.4). In the observation-label pair $z^t$, if the label from instructor $\hat{x}_1^t$ is $j$ and observed measurement is $y_t$, then conditional density $p(Y|\hat{X}_1 = j)$ can be estimated by sequential learning. Using the recursive density estimator from [130], the observation density conditioned on label is updated as follows:

$$\tilde{p}_j^t(y;1) = \frac{n_j^t - 1}{n_j^t} \tilde{p}_j^{t-1}(y;1) + \frac{1}{n_j^t h(n_j^t)^d} K \left( \frac{y - y_t}{h(n_j^t)} \right),$$

(7.5)

where $n_j^t$ is the number of observations which have been assigned label $j \in \mathcal{X}$ in $t$ time steps, $K(\cdot)$ is a kernel function, $h(n)$ is the kernel width after $n$ updates, $d$ is the dimensionality of the observation, and increment $n_j^t = n_j^{t-1} + 1$. For all other labels $i \in \mathcal{X} \setminus j$, the densities remain the same $\tilde{p}_i^t(y;1) = \tilde{p}_i^{t-1}(y;1)$ and observation counts remain the same $n_i^t = n_i^{t-1}$. Note that if the label from an instructor is unavailable, i.e., $j = \phi$, then all densities and observation counts stay the same for that instructor.

Under certain assumptions on kernel function $K(\cdot)$ and kernel bandwidth sequence $\{h(n)\}$, leveraging results from [130, 64], it is shown that $\tilde{p}_i^t(y;1)$ is an unbiased and consistent estimator of $\tilde{p}_i(y;1)$. The relevant assumptions and the theoretical results are given in Section 7.3.3. Using all observation densities conditioned on estimated label, we can obtain the estimate of the measurement model sequentially using (7.4) as follows:

$$p_i^t(y) = \sum_{j=1}^L \beta_{ij}^t \tilde{p}_j^t(y;1),$$

(7.6)

7.3.3 Analysis

This section first gives the assumptions on the kernel (See Assumption 7.1, 7.2) and the kernel bandwidth sequence (See Assumption 7.3). These assumptions allow us to leverage the results from [130] to prove that the proposed density estimator is consistent and provide an upper bound of the error.
Assumption 7.1. Kernel $K$ is a Borel-measurable function, which is bounded, i.e., $\sup_{y \in Y} |K(y)| < \infty$, absolutely integrable, i.e., $\int_{y \in Y} |K(y)| < \infty$, and $\lim_{\|y\| \to \infty} |yK(y)| = 0$.

Assumption 7.2. Suppose the function $K$ has Fourier transform $K^\ast$, i.e., $K^\ast(u) = \int_{-\infty}^{\infty} e^{-iuy} K(y) \, dy$, then for some $r \in \mathbb{N}$, $\lim_{u \to 0} \{1 - K^\ast(u)/|u|^r\} = k_r$ is finite and the $r$th derivative of the observation density exists, i.e., $(\tilde{p}_i)^{(r)}(y; s)$ exists for all $s \in S$.

Assumption 7.3. The kernel bandwidth sequence satisfies $h(n) \to 0$ and $nh(n) \to \infty$. More specifically, the sequence is considered to be $h(n) = bn^{-\gamma}$ with $\gamma \in (0, 1)$ and $b > 0$.

Assumption 7.4. The constant $r \in \mathbb{N}$ from assumption 7.2 and the constant $\gamma \in (0, 1)$ from assumption 7.3 satisfies $\gamma r < 1$.

Using these assumptions on the kernel and the kernel bandwidth sequence, the error in conditional density given label from instructor becomes uniformly bounded based on the following lemma.

Lemma 7.1. If the assumptions 7.1, 7.2, 7.3, and 7.4 hold, then the error satisfies the following relation:

$$\sup_{y \in Y_0} |\tilde{p}_j(y; 1) - E\tilde{p}_j^* (y; 1)| = O((n_j^r)^{-\gamma r}).$$

Proof. The Theorem 1(b) from [130] will be used to prove this Lemma 7.1. The theorem is given here for completeness. Suppose that unknown density be represented by $f(y) = \tilde{p}_i(y; s)$ and the recursive estimator from (7.5) is $f^*_t(y)$.

Theorem 1(b) from [130]: Suppose kernel $K$ satisfies assumption 7.1, 7.2 and kernel bandwidth sequence $h(n)$ satisfies $h(n) \to 0$ and $nh(n) \to \infty$. Suppose that $nh(n)^r \to \infty$ and $1/(nh(n)^r) \sum_{j=1}^{n} h(j)^r$ converges to some $\gamma_r \in \mathbb{R}$, where $r$ is given in assumption 7.2. Then

$$\frac{ Ef_t^*(y) - f(y)}{h(n)^r} \to \gamma_r k_r f^r(y).$$
Note that if the kernel sequence is chosen to be $h(n) = bn^{-\gamma}$ (as in this work), then $\gamma_r = 1/(1 - \gamma r)$, so $Ef_t^*(y) - f(y) = O(n^{-\gamma})$, since $\|f^*\|_\infty$ is bounded (assumption 7.2).

The true observation density can be computed, if all error probabilities are known, that is, $p(\hat{X}_s = i|X = j)$ is known for all $i, j \in X$. The necessary and sufficient condition for a sensor $s \in S$ to be an instructor (i.e., enable unique estimation of true measurement density) is that matrix $A_s = [\alpha_{ij}^s]$, where $\alpha_{ij}^s$ is given in (7.2), is invertible. The implications of the invertibility condition and the approach to address the non-invertible case is given in the remark ahead.

**Remark 7.1 (Invertibility condition and non-invertible cases).** The invertibility condition depends on the prior probability and the instructor’s classification performance as follows:

**Prior probability:** The matrix $A_s$ is not invertible if the prior probability of any class $j$ is 0, i.e., $w_j = 0$. In this case, the measurement model $p(Y|X = j)$ for all $j$ with $w_j = 0$ cannot be obtained from this technique as the prior probability suggests that the class $j$ cannot occur. However, an estimate for models of other classes can be obtained by removing the columns corresponding to the zero probability classes from $A_s$ to obtain matrix $\tilde{A}_s$ with full column rank. The pseudo-inverse of this non-square matrix $\tilde{A}_s$ is then computed as $B_s = (\tilde{A}_s^T\tilde{A}_s)^{-1}\tilde{A}_s^T$.

**Instructor’s classification performance:** If all prior probabilities are positive, then the matrix $A_s$ is not invertible if the columns are linearly dependent. Here linear dependence implies that the instructor is confused between certain classes or performs poorly to detect some specific classes. The measurement model corresponding to these classes cannot be recovered from this technique as the instructor more often wrongly labels the observations. If the linearly dependent columns are removed from $A_s$ to obtain matrix $\tilde{A}_s$, then the pseudo-inverse of $\tilde{A}_s$ can be used to obtain the density estimates for other classes.

It is noted that, if $M = 1$, then the invertibility condition would also be the condition for identifiability of the unknown measurement models and that condition does not depend on the underlying true observation densities.
Theorem 7.1. (Main Result) Suppose that assumptions 7.1, 7.2, 7.3, and 7.4 hold. If \( A_1 = [\alpha_{ij}^1] \), from (7.2), is invertible and the inverse is given by \( B_1 = [\beta_{ij}^1] \), and if the proposed estimator given in (7.6) is used to estimate the conditional density \( p(Y_0 \mid X = i) \) given as \( p_i(y) \), then the error in estimation is given by:

\[
\sup_{y \in \mathcal{Y}_0} |p_i(y) - E_{p_i}(y)| = O\left( \sum_{j=1}^{L} |\beta_{ij}^1| \right),
\]

where \( E \) denotes the expectation, \( n_j^t \) is the number of observations labeled as \( \hat{x}_1 = j \) after \( t \) total updates.

Proof. Using the result from Lemma 7.1 and definition of big O notation, we know that for some \( n_j^t \geq \tilde{n}_j \), there exists a \( \theta_j \in \mathbb{R}^+ \) satisfying:

\[
\sup_{y \in \mathcal{Y}_0} |\tilde{p}_j(y; 1) - E_{\tilde{p}_j}(y; 1)| \leq \theta_j (n_j^t)^{-\gamma r}.
\] (7.7)

On the other hand, taking expectation on (7.6) and using (7.4), we get:

\[
p_i(y) - E_{p_i}(y) = \sum_{j=1}^{L} \beta_{ij}^1 \left( \tilde{p}_j(y; 1) - E_{\tilde{p}_j}(y; 1) \right).
\]

Using Jensen’s inequality [76], for the absolute value (convex) function, we obtain:

\[
|p_i(y) - E_{p_i}(y)| \leq \sum_{j=1}^{L} |\beta_{ij}^1| |\tilde{p}_j(y; 1) - E_{\tilde{p}_j}(y; 1)|.
\]

Combining this result with (7.7) and using \( \theta_{\max} = \max_{j \in \mathcal{X}} \theta_j \):

\[
|p_i(y) - E_{p_i}(y)| \leq \sum_{j=1}^{L} |\beta_{ij}^1| \theta_j (n_j^t)^{-\gamma r},
\]

\[
|p_i(y) - E_{p_i}(y)| \leq \theta_{\max} \sum_{j=1}^{L} |\beta_{ij}^1| (n_j^t)^{-\gamma r},
\]

for some \( n_j^t \geq \tilde{n}_j \) for all \( j \in \mathcal{X} \) for any \( y \in \mathcal{Y}_0 \). Then using multivariate big O...
notation, we get:

$$\sup_{y \in \mathcal{Y}_0} |p_i(y) - \mathbb{E}p_i^t(y)| = O\left(\sum_{j=1}^L |\beta_{ij}^1| (n_j^t)^{-\gamma r}\right).$$

This theorem gives the error as a function of number of different training labels from a particular instructor, the performance of classifier of the instructor, and prior class probabilities. In an extreme case, if we consider that the instructor is perfect, then the following remark shows the error relationship.

