ADAPTIVE PATTERN RECOGNITION IN COMPLEX SYSTEMS
VIA SYMBOLIC TIME SERIES ANALYSIS

A Dissertation in
Mechanical Engineering
by
Yue Li

© 2016 Yue Li

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Doctor of Philosophy

August 2016
The dissertation of Yue Li was reviewed and approved* by the following:

Asok Ray
Distinguished Professor of Mechanical Engineering
Dissertation Advisor, Chair of Committee

Christopher D. Rahn
Professor of Mechanical Engineering
Associate Head of Administration, Department of Mechanical and Nuclear Engineering

Thomas A. Wettergren
Adjunct Professor of Mechanical Engineering
Senior Research Scientist, Naval Undersea Warfare Center

Hosam Fathy
Associate Professor of Mechanical Engineering

Minghui Zhu
Assistant Professor of Electrical Engineering

Shashi Phoha
Graduate Faculty of Electrical Engineering
Division Director of Information Sciences and Technology, Applied Research Laboratory

Karen A. Thole
Professor of Mechanical Engineering
Department Head of Mechanical and Nuclear Engineering

*Signatures are on file in the Graduate School.
Abstract

This dissertation represents a framework for adaptive information extraction from complex systems via symbolic time series analysis (STSA). The key idea for STSA is to convert original time series of digital signals into a sequence of (spatially discrete) symbols from which embedded dynamic information can be extracted and analysed. The main challenges are: 1) selection of symbol alphabet size; 2) identification of partitioning locations in signal space of time series; and 3) dynamic modeling of symbol sequences to extract embedded temporal patterns. In this context, probabilistic deterministic finite-state automata (PDFA), a special class of Hidden Markov models (HMMs), are used to learn the temporal patterns embedded in symbol sequences. A novel unsupervised symbolization algorithm is developed to construct PDFA models from time series by maximizing the mutual information measurement between symbol set and state set. It is demonstrated that the proposed approach can capture the underlying dynamic information of time series more effectively than existing unsupervised symbolization methods such as equal frequency partition and equal width partition. In order to evaluate the information dependence and causality between two time series, a special class of PDFA, called \( \times D \)-Markov (pronounced as cross \( D \)-Markov) machines, are adopted in this dissertation. To quantify the information flow from one time series to the other, an information-theoretical measurement derived from the concept of transfer entropy is introduced.

In this dissertation, the proposed STSA approaches are adapted and applied to three different types of complex systems for different purposes. The first is state estimation and parameter identification for SISO systems via symbolic dynamic modeling of synchronized input-output time series. By considering input-output jointly instead of system output alone, the proposed data-driven approach has ability to provide robust results under fluctuating or varying input patterns. To overcome the deficiencies of solely model-based filtering and those of solely data-driven approaches, the estimation framework is constructed based on a model-based re-
cursive Bayesian estimator combined with a data-driven measurement model. The second type of system under investigation are large scale sensor networks. The problem is to identify and locate useful information under a dynamic ambient noise from the environment which impacts each individual sensor differently. In general, the changes in a dynamic environment may significantly affect the performance of pattern recognition due to limited training scenarios and the assumptions made on signal behavior under a static environment. Both symbol level and feature-level fusion schemes are proposed to evaluate the information content of sensor nodes. The third research topic is dimensionality reduction of high dimensional data (e.g., video) for information extraction. The main challenge is to develop feature extraction and information compression algorithms with low computational complexity such that they can be applied to real-time analysis of video (i.e., a sequence of image frames) captured by a high-speed camera. In the proposed method, the sequence of images is converted to a sequence of symbols where embedded dynamic characteristics of the physical process are preserved.
# Table of Contents

List of Figures ix  
List of Tables xiii  
Acknowledgments xiv  

Chapter 1
Introduction 1  
1.1 Main Tasks in Time Series Symbolization 1  
1.2 Information Extraction from Complex Systems 3  
1.2.1 Battery System 3  
1.2.1.1 Battery State-of-Health 3  
1.2.1.2 Battery State-of-Charge 4  
1.2.2 Combustion Flame Image Processing 5  
1.2.3 Sensor Networks under Dynamic Environment 6  
1.3 Organization of the Dissertation 7  

Chapter 2
Mathematic Preliminaries and Previous Works 10  
2.1 Symbolic Dynamics 10  
2.2 Probabilistic Deterministic Finite Automata (PDFA) 13  
2.3 Hidden Markov Models 15  
2.4 Information Theory 19  
2.5 Existing framework for PDFA Construction 21  
2.5.1 Selection of Alphabetize 21  
2.5.2 Determination of Partitioning locations 22  
2.6 Symbolic Dynamic Filtering (SDF) 24
List of Figures