**Remark 7.2 (Oracle Instructor).** If prior class probabilities $p_j \in (0,1)$, then the perfect instructor is modeled by matrix $A = I_{L \times L}$, i.e., identity matrix of size $L$. This implies that if $i = j$, then $\beta_{ij}^1 = 1$, else $\beta_{ij}^1 = 0$. Using result of Theorem 7.1, we get $\sup_{y \in \mathcal{Y}_0} |p_i(y) - \mathbb{E}p_i^t(y)| = O((n_i^t)^{-\gamma r})$. This result agrees with the known result for recursive density estimation and is an extreme case of our model, which considers imperfect instructors too.

The number of examples needed from an imperfect instructor to get an error of the same order as that from an oracle is given in the next remark.

**Remark 7.3 (Relative Sample Complexity).** Let us assume that we have equal number of observations for each of the $K$ labels after $N$ updates from imperfect instructor and after $N_o$ updates from an oracle. In order to have an error of same order, the following relation must hold for all $i \in \mathcal{X}$ from Remark 7.2 and Theorem 7.1:

$$\left(\frac{N_o}{K}\right)^{-\gamma r} = \sum_{j=1}^K |\beta_{ij}^1| \left(\frac{N}{K}\right)^{-\gamma r}.$$

Rearranging this relation we get:

$$\tilde{\beta} = \frac{N}{N_o} = \left(\sum_{j=1}^L |\beta_{ij}^1|\right)^{\frac{1}{\gamma r}}.$$

Thus, the error introduced by imperfect instructions can be compensated by the proposed estimator in (7.6) and (7.5) by using $\tilde{\beta}$ times more observations.

For example, consider a binary state scenario, one sensor as instructor with equal mis-detection rate and false alarm rate, and the prior probability of each
Figure 7.2: Relative sample complexity for different misclassification rates

class is 0.5, then relative sample complexity from Remark 7.3 gives us that $\tilde{\beta} \approx 3$, if misclassification rate is 0.1; whereas, $\tilde{\beta} \approx 1.7$, if misclassification rate is 0.05. Thus, learning from a noisy sensor is indeed practically possible according to this analysis. The value of $\tilde{\beta}$ for different values of misclassification rates from $[0, 0.5)$ is given in Figure 7.2. Note that as misclassification rate approaches 0.5, $\tilde{\beta} \to \infty$.

This section has developed a sequential estimator for the measurement density of a single instructor (i.e., $M = 1$). The framework is now extended for multiple instructors (i.e., $M \geq 2$) in the following part. The linear combination of unknown measurement densities in (7.1) has been shown to be equal to the label-conditioned observation densities, obtained from labels from an instructor. Since the unknown measurement densities are the same irrespective of the instructor, the relation in (7.1) can be generalized for $M \geq 2$ as follows:

$$
\begin{bmatrix}
\tilde{p}_1^t(y; 1) \\
\vdots \\
\tilde{p}_L^t(y; 1) \\
\tilde{p}_1^t(y; 2) \\
\vdots \\
\tilde{p}_L^t(y; 2) \\
\vdots \\
\tilde{p}_1^t(y; M) \\
\vdots \\
\tilde{p}_L^t(y; M)
\end{bmatrix}
= 
\begin{bmatrix}
\alpha_{11}^t & \cdots & \alpha_{1L}^t \\
\vdots & \ddots & \vdots \\
\alpha_{L1}^t & \cdots & \alpha_{LL}^t \\
\alpha_{11}^2 & \cdots & \alpha_{1L}^2 \\
\vdots & \ddots & \vdots \\
\alpha_{L1}^2 & \cdots & \alpha_{LL}^2 \\
\vdots & \ddots & \vdots \\
\alpha_{11}^M & \cdots & \alpha_{1L}^M \\
\vdots & \ddots & \vdots \\
\alpha_{L1}^M & \cdots & \alpha_{LL}^M
\end{bmatrix}
\begin{bmatrix}
p_1(y) \\
p_2(y) \\
\vdots \\
p_L(y)
\end{bmatrix}
$$
\[ \tilde{P} = A \, P, \]
\[ P = (A^T \, A)^{-1} \, A^T \, \tilde{P} = B \, \tilde{P}. \]

If \( A \) has full column rank, then \( B \) is well-defined and a linear estimator of the measurement density is obtained as follows:

\[ p(Y_0 \mid x = i) = \sum_{s=1}^{M} \sum_{j=1}^{L} \beta_{ij}^s p(Y_0 \mid \hat{x}_s = j). \quad (7.8) \]

where, for brevity, the elements of \( B \) are denoted as \( \beta_{ij}^s \) instead of \( \beta_{i,L(s-1)+j} \).

It is noted that the coefficients \( \beta_{ij}^s \) in the estimator may have negative values. Consequently, after finitely many updates, the estimate of density at few values of \( y \in Y_0 \) can be negative. Thus, for practical usage, only the positive-part is considered and the estimator could be modified as:

\[ p(Y_0 \mid x = i) = \eta \left\| \sum_{s=1}^{M} \sum_{j=1}^{L} \beta_{ij}^s p(Y_0 \mid \hat{x}_s = j) \right\|^+, \quad (7.9) \]

where \( \eta \) is a normalization constant and the notation \( | \cdot |^+ = \max(0, \cdot) \).

7.4 Results and Discussion

In this section, the proposed sequential density estimator is validated by using a simulation example. In the example, we consider a scenario in which two existing sensors with known models use a maximum likelihood classifier to generate state estimates. These estimates serve as noisy training labels for a new sensor with a different and unknown sensor model. Referring to Section 7.1, the density estimator for intelligent student is compared with that for naïve student, which assumes that the estimated label is the true label, and also with that for blessed student, which obtains the true label from an oracle (see Remark 7.2). The details of the simulation and the results are presented ahead in this section.

A binary classification problem is considered, i.e., \( \mathcal{X} = \{0, 1\} \) and \( L = 2 \), with two instructors, i.e., \( \mathcal{S} = \{1, 2\} \) and \( M = 2 \). The sensor models in the simulation are chosen to be independent Gaussian distributions. Specifically,
Figure 7.3: Comparison of sequential updates of measurement densities for different approaches

\[ p(Y_i|X) \sim \mathcal{N}(\mu_i^X, \Sigma^i_X), \]
\[ \text{where } \Sigma^i_X = 1 \text{ for all } i \in \{0, 1, 2\}, \mu^1_0 = \mu^2_0 = -1, \mu^0_0 = -3, \]
\[ \mu^1_1 = \mu^2_1 = 1, \text{ and } \mu^0_1 = 3. \]

Note that the unknown model \( p(Y_0|X) \) is less overlapping compared to those of the instructors \( p(Y_1|X) \) and \( p(Y_2|X) \). The state estimates are generated by the instructor \( s \in S \) by using observation \( y_s \in \mathcal{Y}_s \) and the maximum likelihood classifier is as follows:

\[ \hat{x}_s(y_s) = \arg \max_{x \in \mathcal{X}} p(y_s|x). \]

The normalized confusion matrix for the instructor’s classifier, i.e., \( p(\hat{X}_s|X) \), can be computed either by sampling or by using the known cumulative density function. The results of the sequential estimator for first 1000 updates are shown in
Figure 7.4: Comparison of error in density estimation using different approaches

Figure 7.3. Note that, for the na"ive student (see Figure 7.3a), the label noise leads to an incorrect measurement model with two visible components, whereas the intelligent student (see Figure 7.3b) removes the spurious components after using more observations to be closer to the blessed student (see Figure 7.3c). The esti-
mated model after 1000 updates is also shown in Figure 7.4a. Since the true model is known for the new sensor, we compare the estimated model with the true model by computing the Kullback-Liebler (K-L) divergence, as shown in Figure 7.4b. The naïve student converges to a relatively poor model, whereas the proposed approach keeps improving the estimate as more samples become available; however, the improvement is less significant compared to learning from an oracle as expected. The estimate from two instructors is only slightly better than that from a single instructor, because the proposed method computes a weighted average of the estimates from each instructor and the weights depend on the classification performance of the individual instructor. It is noted that, in this simulation example, a noisy sensor with overlapping classes has been used to successfully train a sensor with relatively less overlapping classes. Since the generative model for data in each class is obtained sequentially, the estimate of the maximum likelihood classifier can also be obtained after each update. In this way, the proposed method has been validated to sequentially update the density estimate of measurement model of a sensor using a streaming sequence of labels from the instructors in the network along with the sensor’s own measurements.

7.5 Conclusion

This work presents a new perspective of learning from other sensors in a network and it is shown that existing sensors can teach new sensors. Thus, this chapter provides a concrete solution to the problem of learning from other sensors, which was posed in Chapter 1. The proposed framework has been developed by using recursive density estimation and probabilistic graphical models to obtain a sequential estimator for measurement models. The developed estimator uses estimated labels from existing sensors and their state-dependent statistics of label noise to obtain a generative model of true observation density. The estimation error bound is obtained by using results from recursive kernel density estimation [130].