2.1 Shifting Block Code .................................................. 12
2.2 Topological conjugacy from X to Y ................................. 13
2.3 Relationship among M-step shifts of finite type (M-SFT) in symbolic dynamics, probabilistic deterministic finite-state automata (PDFA) in the theory of computation, and hidden Markov models (HMM) in stochastic processes. ........................................ 17
2.4 ×D-Markov (pronounced as cross D-Markov) modeling and feature extraction ................................................. 18
2.5 An example of time series symbolization: First, a partition is applied to the original time series; then, each signal is mapped to the symbol which is associated to the region; finally, the state is generated by combine symbol blocks. ........................................ 21
2.6 Maximum Shannon entropy and entropy excess at different alphabet sizes. .................................................. 22
3.1 Procedure for wavelet-based segmentation ....................... 33
3.2 Construction of finite state automata (FSA) from time series .... 35
3.3 The flow chart for the proposed method of feature extraction . 36
3.4 Profiles of input and output data from battery experiments .... 37
3.5 Degradation of battery capacity ................................. 38
3.6 Wavelet-based segmentation for normalized input-output. .... 38
3.7 Normalized input-output pairs of synchronized data for one duty cycle. .................................................. 39
3.8 Profiles of input current and output voltage for one duty period that has 65 small input cycles. Plots (c), (d), (g), and (h) depict the variations in the (normalized) input and output profiles for cycles within one duty period. .................................................. 40
3.9 Maximum entropy partitioning (MEP) in the input-output space corresponding to the alphabet size $|\Sigma| = 3$ for both dimensions: The first partition is in one dimension, followed by the partition in second dimension for each segment of the first dimension. (a) First x-axis and then y-axis; (b) first y-axis, then x-axis; (c) first magnitude and then phase, and (d) first phase and then magnitude.

3.10 Results of SOH estimation with alphabet size $|\Sigma| = 4$ for both input and output.

4.1 Schematic description of the hybrid recursive Bayesian filtering SOC estimation algorithm.

4.2 Typical profiles of long-term data: input current, output voltage, and battery SOC.

4.3 Examples of dynamic zone and steady-state zone in the experimental data.

4.4 Examples of recursive SOC estimation for different alphabet size $|\Sigma|$.

4.5 Root-mean-square (RMS) error of SOC estimation for different alphabet size $|\Sigma|$.

5.1 Proposed approach for high-speed video symbolization.

5.2 Flow chart of the proposed method training and testing phases.

5.3 Schematic view of the combustor apparatus (fluid flow from left to right).

5.4 Testing examples for different combustor length at equivalent ratio of 0.55 with Fractal dimension measurement.

5.5 Posterior probabilities as functions of test data length at different equivalence ratios and combustor lengths.

5.6 Average misclassification rate for different combustor lengths.

5.7 Average misclassification rate for different Equivalence ratios.

6.1 Feature-space illustration in a dynamic background environment.

6.2 Experimental setup for target detection in a dynamic environment.

6.3 Typical profiles for collected data.

6.4 Symbolization procedure for multi-dimensional time series.

6.5 Distribution of cluster distances for environment-only ($H_0$) and target-present ($H_1$) scenarios in the feature space with cosine distance (similarity).

6.6 Receiver operating characteristic (ROC) curves for feature-level target detection.
7.1 Flow chart to generate information contribution measurement for each sensor in the network ........................................ 87
7.2 Example: (a) and (b) present the partition of sensor time series before/after adding noise with alphabet size of 4. Time series of sensors are marked in blue. The line for symbolization is marked in red. (c) and (d) depict the distance among PDFA features using cosine distance measurement ........................................ 88
7.3 Average conditional entropies of Markov machines for both "Target Present" and "Target Absent" cases ................................. 90
7.4 Average information causality measures for both target present and target absent cases ...................................................... 90
7.5 Average PCA coefficients both target present and target absent cases ................................................................. 91
7.6 ROC curve for target detection problem under different choice of alphabet size ................................................. 92

8.1 An example of state splitting with alphabet $\Sigma = \{\sigma_1, \sigma_2, \sigma_3\}$. The state $q$ is first split as $\{\sigma_1 q, \sigma_2 q, \sigma_3 q\}$. The new state $\sigma_2 q$ is not further split because its occurrence probability $P(\sigma_2 q)$ is less than the threshold $\varepsilon_p$. The new state $\sigma_3 q$ is not further split because the sum of entropy rate of generated states $\{\sigma_1 \sigma_3 q, \sigma_2 \sigma_3 q, \sigma_3 \sigma_3 q\}$ is not significantly reduced compared to that of state $\sigma_3 q$. The state $\sigma_1 q$ is split as both criteria are satisfied. States after splitting are $\{\sigma_1 \sigma_1 q, \sigma_2 \sigma_1 q, \sigma_3 \sigma_1 q, \sigma_2 q, \sigma_3 q\}$. ......................................................... 98
8.2 Example of simulation validation via time series generated from a PDFA model ........................................... 99
8.3 The 29 candidate partitioning locations for the simulation time series generated from PDFA .................................................. 100
8.4 Mutual information for each candidate partitioning locations for each iteration at different depth $D$ of modeling .............................. 101
8.5 Examples: Three different types of partitions for sensor time series: (a) Partitioning for the proposed unsupervised time series symbolization algorithm which is compared with two other unsupervised symbolization algorithms: (b) Uniform partitioning and (c) Maximum entropy partitioning. Results in plates (a), (b), and (c) are for alphabet size $|\Sigma| = 4$, where the partitioning line segments for symbolization are shown by (red) dashed horizontal lines and time series of sensors are marked in (blue) solid-line profiles. Plates (d), (e), and (f) depict the Hamming distance among generated symbol strings the respective binary trees. The distance between two symbol strings is measured by the Hamming distance defined in Eq. (2.21). The distance between two clusters of symbol strings is measured by the cluster distance defined in Eq. (2.25). 

8.6 Average maximum mutual information over 9 sensors for different alphabet sizes. The ratio between the values of alphabet size $|\Sigma| = 4, 5$ is lower than the predefined threshold, therefore $|\Sigma| = 4$ is chosen as the selected alphabet size, which is marked in red.

B.1 Examples of binary tree structure formed by multiple sensors.
List of Tables

3.1 Coefficient of Determination (COD) for SOH estimation at alphabet size $|\Sigma| = |\Sigma_1| \times |\Sigma_2|$ ................................................. 41
3.2 Execution time and required memory for different analyzed data length ................................................................. 42

6.1 Average confusion matrices of cross validation for decision-level target detection .................................................. 75
6.2 Average confusion matrices of cross validation for data-level target detection with # of clusters $N_C = 3$ .................. 77
6.3 Average confusion matrices for feature-level fusion target detection I ................................................................. 79
6.4 Average confusion matrices for feature-level fusion target detection II ................................................................. 79
6.5 Cluster purity for target-present scenarios ................................................. 81

7.1 Average feature divergence ratio between Target Present and Target Absent cases under different alphabet sizes .......... 89

8.1 The occurrence probabilities of State and Symbol emission for the simulation time series ........................................... 101
Acknowledgments

This thesis would not have been possible without my advisor Dr. Asok Ray and his guidance, encouragement, and mentorship he kindly offered for the last six years. I also would like to express my sincere gratitude to Dr. Christopher D. Rahn and Dr. Thomas A. Wettergren whose guidance and enlightenment have been invaluable to my dissertation. Moreover, I am deeply grateful to Dr. Hosam Fathy, Dr. Minghui Zhu, and Dr. Shashi Phoha for serving my committee and providing your valuable suggestions and time to improve the quality of my thesis in a unique way.

I would like to thank Dr. Zheng Shen and Dr. Kushal Mukherjee for all the insightful discussions and valuable suggestions that moved forward my research. I would especially like to thank my colleague Mr. Devesh Kumar Jha for sharing vision and knowledge in our academic projects. I would like to thank Ms. Pritthi Chattopadhyay, Mr. Michael Hauser, Mr. Jihang Li and Mr. Sihan Xiong for their intelligent contributions in our cooperated publications. I would also like to thank Mr. Yiwei Fu, Mr. Nurali Virani, Dr. Eric Keller, and all other members of Dr. Ray’s group for their professional association.

I would like to thank my parents and my wife for their love and support along the way. I am deeply indebted to them for their dedication, encouragement and inspiration without expecting anything in return.

Lastly, I would like to gratefully acknowledge the financial support I received throughout these years: the U.S. Office of Naval Research under Grant N00014-14-1-0545, the U.S. Air Force Office of Scientific Research under Grant FA9550-15-1-0400, and the U.S. Army Research Laboratory under Grant W911NF-14-2-0068.
Dedication

This thesis is dedicated to my beloved grandmother who always believed in me.
Chapter 1

Introduction

Symbolic time series analysis, as a data interpretation technique, has been studied in the open literature for over a century. In 1898, Hadmard [1] represented geodesic flows on surfaces of negative curvature using a symbolic description. In 1938, Morse and Hedlund [2] extended Hadmard’s work [1] by studying the symbolic-based dynamical features for periodic orbits in classical systems, where the term symbolic dynamics was introduced for the first time. Later on, the concepts of symbolic dynamics-based analysis have been applied to many other research topics. Examples are: communication encoding in information theory [3], Markov decision processes in discrete dynamical systems [4], information extraction in supervised data-mining [5].

Information loss is often a concern in symbolization of a continuous time series that is discretized into the corresponding symbol series. This issue has been addressed by Beim Graben [6] in the sense that symbolization can directly improve signal-to-noise ratio (SNR) in noise-contaminated signals. In addition, symbolization makes digital communication and numerical computation more efficient and effective as compared to similar operations with continuous-valued data. Discussions on advantages of working with symbolized data can be found for different applications [7, 8, 9].