Currently, the proposed framework fuses the densities from each instructor considering only their classification accuracy and does not consider the number labels contributed by each instructor, which is expected to affect the uncertainty in the density estimate. The framework will be extended to explicitly consider the num-
ber of labels contributed from each instructor and thus, capture the uncertainty in the estimation more accurately.

In order to use the results from [130, 64, 63], this work has assumed that the sequence of measurements conditioned on state is IID; however, consistency and asymptotic normality properties of recursive kernel estimators has also been shown under certain dependency conditions [94]. Thus, the relaxation of IID assumption will also be explored in future.
8.1 Introduction

A measurement model is a model which captures the relation between state and observations including any uncertainty due to contextual effects. A generative model for measurements is the conditional density function, which assigns a probability measure over the space of observations for each value of state. As noted in Chapter 1, the relationship between state and observations as well as the structure of uncertainty model is restricted in several common parametric approaches. Unlike the parametric approaches that need the user to choose the type of model to fit to the data, this work focuses on the use of nonparametric approaches, which can choose the model complexity on their own and do not need any prior information from the user regarding the underlying model shape or statistics of the measurement noise. Likelihood function estimation using nonparametric density estimation will be explored in this work.

The measurement model in this work is a likelihood function $p(Y|X)$, where random variable $Y$ is a feature or an observation in the measurement space $Y$ and random variable $X$ is a state in state space $X$. The nonparametric regression
techniques, such as, k-NN [114], support vector regression [108], and Gaussian process regression [85] have been previously used for function estimation. However, they assume Gaussian uncertainty for each state with same covariance. Typically, a function $f_\mu(x)$ is trained to estimate the expected value of measurement for a given state, i.e., $E(Y|x) \approx f_\mu(x)$, then assuming state-independent noise covariance $\Sigma_Y$, the likelihood function can be obtained using the Gaussian distribution with mean $f_\mu(x)$ and covariance $\Sigma_Y$. Thus, the regression techniques can neither necessarily capture all of the information available in the training set, nor extract different clusters in the data. This work explores the density estimation technique in order to relax the restrictive assumption of known additive Gaussian noise and known structure. The Nadaraya-Watson (N-W) estimator [39, 42, 23], which is a nonparametric estimator for conditional density, can be used for learning the measurement model. The N-W estimator needs all the data to represent the conditional density, hence it is not suitable for near real-time tracking. Thus, there is a need for learning a sparse model to capture the conditional density that can be used in tracking.

To account for this limitation and capture all available information in the training data, this work develops a new density estimation-based technique to directly learn the likelihood function without any assumptions about the data or underlying model. This technique is a new paradigm for learning measurement models, which has no assumption on the structure of relation between the states and measurements, and no assumption on the statistics of noise, i.e., the noise can possibly be multi-modal and state-dependent. Thus, the proposed approach can capture all the uncertainty in the data to give a concise yet accurate measurement model. In this work, two approaches have been developed for learning nonparametric models for state-observation relationship. Of the two approaches (namely, support vector regression (SVR) and density estimation (DE)), the first is an existing regression technique, which has flexibility in learning the relationship, but has a parametric uncertainty model; whereas, the second approach is the newly developed density estimation technique, which has both, flexibility in learning the relationship as well as flexibility to capture the uncertainty model.

The developed measurement models will be used in Bayesian filtering framework for state estimation and tracking of a physical system. Since, the measure-
ment models are not available as functions with additive uncertainty and the conditional density models allow point-wise evaluation of the measurement likelihood, the particle filters [4] are ideal candidate for estimation and tracking in this study. The developed measurement models and tracking procedure will be implemented in simulations and experiments of indoor localization of a human-controlled robotic wheelchair using ambient magnetic fields. Furthermore, the probability distribution from state estimation and tracking is used as output uncertainty to develop an input classification technique. This classifier can identify unsafe inputs coming from a controller to prevent the system from entering unsafe regions of the state space. The notion of safety is incorporated as a chance constraint and numerical integration using particle distributions enables verification of the constraint in near real-time. This new framework has been validated on a simulation testbed of human-controlled wheelchair.

This chapter is organized into 6 more sections. Section 8.2 includes a nonparametric regression-based technique and a novel density estimation-based technique to learn measurement models from data. In Section 8.3, few discrete-time kinematic motion models for a robotic wheelchair are given for 1D and 2D motion, and also the theory and algorithms for particle filters have been explained. Modeling and tracking results from simulations and experiments are shown and discussed in Section 8.4 an Section 8.5, respectively. A Bayesian approach for tracking provides output uncertainty by means of a posterior distribution for the state estimate and this distribution is used to introduce a notion of input classification for safety in Section 8.6. Motivation for input classification and the corresponding simulation results have also been shown in Section 8.6. Finally some concluding remarks and future work has been discussed in Section 8.7.

8.2 Measurement Model Learning Techniques

In this section, a novel nonparametric density estimation-based technique will be developed to learn measurement models from data. An alternative approach using regression will also be shown. The optimization problems for obtaining the models and the corresponding analysis will also be shown in this section ahead.
8.2.1 Regression

Typical measurement models are based on regression, where the expectation $E[Y|X]$ is modeled as a function $f_\mu(x)$ with unknown parameters $\mu$. The restriction on the structure imposed the choice of function can be eliminated by using nonparametric regression. Support-vector regression is one such nonparametric technique. If the measurement space is one-dimensional, i.e., $Y \subset \mathbb{R}$, and a linear model needs to be obtained, then the objective is to learn the function $f(x)$ as:

$$ y = f(x) = \sum_{i=1}^{N} a_i \langle x_i, x \rangle + b, \quad (8.1) $$

where $a_1, a_2, \ldots, a_N, b \in \mathbb{R}$ and $\{x_i, y_i\}$ are the observations in the training sample of size $N$. The differentiating aspect of support vector regression is that there is an error margin $\varepsilon > 0$, such that only the absolute residuals greater than $\varepsilon$ are penalized, as shown in Fig. 8.1. This technique is known as $\varepsilon$-SVR [108] and the observations with positive $\xi_i$ are known as support vectors. An extension of this problem, called $\nu$-SVR [108], was found to be more suitable for learning data-driven models since it also optimizes the error margin, $\epsilon$, by penalizing the size of the margin in the cost function. The $\nu$-SVR problem is formulated as follows:

$$ \text{minimize} \quad \frac{1}{2} \|a\|^2 + C \left( \frac{1}{N_t} \sum_{i=1}^{N_t} (\xi_i^+ + \xi_i^-) + \nu \varepsilon \right) $$

Figure 8.1: Cost from $\varepsilon$-margin loss function (Source: [108])
subject to
\[ \sum_{j=1}^{N_t} a_j \langle x_i, x_j \rangle + b - y_i \leq \varepsilon + \xi_i^- \]
\[ y_i - \sum_{j=1}^{N_t} a_j \langle x_i, x_j \rangle - b \leq \varepsilon + \xi_i^+ \]
\[ \varepsilon \geq 0, \quad \xi_i^+, \xi_i^- \geq 0, \quad \forall i \in \{1, 2, \ldots, N_t\} \]

It is important to point out that \( \langle x_i, x \rangle \) in (8.1) can be replaced with a kernel function \( K(x_i, x) = \phi(x_i)^T \phi(x) \), where the function \( \phi(x) \) maps \( x \) in to a higher dimensional space in which linear regression might be more suitable. Common examples of these kernels are Gaussian kernels, where \( K(x_1, x_2) = \exp(-\gamma \|x_1 - x_2\|^2) \) with \( \gamma > 0 \), and polynomial kernels, where \( K(x_1, x_2) = (x_1^T x_2 + c)^\gamma \). In this work, \( \nu \)-SVR with Gaussian kernels were used to learn the expectation of measurements given the state, i.e., \( E(Y|x) = f(x) \), and a common covariance estimate \( \sigma_y^2 \) was used to obtain the likelihood of measurement \( y \) as follows:

\[ p(y|x) = \frac{1}{\sqrt{2\pi}\sigma_y} \exp \left( -\frac{(y - f(x))^2}{2\sigma_y^2} \right). \]  

\[ (8.2) \]

8.2.2 Nonparametric Density Estimation

The relation between joint and conditional density suggests a way to obtain the likelihood function \( p(Y|X) \) as the ratio of joint density \( p(X,Y) \) and marginal density \( p(X) \). In other words, the relation \( p(X,Y) = \frac{p(X,Y)}{p(X)} \) can be used to obtain the desired model. However, marginalization of any general joint density to obtain \( p(X) \) can be a computationally expensive operation, due to integration over the full measurement space. On the other hand, independently estimating the densities to a high degree of accuracy and then taking the ratio of joint density and marginal density cannot guarantee that the likelihood estimate is bounded. Therefore, an alternative approach was developed using a special type of kernel to address this problem. The desired kernel function \( K: (\mathcal{X} \times \mathcal{Y})^2 \rightarrow \mathbb{R} \) is of the form

\[ K\left([x_1, y_1]^T, [x_2, y_2]^T\right) = K_x(x_1, x_2)K_y(y_1, y_2), \]  

\[ (8.3) \]
with
\[
\int_{\mathcal{Y}} K_y(y_i, y) \, dy_i = 1, \quad \int_{\mathcal{X}} K_x(x_i, x) \, dx_i = 1, \quad (8.4)
\]
\(\forall y \in \mathcal{Y}, \quad \forall x \in \mathcal{X},\) and
\[
K_y(y_1, y_2) \leq k_y < \infty, \quad \forall y_1, y_2 \in \mathcal{Y}. \quad (8.5)
\]

The continuous kernel functions that satisfy the conditions (8.3), (8.4), and (8.5) are known as admissible kernels. These kernels allow us to derive a sufficient condition for boundedness of the likelihood estimate and enable straightforward marginalization. Hence, once an estimate of the joint density is obtained, the conditional density can be obtained easily.