1.1 Main Tasks in Time Series Symbolization

There are two main tasks in time series symbolization [10], namely, (i) definition of symbols, and (ii) construction of words (i.e., symbol blocks) that have relevance for information representation [8, 11]. Symbols replace the data points in the original time series with a pre-defined mapping. The most common approach is to partition the range of time-series data into a finite number of mutually exclusive and exhaustive segments. Each segment is associated with a unique symbol, to which all data points in that segment are mapped accordingly. The choice of total num-
ber of symbols (i.e., alphabet size) may vary from 2 to the number of distinct data point values in the time series (i.e., without any information loss). Although more information would be retained when the alphabet size is increased, the detrimental effects of any measurement noise would likely be more profound [6]. It is obvious that both alphabet size and partition map would influence preservation of the original information in the generated symbol series; therefore, the choice of both the number of symbols and partition locations are crucial for subsequent signal analysis. After symbols are defined, the next task is to determine the (finite-length) symbol blocks (also called words) that represent meaningful temporal patterns. This step is crucial for generating dynamical models from the symbol series that can be used for event prediction [12] as a pattern recognition problem. Construction of words is analogous to time delay embedding in phase space [13], where a combination of words may contain complete dynamical information of the symbol series. The results, generated from word construction, can be represented as a context tree in variable-order Markov model [14], where the symbols in a series with Markov property depend on previous symbol blocks (i.e., words). Naturally, some words may not exist since it does not appear at all for a given symbol series, and some existing words may not have any significant contribution to the overall dynamics. Therefore, instead of investigating all possible words with (or up to) a fixed length, it is desirable to explore the potentially meaningful words with adaptive and individual length increment to achieve variable embedding [12].

To appropriately choose partitions for time series generated by unknown dynamic process with unknown noise level, several ad hoc methods have been reported in the literature. For example, Rajagopalan and Ray [15] proposed two partitioning schemes based on equal interval width and equal visiting frequency between segments respectively for one dimensional data. Ahmed et al. [16] investigated a data discretization technique called Frequency Dynamic Interval Class (FDIC), which is based on the data statistical frequency measure and k-nearest neighbor (kNN) algorithm. To obtain optimal partition locations, the Shannon entropy measure [3] has been used by other researchers to maximize the dynamic information level in the symbol series. For example, Lehrman et al. [17] defined an entropy measure of information contents in the symbol series and maximized it by a optimization process based on the number of symbols and locations of partitions. Morchen et al. [18] proposed an unsupervised discretization algorithm based on the Kullback-Leibler divergence [19] between the marginal and self-transition probability distribution of the symbol series. Other studies have been reported about word construction with variable embedding. For example, Kennel et al. [12], motivated by the minimum description length (MDL) principle [20] and context tree weighting (CTW) algorithm [21], constructed a compact and robust context-tree predictive model via a finite-state automaton with probabilistic emission probabilities. Mukherjee and Ray [11] proposed a state splitting (i.e., increment of word length) and merging (i.e., decrement of word length) algorithm by minimizing the
conditional entropy measures between symbols and potential words in the symbolic series.

1.2 Information Extraction from Complex Systems

1.2.1 Battery System

Definitions of pertinent battery parameter (i.e., state-of-health) and state (i.e., state-of-charge) at a given ambient temperature [22] are introduced and the literature review is provided for the parameter identification and state estimation.

1.2.1.1 Battery State-of-Health

**Definition 1.2.1.** (Battery capacity) The capacity $C(t)$ of a battery at time $t$ is the maximum charge (in units of ampere-hours) that can be discharged from a fully charged condition at a rate $C(t)/30$ (in units of amperes).

**Definition 1.2.2.** (SOH) Let a new battery be put into service at time $t_0$. The state of health $SOH(t)$ of the (possibly used) battery at the current time $t$, where $t \geq t_0$, is defined to be the ratio of the battery capacities at time epochs $t$ and $t_0$, i.e.,

$$SOH(t) = \frac{C(t)}{C(t_0)} \text{ for all } t \geq t_0$$  

(1.1)

**Remark 1.2.1.** It is noted that $SOH \in [0, 1]$ for all time $t \geq t_0$, where $t_0$ is the time of putting a new battery into service. In this context, the battery is assumed to fail to meet the operational requirement, i.e., it needs to be replaced, when the SOH parameter is less than 80%.

The current state-of-the-art for SOH parameter identification can be divided into two broad categories: (a) model-based analysis, and (b) data-driven analysis, both of which need databases of experimental measurements for validation. The model-based analysis can be further divided into two sub-categories: (i) empirical modeling; and (ii) physics-based modeling. Empirical modeling methods (e.g., impedance measurements for SOH) [23][24] often employ dedicated hardware and/or software and require battery cells for testing. Although these empirical methods may provide good estimates in specific cases, they are time-consuming and cost-prohibitive for general applications. On the other hand, physics-based modeling methods have been extensively used for identification of battery parameters; examples are reduced-order system identification [25], linear switch models [26], and Kalman filtering [27]. However, physics-based modeling methods require thorough knowledge of the electrochemical characteristics of the battery cells.
to develop appropriate model structures for parameter identification at different operating points of the (nonlinear) battery dynamics.

Compared to model-based approaches, dynamic data-driven methods do not explicitly rely on dedicated hardware/software and physics-based models of battery dynamics. If comparable and adequate training data are available under different operating conditions, data-driven methods are significantly more efficient than model-based methods in terms of computation execution time and memory requirements. Several data-driven methods have also been reported in literature for battery parameter identification. Nuhic et al. [28] used support vector machines (SVM) to estimate SOH under different environmental and load conditions by processing the training and test data based on load collectives. Lin et al. [29] constructed a probability neural network (PNN) and trained it for SOH identification. He et al. [30] proposed a data-driven method based on dynamic Bayesian networks (DBN), in which a DBN was trained for each class of SOH values in the training data. A forward algorithm was then applied to estimate the SOH in real time. Hu et al. [31] conducted capacity estimation of Lithium-ion batteries by k-nearest neighbor (k-NN) regression [32].