Joint density estimation problem to obtain \(p(X,Y)\) is same as density estimation to obtain \(p(Z)\), where \(Z = [X,Y]\). The optimization problem P4 or the corresponding linear program P5 can be solved using the training data \(\{(x_i, y_i) : i = 1, 2, \ldots, N\}\) (or \(\{z_i : i = 1, 2, \ldots, N\}\)) with an admissible kernel to obtain the density. The resulting density is given as a mixture model as follows:
\[
p(Z) = \sum_{i=1}^{N} \pi_i K(Z, Z_i), \quad (8.6)
\]
where the solution \(\pi\) is a sparse vector, thus the coefficient of several components is zero. Once this density is obtained, the likelihood function estimate is the ratio of joint density and marginal density. A sufficient condition for boundedness of the likelihood estimate is shown in the following lemma.

**Lemma 8.1 (Boundedness of Likelihood Estimate).** Let \(X\) and \(Y\) be random variables which can take values from a state space \(\mathcal{X} \subset \mathbb{R}^n\) and a measurement space \(\mathcal{Y} \subset \mathbb{R}^m\), respectively. Their joint density estimate \(\hat{p}(X,Y)\) is obtained using the density estimation technique. If the kernel used for density estimation is admissible, then the likelihood estimate is bounded as follows:
\[
\hat{p}(Y|X) = \frac{\hat{p}(X,Y)}{\hat{p}(X)} \leq k_y.
\]
Proof. The joint density is approximated as a mixture density as shown in (8.6), then using (8.3)

\[ \hat{p}(X, Y) = \sum_{i=1}^{L} \pi_i K_x(X, X_i)K_y(Y, Y_i) \] (8.7)

is obtained. Marginalization to obtain \( \hat{p}(X) = \int_Y \hat{p}(X, y) dy \) is directly available using (8.4) as follows:

\[ \hat{p}(X) = \sum_{i=1}^{N} \pi_i K_x(X, X_i) \int_{Y} K_y(y, Y_i) dy \] (8.8)

The likelihood estimate is then given by the ratio of (8.7) and (8.8), then the upper bound is obtained using (8.5) as follows:

\[ \hat{p}(Y|X) = \frac{\sum_{i=1}^{N} \pi_i K_x(X, X_i)K_y(Y, Y_i)}{\sum_{i=1}^{N} \pi_i K_x(X, X_i)} \leq \frac{\sum_{i=1}^{N} \pi_i K_x(X, X_i)k_y}{\sum_{i=1}^{N} \pi_i K_x(X, X_i)} \]

\[ \hat{p}(Y|X) \leq k_y. \]

This lemma shows that the likelihood function estimate is guaranteed to be well-defined everywhere. For example, if a Gaussian kernel is used with a diagonal covariance matrix, where \([\sigma_x^2, \sigma_y^2]\) lie along the diagonal,

\[ K(s_1, s_2) = \frac{1}{2\pi \sigma_x \sigma_y} \exp\left(-\frac{(x_1 - x_2)^2}{2\sigma_x^2}\right) \exp\left(-\frac{(y_1 - y_2)^2}{2\sigma_y^2}\right), \] (8.9)

for \( s_1 = [x_1, y_1]^T, s_2 = [x_2, y_2]^T, x_1, x_2 \in \mathcal{X}, \) and \( y_1, y_2 \in \mathcal{Y}. \) This can be verified to be an admissible kernel and it trivially enables the marginalization of the joint density. The likelihood function is then computed as,

\[ p(y|x) = \frac{\left(\sum_{i=1}^{N} a_i \exp\left(-\frac{(x-x_i)^2}{2\sigma_x^2}\right) \exp\left(-\frac{(y-y_i)^2}{2\sigma_y^2}\right)\right)}{\sqrt{2\pi \sigma_y} \left(\sum_{i=1}^{N} a_i \exp\left(-\frac{(x-x_i)^2}{2\sigma_x^2}\right)\right)}, \] (8.10)
which can be verified to be bounded above by \( \frac{1}{\sqrt{2\pi}\sigma_y} \).

### 8.3 Particle Filtering for Tracking

If the system is linear and noise is Gaussian with known parameters, a Kalman filter is known to be an optimal estimator. However, the measurement model, which is available for point-wise evaluation from the learned measurement model, is not available as a function, but as a likelihood model \( p(Y_k|X_k) \). Hence, a recursive Bayesian filter, called the particle filter, was needed for state estimation and localization. The prediction step needs a motion model to propagate the particles ahead in time and the correction step needs a measurement model, as obtained in Section 8.2, to evaluate the likelihood of each particle. The motion models will be shown in this section and then together with measurement model it will be used to obtain a particle filter.

#### 8.3.1 Motion Models

A motion model is a conditional density \( p(X_{k+1} | X_k, u_k) \), which captures the relationship between input, current state, and next state. Usually, a motion model is represented by a deterministic function and additive noise as follows—\( X_{k+1} = h(X_k, u_k) + \eta_k \). Few such models used in this work are as follows:

1. **White Noise Acceleration Model (1D):**

   \[
   D_{k+1} = D_k + TV_k + \frac{1}{2}T^2\eta_k, \\
   V_{k+1} = V_k + T\eta_k,
   \]

   where \( X_k = [D_k, V_k] \) with \( D_k \) as displacement and \( V_k \) as velocity. The acceleration was chosen to a zero-mean Gaussian white noise with variance of \( \sigma_a^2 \), i.e., \( \eta_k \sim \mathcal{N}(0, \sigma_a^2) \). Here \( T \) is the time step for discretization.

2. **Odometry-based Position Model (1D):**

   \[
   X_{k+1} = X_k + u_k + \eta_k^r,
   \]
where \( u_k = \hat{x}_k - \hat{x}_{k-1} \) is the output from odometry for distance traveled and \( \eta_k^x \sim N(0, \sigma_x^2) \).

3. Differential Drive Model (2D):

\[
\begin{align*}
D_{k+1}^1 &= D_k^1 + \left(\frac{u_r^k + u_l^k}{2}\right) T \cos(\theta_k) + \eta_k^1, \\
D_{k+1}^2 &= D_k^2 + \left(\frac{u_r^k + u_l^k}{2}\right) T \sin(\theta_k) + \eta_k^2, \\
\theta_{k+1} &= \theta_k + \left(\frac{u_r^k - u_l^k}{2l}\right) T + \eta_k^3, \\
\end{align*}
\]

where \( X_k = [D_k^1, D_k^2, \theta_k] \) with \( D^1 \) is the displacement along \( x \)-axis and \( D^2 \) is the displacement along \( y \)-axis of a 2D-environment and \( \theta \) is the heading direction of the robot. The input \( u^r \) is the speed of the right wheel and input \( u^l \) is the speed of the left wheel. Here \( T \) is the time step for discretization and \( l \) is the distance between the two wheels of the robot.

Now that we have seen few examples of motion models to be used in the prediction step, we can provide algorithms for particle filtering to track the state of any system.

### 8.3.2 Particle Filters

The underlying goal of a Bayesian approach to state estimation is to calculate the posterior filtered density \( p(X_k|I_k) \), which can be read as the probability of being at state \( X \) at time \( k \) given all information from previous measurements, i.e., \( I_k = (Y_1, Y_2, \ldots, Y_k) \). Consider a simplified localization problem utilizing the magnetic field strength, the state of interest is one-dimensional position along a linear trajectory, denoted as \( X \), and the measurement is the magnitude of the magnetic field strength, denoted as \( Y \).

The particle filter estimates the posterior filtered density by first dispersing a large but finite number of particles (say \( N_p \)) within the domain of interest. Each particle is then assigned a state and an associated weight \( \{x^i_k, w^i_k\} \), where \( i \) represents the \( i \)-th particle and \( k \) is the iteration number representing the discrete time step. By doing so, it is shown in [4] that the posterior filtered density can
Algorithm 2: SIR Particle Filter Algorithm

1. Initialize by distributing $N_p$ particles in the domain

2. for $k = 1 : MAX$

3. for $i = 1 : N_p$

4. $x_i^k = \text{Motion update using kinematic models}$

5. $p(y_k|x_i^k)$ using either (8.2) or (8.10)

6. Update Weights: $w_i^k = w_{i,k-1}p(y_k|x_i^k)$

7. Normalization: $w_i^k = \frac{w_i^k}{\sum_{i=1}^{N_p} w_i^k}$ for all particles

8. $\{x_i^k, w_i^k\}_{i=1}^{N_p} = \text{Resample using Algorithm 3}$

9. Estimation: $\hat{x}_k = g(\{x_i^k\}_{i=1}^{N_p})$

Algorithm 3: Low Variance Resampling Algorithm, as given in [118]

1. Set $i^* = 1$, $c_1 = w_k^1$

2. Sample $u_1$ from $U[0, N_p^{-1}]$

3. for $j = 1 : N_p$

4. $u_j = u_1 + \frac{j-1}{N_p}$

5. while $u_j > c_{i^*}$

6. $i^* = i^* + 1; c_{i^*} = c_{i^* - 1} + w_{i^*}^*$

7. $\tilde{x}_j^k = x_{i^*}^k; \tilde{w}_j^k = \frac{1}{N_p}$

8. return $\{\tilde{x}_j^k, \tilde{w}_j^k\}_{j=1}^{N_p}$

then be approximated as:

$$p(X_k|I_k) \approx \sum_{i=1}^{N_p} w_k^i \delta(X_k - x_k^i), \quad (8.12)$$

where $w_k^i$ is proportional to $u_k^{i-1}p(y_k|x_k^i)$ and $\delta(\cdot)$ is the Dirac measure. By using the state of each particle $x_k^i$ and the measurement $y_k$, the measurement model gives the likelihood value, $p(y_k|x_k^i)$, for each particle which can be used to update the particle weights. This means that in order to converge to a correct solution, there needs to be a technique to eliminate particles with low weight and duplicate particles with high weight, while avoiding degeneracy. Degeneracy arises when only one particle has weight 1 and all other particle weights are 0. Degeneracy is avoided here by using resampling, and the technique chosen for resampling in this work is
the low variance resampling (LVR) algorithm [4, 118]. In a sampling importance resampling (SIR) implementation of a particle filter, resampling occurs at each time step. Consequently, the LVR algorithm was chosen for implementation since it is amongst the most time-efficient methods for resampling with a time complexity of $O(N_p)$, which also guarantees a good coverage of the state space. The particle filtering technique is provided in Algorithm 2 and the resampling algorithm is outlined in Algorithm 3.

The developed measurement learning model will be used with particle filters in simulation and in experiments to illustrate the utility of the proposed model in this chapter ahead.

8.4 Results from Simulation

In this section, the measurement model learned with density estimation will be compared with a reference model in a simulation. An indoor environment is simulated with an ambient magnetic field and a wheelchair robot is also simulated which can be controlled with a joystick. The true motion of the wheelchair for the user inputs given via a joystick is obtained by solving the following differential equations using ode45 function in MATLAB:

$$
\dot{D}_1 = \frac{(w^r + w^l)}{2} \cos(\theta),
$$
$$
\dot{D}_2 = \frac{(w^r + w^l)}{2} \sin(\theta),
$$
$$
\dot{\theta} = \frac{(w^r - w^l)}{2l}.
$$

The first order approximation of these equations is used as the motion model for propagating the particles, as shown in (8.11). This approximation and additive noise in the motion model will lead to diffusion of particles and the measurement update will be used to reduce the uncertainty. The details about the simulation setup, ambient magnetic field, and user data acquisition is given in Section A.3.1 in Appendix A. The results from the proposed approach are compared with a reference approach to learn the measurement model. In the reference model, the measurement model is a Gaussian at each state, where the mean is the true value
according to the simulated ambient magnetic field strength and the variance is considered to be state-independent constant equal to 0.1. The comparison results are given ahead in this section.

In the simulation, the wheelchair robot has 4 magnetometers to measure the strength of the magnetic field. The magnetic field strength at any position in the environment is assigned by a simulated model (See Section A.3.1). The measurement likelihood is computed as the product of the likelihood of the measurement of the four magnetometers. The measurements are only used when the strength of the field is stronger than 2 units. The motion update uses the model in (8.11) with $T = 0.2$ seconds. The user input was recorded from a joystick and the same input was used in the comparison of different measurement models.

The simulated ambient magnetic field is shown as a contour plot in the background of the Figure 8.2. In the Figures 8.2 and 8.3, the true position of the robot is shown with a red cross ($\times$) and the red arrow shows the heading, the estimated position is shown by a black cross, whereas the particles capturing the state distribution are shown with blue arrows. The solid blue line shows the estimated path using particle filters with 100 particles (i.e., $N_p = 100$) and the dashed black line shows the true path. The Figure 8.2 shows the tracking performance at three different time instances in the same run with the reference model. In the proposed model, the kernel bandwidth was chosen to be $[0.1, 0.01, 0.01]$ for the joint density $p(Y, D^1, D^2)$. The resulting density model had 86 components when trained with a sample size of 500. The Figure 8.3 shows the tracking performance at three different instances in the same run with the proposed measurement model. After repeating the simulations under each measurement model for the same recorded sequence of user inputs for 10 times, the men square tracking error and the mean distance error was computed for both models. The results of the error comparison is shown in Table 8.1. It is noted that the mean tracking error of the proposed approach was observed to be less than 5.2 cm, which is comparable to and better than the error of approximately 7.5 cm of the reference model. The videos of the tracking simulations are also available on short url for interested readers.

The simulation results are encouraging and we will see the experimental results ahead in the next section. However, there are a few caveats for tracking with ambient magnetic fields and simulation studies for evaluating tracking errors, these
Figure 8.2: Example of tracking result with the reference model

<table>
<thead>
<tr>
<th>Technique</th>
<th>Root Mean Square Error (in m)</th>
<th>Mean Distance Error (in m)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reference</td>
<td>0.1292</td>
<td>0.0752</td>
</tr>
<tr>
<td>Proposed</td>
<td>0.0624</td>
<td>0.0517</td>
</tr>
</tbody>
</table>

are listed as follows:

- Ambient magnetic fields have smooth profiles. These smooth profiles create contours of magnetic field strength and regions, where identical measurements are observed, will be in close proximity leading to diffusion of particles
Figure 8.3: Example of tracking result with the proposed model along those contours. Thus, magnetometer observations with measurement noise need not have precise localization capabilities.

- The kinematic models used in simulation have process noise, but do not consider wheel slip. This factor can create additional tracking errors.

- In simulation, the joystick input to motor speed transformation is accurately known; however, this transformation needs to be calibrated for real systems. However, it might be dependent on terrain conditions and battery state-of-charge. This additional uncertainty might adversely affect the performance.
Experiments for validation of the proposed measurement modeling and tracking approach was conducted in a controlled setting using a custom-built robotic wheelchair. The objective is to track a human-driven wheelchair on a linear path using ambient magnetic fields, where the true position is obtained by using a motion capture system. The details on the hardware, experiment, and sensor calibration can be found in Section A.3.2 in Appendix A.

Leave-one-out cross-validation was performed for the collected data of the three linear path traversals to compare the different approaches. In this technique, the maps are learned using any two samples and the third sample is used for testing the tracking performance. The regression fit to obtain the expected value of the measurement and the density estimation-based map for a particular case is shown in Fig. 8.4 and Fig. 8.5, respectively.

The benefit of using support vector regression and the proposed density estimation approach is highlighted in Table 8.2, where it is shown that the algorithm itself selects the model complexity by deciding the number of observations to be used to represent the map. The specific count of these non-zero coefficients depends on the kernel parameters and the value of the optimization parameters. However,
Table 8.2: Number of observations used to represent the map

<table>
<thead>
<tr>
<th>No.</th>
<th>Observations</th>
<th>Density estimation</th>
<th>Regression</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>267</td>
<td>102</td>
<td>164</td>
</tr>
<tr>
<td>2</td>
<td>269</td>
<td>92</td>
<td>183</td>
</tr>
<tr>
<td>3</td>
<td>262</td>
<td>145</td>
<td>169</td>
</tr>
</tbody>
</table>

there are analytical results in [108] that do allow some control over the fraction of observations to be used as support vectors for large sample sizes. This control can help to reduce the allotted memory and computation time.

The tracking results for the two mapping techniques using the two kinematic models are shown in Figure 8.6 for average of the three tracks by performing the leave-one-out cross-validation. In order to obtain a metric for quantitative comparison, this work uses two different quantities. $T_{err}$ is the mean absolute tracking error over all the iterations and $L_{err}$ is the mean localization error computed over estimates obtained after initial 100 updates. The localization and tracking results in Fig. 8.7 are for the test case 1. The tracking result for DE map using DIO motion model shows very accurate tracking and the tracks also look reasonable. However, the result of particle filtering based tracking, which can have multi-modal belief structure, can not be correctly visualized by just looking at the mean of all particles. Thus, Fig. 8.8a and Fig. 8.8b shows the location belief for each iteration.
Figure 8.6: Global localization and tracking results for both kinematic models with regression (SVR) and density estimation-based (DE) mapping techniques

Figure 8.7: Trajectory comparison of the both kinematic models with SVR and DE mapping techniques

as an intensity plot for DIO and WNA model, respectively. The true trajectory of the wheelchair as obtained from the motion capture system is also shown in these figures. Considering the analysis presented in this section, the density estimation-based maps performed better than those using regression technique. The best performance of 30.62cm location accuracy was obtained in tracking using DIO
8.6 Safety Constraint for Input Classification

Robotic systems can have perception algorithms using machine learning techniques, which can interpret the surroundings incorrectly leading to wrong input selected for execution. Moreover for human-in-the-loop systems, where the input from the human is given to the machine via a brain-computer interface (BCI) (for patients with paralysis), voice, or eye-ball movement, the input can be wrongly interpreted by the machine, due to imperfect classification techniques. Also the patient using the device might not be giving the correct signals for the intended input, due to lack of attention or deterioration of mental health. In such systems, a wrong input might lead to severe consequences, if it is executed by the machine. Consider an example of a paralysis patient controlling a robotic wheelchair with a BCI using EEG signals as shown schematically in Figure 8.9. The error can be introduced by incorrect command from the human or incorrect interpretation by the machine and the execution of that input might lead to unsafe behavior, such as collision with static obstacles, falling down a staircase, etc. In order to avoid such undesirable outcomes, this work introduces the notion of input classification into two classes—namely, safe and unsafe. The input classified as unsafe will not be executed on the system.
Figure 8.9: Schematic of a patient controlling a wheelchair using a BCI. Few inputs can lead the wheelchair to unsafe states

The state estimation and localization technique developed above in this chapter provides the uncertainty in the state estimate by means of the posterior distribution. This distribution is propagated ahead in time using the estimated input to check if it leads to unsafe states with significantly high probability to identify unsafe inputs. These unsafe inputs are not executed on the system, thus avoiding any severe unintentional consequences. This section ahead formulates the input classification problem and then solves it with the particle distribution from the Bayesian filters discussed in Section 8.3. Some simulation results to illustrate the notion of input classification have also been shown.