1.2.1.2 Battery State-of-Charge

**Definition 1.2.3.** (DOD and SOC) Let a battery be fully charged at time $T$ (in the slow time-scale) and let $I(\tau)$ be the applied current (in units of amperes) at time $\tau$ (in the fast time-scale). Then, depth of discharge (DOD) and state of charge (SOC) at time $T + \Delta T$ are respectively defined as:

$$DOD(T + \Delta T) = \frac{1}{C(T)} \int_T^{T+\Delta T} I(\tau) \, d\tau, \quad \Delta T \geq 0 \quad (1.2)$$

$$SOC(T + \Delta T) = 1 - DOD(T + \Delta t), \quad \Delta T \geq 0 \quad (1.3)$$

If the battery capacity $C$ is a known parameter that is assumed to be a constant for the purpose of SOC estimation, then DOD depends only on the input current that is also a known variable. Since direct measurements of SOC are beyond the current state of the art in battery technology, the challenge is to develop analytical measurement models of SOC in real time by relying on the time series of input current and output voltage.

To achieve a desired level of estimation accuracy in the presence of battery nonlinear dynamics, several algorithms have been proposed based on the concept of Kalman filtering; examples are: extended Kalman filtering (EKF) [33], adaptive extended Kalman filtering (AEKF) [34, 35, 36], and unscented Kalman filtering (UKF) [37, 38]. However, such techniques may not construct accurate filters for battery state estimation due to the lack of adequate understanding of battery electrochemical dynamics. This shortcoming of model-based filtering could be complemented by dynamic data-driven approaches. The data-driven analysis makes
use of the information derived from time series of the measurement ensemble and
does not apparently require detailed knowledge of battery electrochemistry; there-
fore, data-driven approaches could be more efficient in terms of computational
complexity (e.g., execution time and memory requirements) [39, 40].

With the recent development of sensing techniques and sophisticated testing
platforms, large volume of data with high quality could be made available to cover
wide ranges of operating conditions. This leads to the possibility of formulating
robust and accurate data-driven SOC estimation algorithms that can be applied
to practical battery management systems; for example, k-nearest neighbor (k-
NN) regression [41], support vector machines (SVM) [42], and artificial neural
networks (ANNs) [43]. However, there exists common problems for purely data-
driven methods, namely, over-fitting and under-fitting, which may compromise the
performance of SOC estimation.

To overcome the deficiencies of solely model-based filtering and those of solely
data-driven algorithms, several hybrid methods have been proposed by combin-
ing the merits of both model-based and data-driven approaches. For example,
Charkhgard et al. [44] used neural networking to approximate a state-space model
and combined it with an extended Kalman filter for SOC estimation. Xu et al. [45]
reported a similar combination with a stochastic fuzzy neural network (SFNN)
that has filtering effects on noisy input. He et al. [46] have adopted an unscented
Kalman filter (UKF) to enhance accuracy of nonlinear estimation.

1.2.2 Combustion Flame Image Processing

Much research has been reported in open literature on flame instabilities and lean
blowout from different perspectives. Large eddy simulations (LES) have been
applied for identification of combustion instabilities under variable-length com-
bustors [47] and fluctuating equivalence ratios [48]. However, such high-resolution
models of the physical process require very high computational capabilities that are
not suitable for real-time applications. Multiple types of sensors are also commonly
used for monitoring different characteristics of combustion processes that include
flow velocity [49], pressure fluctuation [50], and optical properties (e.g., chemilumi-
nescence) [51]. Even with high-precision and sensitive sensing instrumentation, the
captured data can only represent the average dynamics in its neighboring area; it is
still very difficult to have global information of the combustion process, especially
for details of the flame patterns.

Image processing techniques have been applied to combustion flame analysis
for over a decade [52]. With the abundance of high-speed digital image data, two
(or three)-dimensional statistical information on flame fluctuations can be cap-
tured and used for identifying the combustion state. Many publications discuss
flame edge detection and modeling for identifying the flame pattern characteris-
tics [53] [54]. For example, Chen et al. [55] applied the pattern recognition tool
of principal component analysis (PCA) \[32\] to extract important trends and variations in the operational video. However, most state-of-the-art image processing algorithms have focused on a single frame of the entire video of the combustion flame; there is little research in the analysis of the dynamic evolution of the flame from a sequence of images.

1.2.3 Sensor Networks under Dynamic Environment

Sensor networks serve the important role of collecting and processing information in a variety of (possibly dynamic) environments, whose structure may be partially or completely unknown \[56\]. Such a network consists of a number of sensor nodes, each having the capability to process the signal that it measured from its own or a neighboring environment. The sensor nodes communicate with each other for information aggregation to enhance the quality of collected information in the network. However, due to the constraints on power, computation, and communication resources \[57\], it becomes a challenge to reduce both installation and operating cost while enhancing the system performance \[58, 59\]. From the perspectives of target detection in a dynamic environment, sensor network design (e.g., sensor placement, sensor selection, and the associated decision criteria) relies on the statistical characteristics of target behavior in the surveillance region. For temporally and spatially distributed events, a potential problem may arise due to large volumes of data sets with spurious background noise; this problem could be partially mitigated by data collection with a sparse set of sensors across the network.

If a large field of passive sensors is deployed, the decision parameters of the individual sensors need to be tuned for accurate detection, classification, and tracking. Most of the work on distributed decision fusion, which has been based on threshold rules, use decision rules such as voting, Bayes criterion, and Neyman-Pearson methods \[60\] by using \textit{a priori} knowledge on the probability of target presence \[61, 62, 63\].

The above methods rely on available sensor models that often do not include the effects of background noise in the operating environment. However, it is very likely that a dynamic environment would interfere with the sensor data that correspond to the events of interest (e.g., a mobile target); hence, the decision rules based on thresholding of signal variables may not hold in a different background environment. A more efficient way of event (e.g., target) detection is usage of the pattern matching concept that may require feature extraction, followed by classification of sensor data for event detection. Duarte and Hu \[64\] evaluated several classification algorithms for vehicle detection & tracking, and other types of implementation can be found in \[65, 66, 67, 68, 69\]. However, most of the classification-based detection schemes require significant computational resources for signal conditioning and feature extraction; these techniques may also require scenario-specific training to initialize the classifier. Furthermore, changes in the boundaries of classifica-
tion decision could lead to erroneous predictions if the background environment is dynamic.

In spite of a large volume of work on collaborative target detection in sensor networks, a majority of the algorithms reported in the current literature are not robust relative to environmental changes, especially with the usage of low-fidelity, passive sensors. The rationale is that the passive sensors, used in a network, are sensitive to the background noise of the environment.

In many state-of-the-art techniques, the environmental disturbance is modeled as an additive noise, which is a compromising simplification because the environment is usually non-linearly coupled with the event of interest. It is also noted that noise in a dynamic environment, albeit being different from high-frequency additive noise, may generate a similar texture in the time series, which could be treated as a possible event of interest in the same time scale. Consequently, false interpretations could be derived based on the hard decision boundaries that are learned in the training phase. The decisions made by using thresholds based on sensed data or classification boundaries trained for identifying target classes may lead to false interpretations during the actual operation. Even if the final decisions are made by collaboration between different nodes [70], they could be significantly inaccurate.

Several researchers have addressed optimization of network performance for different applications, such as: target localization [71], target tracking [72, 73], sensor querying [74], and for overall performance improvement [75]. However, most of the reported research in this area considers only supervised static selection methods, in which a fixed subset of sensors in the network are chosen from objective optimization procedures in the training phase. Such is the case only when the dependence between time series from different sensor nodes does not change or evolve over time. Another set of relevant literature has been reported in the field of dynamical systems and mathematical physics (see, for example, [76, 77]), where various information-theoretic measures like permutation entropy and transfer entropy have been defined and explored to analyze the notion of causality between various stochastic processes.

1.3 Organization of the Dissertation

The dissertation is organized into nine chapters, including the present one. The organization of the dissertation along with major contributions is presented below.

1. Chapter 2 presents underlying concepts, which are applied in this dissertation, from multiple topics such as stochastic processes, symbolic dynamics, information theory, and finite state automata. The symbolic Markov models highlighted in this dissertation, named after D-Markov machines and \( \times \) D-Markov machines, belong to shifts of finite type (SFT) in symbolic dy-
namics. SFT is a proper subset of probabilistic deterministic finite automata (PDFA) in finite state automata, the later is used to describe and represent the symbolic Markov models of interest. The hidden Markov model (HMM) is the generalization of the PDFA model in stochastic processes. Next in the chapter, the advantages and disadvantages of the existing framework of Markov modeling based symbolic time series analysis is discussed. Finally, symbolic dynamic filtering, a recently reported tool for time series analysis, is introduced as the foundation and starting point of the research in this dissertation.

2. Chapter 3 and 4 present two frameworks for system parameter identification and state estimation via symbolic analysis of synchronized input-output time series pairs. The main challenge is to identify system characteristics under highly non-linear dynamics and fluctuating input at different working conditions. For system parameter identification where the parameter varies in a slow time scale, we proposed an information extraction scheme in the 2-dimensional input-output space using the trajectories of synchronized input-output pairs. On the other hand, a hybrid recursive Bayesian filter is proposed to combine the D-Markov model, a dynamic data-driven approach, with a system state space model for real-time state estimation. Both approaches are validated on the experimental data collected from a commercial lead-acid battery for state-of-health (SOH) identification and state-of-charge (SOC) estimation.

3. Chapter 5 proposes a general framework of dynamic information extraction from physical processes via video data. Compared to mainstream image processing and computer vision techniques, this research focuses on capturing the dynamic characteristics of physical processes via dimensionality reduction on sequences of images. A low-complexity feature extraction approach with three stages of dimensionality reduction is proposed for this unique purpose. The outcome of this algorithm is a symbol sequence which represents the dominant temporal pattern of dynamics embedded in the original video data. The proposed approach has been applied to classification problems for different combustion working conditions.