8.6.1 Problem Formulation and Solution

The predicted state distribution for the input to be executed is computed as follows:

\[ p(X_{k+1} \mid I_k, u_k = u) = \int_{x \in \mathcal{X}} p(X_{k+1} \mid X_k = x, u_k = u) \, p(X_k = x \mid I_k) \, dx. \]  \hfill (8.14)

If the unsafe region in the state space is known precisely, then the probability that the input \( u \) leads the system into an unsafe state in the next time step is computed as follows:

\[ \alpha_{k+1}(u) = \int_{x \in \mathcal{X}_{\text{unsafe}}} p(X_{k+1} = x \mid I_k, u_k = u) \, dx, \]  \hfill (8.15)
where $\alpha_{k+1}(u)$ is defined to be the un-safety metric. Alternatively, if the knowledge of unsafe region has some uncertainty and $p(S(X) = 0 \mid X = x)$ denotes the probability of a state $x$ being unsafe, then the probability that the input $u$ leads the system into an unsafe state in the next time step is computed as follows:

$$\alpha_{k+1}(u) = \int_{x \in \mathcal{X}} p(S(X_{k+1}) = 0 \mid X_{k+1} = x) \, p(X_{k+1} = x \mid I_k, u_k = u) \, dx. \quad (8.16)$$

If $\alpha_{k+1}(u_k) \geq \varepsilon$, then the input $u_k$ will be classified “unsafe” as there is a significantly high probability of collision (or unsafe behaviors) and it will not be executed on the system.

In order to get $\alpha_{k+1}(u)$, one needs to compute the integration over the complete state space, which can be a computationally expensive operation. However in this work, since particle filters are being used, the prior density $p(X_k \mid I_k)$ is represented by a set of particles and the computation of integration in (8.14) is replaced by propagation of individual particles using the motion model from Section 8.3.1. In (8.16), the metric $\alpha_{k+1}(u)$ can be computed approximately as

$$\alpha_{k+1}(u) \approx \frac{1}{N_p} \sum_{i=1}^{N_p} p(S(x^i_{k+1}) = 0),$$

where $\{x^i_{k+1}, w^i_{k+1} = \frac{1}{N_p}\}$ is the set of particles and their corresponding weights after the motion update step. If the unsafe regions are known precisely then $p(S(x) = 0) = 1$, if $x \in \mathcal{X}_{\text{unsafe}}$. Thus, the un-safety metric can be computed as follows:

$$\alpha_{k+1}(u) \approx \frac{1}{N_p} \sum_{i=1}^{N_p} \mathbb{1}_{\mathcal{X}_{\text{unsafe}}} (x^i_{k+1}), \quad (8.17)$$

where $\mathbb{1}_\mathcal{X}(\cdot)$ is an indicator function for the set $\mathcal{X}$. Hence by using particle filters, the input classification to identify unsafe inputs can be done in near real-time by doing motion update on the particles and then counting the number of particles in the unsafe state set.

A systematic framework for input classification has been developed in this section. Although this concept seems similar to the rules developed by practitioners to be checked during execution for safety or to include fail-safe behaviors for known faults, the input classification concept is not ad-hoc. It is systematically defined as a chance constraint. It leverages the particles used for density representation in state estimation to compute the un-safety metric. Moreover the predicted state
distribution is computed in Bayesian filtering too, here it is computed based on
the intended input and not the executed one, knowing that these inputs will be
exactly the same, only if the input is safe.

8.6.2 Results and Discussion
Considering the motivating scenario shown in Figure 8.9, a simulation setup has
been developed in this work. The details of the setup have been explained in
Section A.3.1. The concept of input classification will be illustrated for an indoor
navigation problem, where a wheelchair is controlled by a human operator using a
joystick. The human gives incorrect input at times which might lead to collisions in
the next time step; however, the system can identify such inputs and successfully
avoid collisions.

Two cases have been demonstrated for input classification and the results have
been shown in Figure 8.10 and Figure 8.11. The figures show the true position
of the robot with a red cross (×) and the red arrow shows the heading, the es-
timated position is shown by a black cross, whereas the particles capturing the
state distribution are shown with blue arrows. The unsafe states are known to be
three rectangular regions in the 2D space represented by red rectangles, thus the
un-safety metric is the same as probability of collision with these unsafe regions.
In case 1, shown in Figure 8.10, as the user gives input to take the differential drive
robot near the obstacle, the system identifies the unsafe input (see Figure 8.10a)
and otherwise it can identify the safe input too (see Figure 8.10b). Similarly, the
system can identify unsafe inputs in case 2 as well, as shown in Figure 8.11. The
estimated collision probability is shown at the bottom of the figures. The threshold
for input classification was chosen to be 0.1. Note that in case 2, the uncertainty in
localization is higher, thus the input taking the robot straight is classified as unsafe
much earlier as compared to that in case 1, where the localization uncertainty is
much smaller. Thus, it can be seen that the input classification takes into account
the state estimation uncertainty too.
Figure 8.10: Demonstration of unsafe/safe input identification: Case 1
8.7 Conclusion and Future Work

A density estimation-based measurement modeling technique is formulated and presented in this chapter to capture all the uncertainty in data. This method in-

Figure 8.11: Demonstration of unsafe/safe input identification: Case 2
troduces a new paradigm in learning data-driven models, which have no restriction on structure of relationship between state and observations or on statistics of noise. The measurement model is represented as a ratio of sparse kernel-based mixture models and it is proven that this ratio is well-defined everywhere. A nonparametric regression-based technique, which also has a sparse structure, is used to compare the performance of the measurement models on localization and tracking tasks.

The particle filtering framework is used with different kinematic motion models to track a robot using the developed measurement model to capture a feature map, such as ambient magnetic field. The model is tested in simulation for 2D localization and tracking of a robotic wheelchair using an array of 4 magnetometers, where the inputs to drive the wheelchair are given using a joystick. The localization using the proposed method is compared with the case in which the magnetic field strength is known, but the magnetometers have noticeable measurement noise. The method was also tested in an experiment with a wheelchair robot for 1D localization and tracking, wherein, it was compared with regression-based model learning technique. The proposed method performed better than the reference in both cases.

The motivation for developing a state estimation and tracking framework with nonparametric measurement model was to eventually develop an input classification framework, which can use the state estimation uncertainty to identify unsafe inputs. This goal was achieved by mathematically formulating the safety condition as a chance constraint. The constraint is checked before executing each input by using a numerical integration procedure using particle approximations to represent densities as in particle filtering. The results on simulation testbed shows that the wheelchair can successfully avoid collision in spite of unsafe inputs from the user. This chapter answers the question posed in Chapter 1 on development of an real-time input classification procedure.
Conclusions and Research Vision

This chapter concludes the thesis with summary of results and suggestions for future work.

9.1 Summary of Results

This dissertation on learning for decision-making has made contributions to uncertainty quantification for learning systems, uncertainty-aware decision-making, data-driven modeling, and safety in control. The work in this dissertation has contributed by addressing the seven important unanswered questions, which are relevant for further development of intelligent physical systems. A summary of the answers to these questions is presented here.

• How to develop a nonparametric model to concisely represent density of a random variable?

Kernel-based density estimation as a convex combination of kernels placed at each observation is used to represent the probability density function (PDF). The coefficients of each kernel are jointly optimized to minimize hinge loss (between empirical cumulative distribution function (CDF) and the estimated CDF) with an adaptive error margin, while constraining the solution to lie on a simplex plane. This optimization gives a nonparametric model of the PDF, which is shown to be accurate and with sparse coefficients. This solution is formulated, analyzed, and illustrated with simulation in Chapter 2.
• **How to tractably fuse information from sensors without assuming conditional independence between measurements given the state?**

A kernel-based mixture model is learned by concatenating data from all sensors for each state using the approach developed in Chapter 2. The latent variable of this mixture model is called context. The kernel used in this mixture model has a “special” structure, due to which each component of the mixture model has conditional independence property given state and context. This conditional independence given the state-context pair is guaranteed and is not assumed. Therefore, Bayes’ rule can be used for exact sensor fusion to obtain posterior density of state-context pair. Marginalization of the context variable gives us the desired posterior density over state. This approach is tractable and accurate for fusing data from multiple sensors. This solution is developed and described in Chapter 3.

• **How to represent information from a multidimensional time-series data using symbolic time-series analysis (STSA)?**

The density estimation technique developed in Chapter 2 gives sparse mixture models from training data. This technique is used to obtain a density estimate for data from a time-series. Using the maximum likelihood classification rule, a partition is created for the multidimensional information space. The observations are assigned to different regions of the partition to obtain a symbol sequence from a time-series data. Markov models capturing the statistical evolution of the symbol sequence are obtained from existing modeling techniques. Thus, a model representing information from a multidimensional time-series data has been obtained. Details of this technique and simulation results are included in Chapter 4.