4. Chapter 6 and 7 discuss the applications of symbolic time series analysis on information fusion and evaluation in sensor networks under dynamic environments. The challenge is to identify and locate event information of interest under the presence of environmental disturbances whose dynamic characteristics are very similar to those of the event. Additionally each sensor node might be impacted differently from the ambient noise due to its location and orientation. The information extraction should be able to minimize the individual differences due to the environment, while simultaneously amplify the
potential event information from the network. In detail, Chapter 6 proposes multiple fusion schemes based on symbolic dynamic filtering (SDF) at the data, feature, and decision level. Chapter 7 proposes an information casualty measurement which can evaluate the contribution from each sensor to the network at the information level.

5. Chapter 8 proposes an unsurprised symbolization algorithm which integrates both partitioning and modeling of time series, as opposed to the majority of existing literature where they are done separately. The goal is to extract temporal patterns embedded in time series and represent them as a symbolic Markov model with an alphabet set and state set. Compared to the existing algorithms, such as symbolic dynamic filtering (SDF), the proposed algorithm requires minimum supervised inputs such as data labels and predefined parameters (e.g., alphabet size and depth of Markov model). The proposed method is validated on both simulation and experimental data, as well as compared to other commonly applied time series symbolization algorithms.

The dissertation is summarized and concluded in Chapter 9 along with a few recommendations of future research directions that can possibly emerge from this research. Finally, the dissertation is supported by three appendices.

1. Appendix A presents three computer vision algorithms applied in the first stage of information extraction from physical process video in Chapter 5.

2. Appendix B presents three unsupervised clustering algorithms applied in the fusion scheme in feature level for target detection problem under dynamic environment in Chapter 6.

3. Appendix C presents the concept and algorithm of Principal Component Analysis (PCA) for image dimensionality reduction in Chapter 5 and network information extraction in Chapter 7.
Mathematic Preliminaries and Previous Works

This first half of this chapter presents the basic concepts and principles in symbolic dynamics, finite automata, hidden Markov models (HMM), and information theory which are applied in this dissertation. The second half provides the existing framework for symbolic time series analysis and the genial symbolic dynamic filtering algorithm which is recently reported to the literature, which are the foundation of both the theoretical and practical application breakthrough and contributions of this dissertation.

2.1 Symbolic Dynamics

Important concepts and related properties are presented without proofs. The details of proofs can be found in Chapter 1, 2 and 6 of Lind and Marcus [78] as well as Chapter 9 [79] and Chapter 10 [80] of Handbook of formal languages, Vol. 2.

Definition 2.1.1. (Alphabet) Alphabet Σ is a (non-empty) finite set of symbols, also called letters: Σ = {σ₁, σ₂, · · · , σₙ}. The cardinality of the alphabet, also called alphabet size, is denoted as |Σ|.

Definition 2.1.2. (Word) A word, also called symbol block, is a (finite-length) sequence of symbols belonging to the alphabet Σ, where the length of a word w = σ₁σ₂ · · · σₗ with σᵢ ∈ Σ is |w| = ℓ. The empty word ε has zero length, i.e., |ε| = 0.

1. The set of all words, including the empty word ε, is denoted by Σ*, while Σ⁺ ≜ Σ* \ {ε};

2. The product of two words u and v, denoted as uv, is the word obtained by concatenating word v after u;
3. A word \( v \) is said to be a factor of a word \( u \) if \( u = u'vu'' \) for some words \( u' \) and \( u'' \); It is a prefix of \( u \) if \( u' = \epsilon \), and a suffix of \( u \) if \( u'' = \epsilon \).

4. The collection of all bi-infinite (two-sided infinite) words over alphabet \( \Sigma \) is denoted as \( \Sigma^\mathbb{Z} \), it is also called full shift space.

5. The collection of all one-sided infinite words over alphabet \( \Sigma \) is denoted as \( \Sigma^\mathbb{N} \).

Definition 2.1.3. (Full shift space) The full shift space \( (\Sigma^\mathbb{Z}) \) is the collection of all bi-infinite words over alphabet \( \Sigma \). In addition, the collection of all one-sided infinite words over alphabet \( \Sigma \) is denoted as \( \Sigma^\mathbb{N} \).

Definition 2.1.4. (Forbidden words) Let \( \mathcal{F} \) be a collection of words over \( \Sigma \). Elements in \( \mathcal{F} \) are called forbidden words.

Definition 2.1.5. (Shift map) For any given bi-infinite word \( x \in \Sigma^\mathbb{Z} \), the shift map is the function \( \sigma : \Sigma^\mathbb{Z} \rightarrow \Sigma^\mathbb{Z} \) that maps \( x \) to \( y \) as:

\[
y_k = (\sigma(x))_k = x_{k+1} \quad \text{for all} \quad k \in \mathbb{Z}
\]

In other words, \( \sigma \) shifts \( x \) to the left by one position.