• **How to dynamically select the most relevant sensor team for state estimation when the environmental conditions affecting those data are unknown?**

A context-aware dynamic sensor selection framework is developed using minimax dynamic programming. Here context-aware measurement models are used for each sensor in the network to incorporate the contextual sensitivity and class separability in the decision-making framework. The difference between the expected cost of decision-making for an existing team and the value function corresponding to adding a new sensor to the team is called the contextual value
of information for the new sensor. This value is then compared with the cost of operating that sensor. The sensor with the highest net value is the most contextually-relevant information source. The formulation, solution approach, and simulation examples for dynamic sensor selection are given in Chapter 6.

- **Can a sensor sequentially learn its correct measurement model from other imperfect sensors?**
  Yes. A sequential learning approach, which incorporates imperfect instructions from multiple instructors in a streaming fashion, to learn the correct measurement model was developed in this work. A probabilistic graphical model (assuming conditional independence given state) is used to obtain a relationship between the true model and the model obtained with labels from the instructor. This relation along with the classical work on recursive kernel density estimation is used to obtain a sequential estimator for the true model. The theoretical analysis on consistency and rate of convergence has also been carried out. The results of analysis and simulations can be found in Chapter 7.

- **How to develop a sequential hypothesis test for streaming data without using the assumption of IID data?**
  Time-series data from physical sensors can be modeled as a Markov model using STSA, which can capture the non-IID behavior. A sequential hypothesis test is then developed for data from the Markov model by (1) deriving the likelihood of a model given a symbol sequence; (2) obtaining a sequential update rule for the ratio of model likelihoods; (3) using a log-likelihood ratio in the same setting as SPRT. The threshold selection for SPRT depends on the desired performance for IID sequences and the same result was obtained for non-IID data from Markov models. Hence, the classical SPRT framework was extended to SPRT with Markov models of time-series data. The analysis, simulation, and experimental results are shown in Chapter 5.

- **How to develop an input classification technique for safety in physical systems?**
  The safety condition is given as a chance constraint, where the probability of the system entering a predefined unsafe state set is bounded by a threshold. This constraint is checked before executing each input by using a numerical integration procedure based on a particle approximation to represent prior and predicted
densities (as in particle filtering). Hence, input classification to identify unsafe inputs can be done in near real-time by doing motion update on the particles using the intended input and then counting the number of particles in the unsafe state set. The input is executed on the system only if the safety condition has been satisfied.

9.2 Suggestions for Future Work

Several concluding remarks and suggestions for future work have been given at the end of each chapter. A list of major proposed research directions and potential applications are given in this section.

9.2.1 Research Directions

*Learning context for systems with continuous states:* The technique to learn context from data assumes that we can construct a training set of measurement for each discrete state. Extension of the concept of context learning and context-aware sensor fusion for systems with continuous states will be considered in the future.

*Uncertainty quantification in decision-making with uncertain Models:* The data-driven model obtained by the method developed in this thesis has some uncertainty due to finite sample size and compression to get a sparse mixture model. This uncertainty in learning is not directly used in the decision-making process yet and needs to be incorporated in the future.

*Truncated multi-hypothesis sequential test with streaming data:* Sequential hypothesis tests for binary classification can be extended to multi-hypothesis testing. Although the sequential test is guaranteed to finish in finite time, there is no upper bound. The effect of truncation of the test at some user-defined finite number on the performance guarantee needs to be explored.

*Dynamic sensor selection with medium-sized sensor teams and multiple targets:* Currently the dynamic sensor selection framework uses exact
dynamic programming to compute all value functions. As the network size grows, the backward induction can become difficult to complete and the resulting lookup table might become impossible to store on-board. Approximate dynamic programming techniques and game-theoretic interaction of multiple teams can be explored in the future.

**Coupling of sequential learning with dynamic sensor selection:** The sensor selection framework assumes that all measurement models are known *a priori* and develops a feedback control policy to activate sensors. Activated high-fidelity sensors can give information feedback to train sensors. The policy obtained after sequential updates of the measurement model will evolve. Characterization of this evolution to understand control of learning systems will be addressed in near future.

### 9.2.2 Potential Applications

The applications used in this thesis are target detection/classification in border surveillance, indoor localization of smart wheelchairs with user-assistance for safety during navigation, and detection of combustion instability using streaming data. These applications have been used to illustrate the general applicability of the results developed in this thesis. In this section, the potential applications of the methods developed in this thesis are given below.

- **Safety in control of complex systems:** The input classification framework can be used in practical systems, where the unsafe regions in state space can be predefined, such as robotic manipulators, vehicles in urban environments, etc.

- **Health monitoring of manufacturing systems:** The context-aware measurement modeling can be used to infer the health status of a machine via dynamic sensor selection, sensor fusion, and state estimation.

- **Machine learning for monitoring and cyber-physical security:** The nonparametric measurement model along with a description of nominal and faulty processes can be used to detect any abnormal behavior. If the behavior does not fit into any process descriptions, then it can be inferred as a system failure, a sensor
malfuction, or a malicious attack. The tracking of the system using a discrete process map can leverage work on localization for security and anomaly detection.

- **Automated cross-sensory calibration**: The work on learning from other sensors can be used to automate sensor calibration process. Small, inexpensive, low-fidelity sensors can be temporarily trained by using bigger, expensive, high-fidelity sensors.
Bibliography


Testbeds for Experimental Validation

This appendix includes description of the three different testbeds used in this dissertation.

A.1 Border Control Testbed

This section briefly describes the experimental setup of a border surveillance system for target detection and classification problems. This experimental setup enables us to collect data from several different sensing modalities for border control. This setup was developed at Penn State as there were no existing publicly available datasets for this problem.

An open area near the Penn State test track facility was chosen for setting up the border control testbed. This area is shown in Figure A.1. The schematic layout of the sensor network is shown in Figure A.2a and a photograph of the actual setup in the field is shown in Figure A.2b. The network has 6 unattended ground sensors (UGS) with seismic, acoustic, and infra-red sensors. A photograph of an UGS system is shown in Figure A.3a. The network also had 3 environmental sensors with on-board power and data acquisition to record the ambient conditions and a photograph of this system is shown in Figure A.3b. The power to two UGS systems was distributed by a custom-designed compact unit, as shown in Figure A.4, which also included signal amplifiers, data-acquisition boards, and
DC-DC power converters. The testbed also included cameras for recording video of the different events.

The experiment was conducted on 3 different days and in total 228 runs were recorded. Each observation belonged either to a human walking class or to a human running class. The human target in the experiment walked for 50-60 meters in the sensor field, while the data was recorded from UGS at 1kHz. The objective was to use various sensors to identify the target class—namely, walking or running. The data collected from this testbed has been used in Chapter 3.
A.2 Combustion Instability Detection Problem

In this section, the data collected for detection of combustion instability in a laboratory environment is described briefly. The test apparatus is a swirl-stabilized, lean-premixed, laboratory-scale combustor (see [50] for details). The combustor consists of an inlet section, an injector, a combustion chamber, and an exhaust section. The combustor chamber consists of an optically-accessible quartz section followed by a variable length steel section. Tests were conducted at a nominal
Table A.1: Operating conditions

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Equivalence Ratio</td>
<td>0.525, 0.55, 0.60, 0.65</td>
</tr>
<tr>
<td>Inlet Velocity</td>
<td>25-50 m/s in 5 m/s increments</td>
</tr>
<tr>
<td>Combustor Length</td>
<td>25-59 inch in 1 inch increments</td>
</tr>
</tbody>
</table>

combustor pressure of 1 atmosphere over a range of operating conditions, as listed in Table A.1.

In each test, dynamic pressure and global OH and CH chemiluminescence intensity in the combustion chamber were measured to study the mechanisms of combustion instability. The measurements were made simultaneously at a sampling rate of 8192 Hz (per channel), and data were collected for 8 seconds, which include a total of 65,536 measurements (per channel). A total of 780 cases for pressure data are collected. This pressure time-series dataset is used in Chapter 5 for combustion instability detection.

### A.3 Smart Wheelchair Testbed

This section describes the simulation and the hardware setup for a smart wheelchair. The simulation setup is used for validation of 2D indoor localization, tracking, and safety, whereas the hardware setup is used for 1D state estimation/localization.

#### A.3.1 Simulation Setup

The simulation was implemented on a Windows PC with MATLAB. A $10 \times 10 m^2$ area was simulated to have an ambient magnetic field. The magnetic field was created by randomly placing 15 Gaussian kernels (with $\sigma^2 = 2$) and by randomly assigning weights to each kernel. A visualization of the magnetic field is shown in Figure A.5a. The dynamic model of the wheelchair was represented using a differential drive model:

\[
\dot{x}_1 = \frac{1}{2} (u_1 + u_2) \cos(x_3); \quad \dot{x}_1 = \frac{1}{2} (u_1 + u_2) \sin(x_3); \quad \dot{x}_3 = \frac{(u_1 - u_2)}{2L},
\]
with $L = 0.5m$. The true motion of the system was simulated with ode45 for a time interval of 0.2s for each input. A user input sequence from the joystick was recorded as the wheelchair was moving in the simulated environment. The tracking was performed with $N_p = 100$ particles using a reference model and the proposed approach for the same pre-recorded input sequence. The results of tracking and of input classification have been discussed in Chapter 8.

### A.3.2 Hardware Setup

A robotic wheelchair platform has been developed for this work by modifying a commercial wheelchair, the Jazzy Pride Select 6 (Pride Mobility Products Corp., USA), with sensors, computers, and a power management system to collect measurements of magnetic field strengths, ground truth location data, and odometry information. An image of the wheelchair is presented in Figure A.6.

The electric wheelchair was retrofitted with a unique power system designed for simplicity, safety, and expandability. The robotic wheelchair’s onboard computer is configured with a 16GB DDR3 RAM and a Quad Core i5 2.9GHz processor. The single board computer serves as the key interface between the joystick, sensors, motor controller, and Robot Operating System (ROS) used to control the wheelchair. The wheelchair was designed with two fixed differential-drive wheels.
Figure A.6: The current configuration of the wheelchair. The enlarged image of the magnetometer is credited to [111]

and four off-centered castor wheels. Each drive wheel is controlled with an independent actuator. An HD25 industrial rugged metal absolute optical encoder is externally affixed to each drive wheel and is used to collect the wheelchair’s odometry information.

**Magnetometer Calibration and Characterization:** An HMC5883L three axis magnetometer was chosen to measure the magnetic field strength. It contains a 12-bit analog to digital converter (ADC) that enables 0.2µT resolution in ±800µT magnetic fields. Calibration was performed using a Pasco Model CI-6520A magnetic field sensor and a Helmholtz coil system, a device capable of producing a nearly uniform magnetic field (range ±1mT; resolution 5µT; accuracy 100µT, 10% of reading).

The sensor characteristics were found by collecting magnetometer data while
the wheelchair’s position remained static. This was accomplished by placing the wheelchair on a platform such that the drive wheels were lifted above the ground but were still able to spin freely. Measurements could then be performed with and without the motors running while the wheelchair was stationary. The experiment for characterization was conducted by running the motors continuously at maximum speed (approximately 2m/s) for five-minute intervals while collecting magnetometer data. Similarly, data was collected for five-minute intervals while the motors were turned off. This test was repeated five times each, with motors on and off, respectively, for consistency. The magnetometer noise for each case is depicted in Fig. A.8. The presence of a systematic bias of 4\(\mu\)T due to running the motors at full speed can also be seen in Fig. A.8. The collected magnetic field data was then analyzed using several techniques to test the hypothesis of Gaussian distribution for the sensor noise. A chi-square goodness-of-fit test [82] accepts the hypothesis of normality. Mardia’s test, which was formulated in [62], also agrees with the chi-square test for normality, as demonstrated by the points in the Q-Q plot falling along the 45\(^\circ\) line in Fig. A.9. The noise standard deviation was estimated to be 0.163\(\mu\)T.

**Data Collection:** A single HMC5883L magnetometer [111] was attached to
the back of the wheelchair via a rigid aluminum bar at a height of approximately 0.05m above the ground. The rear of the wheelchair was chosen in order to ensure the magnetometer was as far away as possible from any magnetic interferences caused by the wheelchair (the DC motors in particular). Throughout the testing, the magnetic field strength was sampled at 10 Hz and stored using ROS. The magnitude of magnetic field strength was then calculated and used to create the maps. Odometry-based measurements, as well as the Vicon system-based ground truth position and orientation information, were also collected during testing.

The data collected using the magnetometers mounted on the smart wheelchair was used for indoor localization in Chapter 8.
Figure A.10: The three-meter linear path of the wheelchair and Vicon motion capture camera system
Symbolic Time-series Analysis

Symbolic time-series analysis (STSA) [10, 22] partitions the measurement space of a dynamical system, where the partitioning is a mapping from a continuous space of measurements to a discrete space of symbols. In machine learning literature, discretization is generally studied as a feature extraction technique. In dynamical system literature, the partitioning or discretization is characterized by the extent to which a dynamical system can be represented by a symbolic one. The data is symbolized based on the choice of a partitioning technique and then, the dynamics of the discrete process is studied. Upon symbolization of statistically stationary (or quasi-stationary) time-series data, the resulting symbol sequences are converted to probabilistic finite state automata (PFSA) for information compression. While the details are reported in [91, 72], the essential information is presented below.

**Definition B.1.** (DFSA) A deterministic finite state automaton (DFSA) is a 3-tuple \( G = (\Sigma, Q, \delta) \) where:

- \( \Sigma \) is a non-empty finite set, called the alphabet, with cardinality \( |\Sigma| \);
- \( Q \) is a non-empty finite set, called the set of states, with cardinality \( |Q| \);
- \( \delta : Q \times \Sigma \rightarrow Q \) is the state transition map.

Note that \( \Sigma^* \) is a countable collection of all finite-length strings with symbols from the alphabet \( \Sigma \) and the (zero-length) empty string \( \varepsilon \).
Definition B.2. (PFSA) A probabilistic finite state automaton (PFSA) is constructed upon a DFSA $\mathcal{G} = (\Sigma, Q, \delta)$ as a pair $\mathcal{M} = (\mathcal{G}, M)$, that is, the PFSA $\mathcal{M}$ is a 4-tuple $\mathcal{M} = (\Sigma, Q, \delta, M)$, where:

- $\Sigma, Q,$ and $\delta$ are the same as in Defn. B.1;
- $M : Q \times \Sigma \to [0,1]$ is the morph function that satisfies the condition $\sum_{s \in \Sigma} m(q, s) = 1 \ \forall q \in Q$. In matrix form $M$ is called the morph matrix and its entries $m_{ij}$ denotes the probability of emitting a symbol $s_j \in \Sigma$ from the state $q_i \in Q$.

A PFSA is viewed as a generative machine capable of probabilistic generation of a symbol string through state transitions. PFSA has found many uses ranging from pattern recognition, machine learning to computational linguistics [123]. For symbolic analysis of time-series data, a class of PFSAs called the $D$-Markov machine have been proposed in [91] as a suboptimal but computationally efficient approach for encoding the dynamics of symbol sequences as a finite state machine. The main assumption, which is the reason for sub-optimality, is that the symbolic process can be approximated as a $D^{th}$ order Markov process. The assumption of finite memory (or order) is reasonable for stable and controlled engineering systems that tend to eventually forget their initial conditions. The states of this PFSA are words over $\Sigma$ of length $D$ (or less); and state transitions are described by a sliding block code of memory $D$ and anticipation length of one [59].

Definition B.3. ($D$-Markov Machine [91, 72]) A $D$-Markov machine is a statistically stationary stochastic process $S = \ldots s_{-1}s_0s_1 \ldots$, where the probability of occurrence of a new symbol depends only on the last $D$ symbols, that is,

$$P(s_n \mid \ldots s_{n-D} \ldots s_{n-1}) = P(s_n \mid s_{n-D} \ldots s_{n-1}) \quad (B.1)$$

where $D$ is called the depth of the Markov machine.

A $D$-Markov machine is thus a $D^{th}$-order Markov approximation of the discrete symbolic process. This machine can be constructed from a symbol sequence by frequency counting to estimate the probabilities of each transition. Let $N_{ij}$ denote the number of times that a symbol $s_j$ is generated from the state $q_i$ in a given
symbol sequence. The maximum a posteriori probability (MAP) estimate of the morph map for the PFSA $M$ is computed by frequency counting as:

$$m(q_i, s_j) \triangleq \frac{C_{ij}}{\sum_{\ell} C_{i\ell}} = \frac{1 + N_{ij}}{|\Sigma| + \sum_{\ell} N_{i\ell}}$$

(B.2)

The rationale for initializing each element of the count matrix $C$ to 1 is that if no event is generated at a state $q \in Q$, then there should be no preference to any particular symbol and it is logical to have $m(q, s) = \frac{1}{|\Sigma|}$ $\forall s \in \Sigma$, i.e., the uniform distribution of symbol emission from the state $q$. The above procedure guarantees that the PFSA, constructed from a (finite-length) symbol string, must have an (element-wise) strictly positive morph map $M$.

Note that in the case of finite-order, finite-state Markov chains the conditional symbol emission probabilities and the initial state summarizes all of the relevant information supplied by any sample [53]. Under stationarity assumptions, the initial state becomes unnecessary and thus, the sufficient statistic is provided by the emission probabilities given by the morph matrix.
Vita

Nurali Virani

Nurali Virani was a graduate research assistant and a Ph.D. candidate in the Department of Mechanical Engineering at the Pennsylvania State University till February 2017. He completed his M.S. in Mechanical Engineering (2015) and a M.S. in Electrical Engineering (2016) from Penn State. He also has a M.Tech. in Industrial Engineering and a B.Tech. in Manufacturing Science (2011) from Indian Institute of Technology (IIT) Kharagpur. He was awarded the Institute Silver Medal and the G. S. Sanyal Cup from IIT Kharagpur in 2011. At Penn State, Nurali was jointly advised by Dr. Asok Ray and Dr. Sean Brennan. His research interests are broadly in the area of learning, perception, planning, and control for dynamic data-driven systems. His areas of research include: learning measurement models from data; context-aware decision-making; fault diagnosis and anomaly detection in complex systems; multi-modal sensor fusion for target detection, classification, and tracking in surveillance systems; indoor localization and mapping; and robust motion planning. He has 12 peer-reviewed publications and 3 patents as contributions to the scientific literature. He has won a Best Student Paper award in KDD (workshop on machine learning for prognostics and health monitoring) and a Best Presentation award in ACC conference. He was a member of the Networked Robotics Systems Laboratory (NRSL), the Intelligent Vehicles and Systems Group (IVSG), and the Multi-agent Networks Laboratory at Penn State.