Definition 2.1.6. (Shift space and forbidden words) Shift space is a subset \( X_\mathcal{F} \subset \Sigma^\mathbb{Z} \) of bi-infinite words \( x \) that do not contain any words belonging to the forbidden set \( \mathcal{F} \) which is a collection of words over an alphabet \( \Sigma \) such that

\[
X_\mathcal{F} = \{ x \in \Sigma^\mathbb{Z} \mid x_i \ldots x_j \notin \mathcal{F} \ (i \leq j) \}
\]

where the elements in \( \mathcal{F} \) are called forbidden words.

It is noted that

- A shift space \( X_\mathcal{F} \) is shift invariant, \( \sigma(X_\mathcal{F}) = X_\mathcal{F} \).
- A forbidden set may be empty, i.e., \( \mathcal{F} = \emptyset \Leftrightarrow X_\mathcal{F} = \Sigma^\mathbb{Z} \).

Definition 2.1.7. (Language) Let \( X \) be a shift space over alphabet \( \Sigma \), and let \( \mathcal{B}_n(X) \) be the set of all words of length \( n \) that occur (or are allowed) in points in \( X \). The language of \( X \) is the collection:

\[
\mathcal{B}(X) = \bigcup_{n=1}^{\infty} \mathcal{B}_n(X)
\]

where it is noted that

1. \( \mathcal{B}(X_\mathcal{F}) \cap \mathcal{F} = \emptyset \) and \( \mathcal{B}(X_\mathcal{F}) \cup \mathcal{F} \subset \Sigma^\mathbb{Z} \).
2. A shift space $X$ is said to be irreducible if for every pair of words $u, v \in B(X)$, there exists $w \in B(X)$ such that $uwv \in B(X)$.

**Definition 2.1.8. (Irreducible shift space)** A shift space $X$ is said to be irreducible if for every pair of words $u, v \in B(X)$, there exists $w \in B(X)$ such that $uwv \in B(X)$.

**Definition 2.1.9. ($N^{th}$ higher block shifts)** Let $X$ be a shift space over $\Sigma$, and $\Sigma_X^{[N]} = B_N(X)$ be the collection of all allowed words of length $N$ in $X$. We can consider $\Sigma_X^{[N]}$ as an alphabet in its own right, and form the full shift $(\Sigma_X^{[N]})^\mathbb{Z}$. Define the $N^{th}$ higher block code $\beta_N : X \to (\Sigma_X^{[N]})^\mathbb{Z}$ by:

$$(\beta_N(x))_k = x_{[k-N+1,k]}$$

Then the $N^{th}$ higher block shift $X^{[N]}$ is the image $X^{[N]} = \beta_N(X)$ in the full shift over $\Sigma_X^{[N]}$.

**Definition 2.1.10. (Sliding block code and symbolic dynamics)** Let $X$ be a shift space over $\Sigma$, and define $\Phi : B_{m+n+1}(X) \to A$ be a block map from all allowed words of length $(m+n+1)$ in $X$ to another alphabet $A$. Then, the sliding block code with memory $m$ and anticipation $n$ induced by $\Phi$ is the map $\varphi : X \to A^\mathbb{Z}$ defined as $y = \varphi(x)$ with $y_k = \Phi(x_{[k-m,k+n]})$. For a shift space $X$, if $\varphi : X \to X$ is a sliding block code of itself, then $(X, \varphi)$ is called a symbolic dynamical system.

![Figure 2.1: Shifting Block Code](image)

**Proposition 2.1.1.** Let $X$ and $Y$ be a shift spaces. If $\phi : X \to Y$ is a sliding block code, then $\phi$ is a continuous map and $\phi \circ \delta_X = \delta_Y \circ \phi$.

1. If $\phi : X \to Y$ is surjective (i.e., onto), then it is called a factor code from $X$ to $Y$.

2. If $\phi : X \to Y$ is injective (i.e., one-to-one), then it is called an embedding of $X$ into $Y$.

3. If $\phi : X \to Y$ is invertible, then there exists a unique sliding block code $\psi : Y \to X$ such that $\psi(\phi(x)) = x$ for all $x \in X$ and $\phi(\psi(y)) = y$ for all $y \in Y$.
Definition 2.1.11. (Topological conjugacy) A sliding block code \( \phi : X \to Y \) is a topological conjugacy from \( X \) to \( Y \) if it is bijective (i.e., one-to-one and onto) and invertible. Two shift spaces \( X \) and \( Y \) are conjugate \((X \cong Y)\) if there is a conjugacy from \( X \) to \( Y \).

\[ \begin{array}{ccc}
X & \xrightarrow{\delta_X} & X \\
\downarrow \phi & & \downarrow \phi \\
Y & \xrightarrow{\delta_Y} & Y
\end{array} \]

Figure 2.2: Topological conjugacy from \( X \) to \( Y \)

Proposition 2.1.2. Let \( X \) be a shift space. If \( \phi : X \to Y \) is a sliding block code, then \( Y \) is also a shift space with induced shift map \( \delta_Y \).

Definition 2.1.12. (Symbolic Dynamical system) A symbolic dynamical system \((M, f)\) consists of a compact metric space \( M \) with a continuous (i.e., continuous and bijective) map \( f : M \to M \).

Remark 2.1.1. Let \( X \) be a shift space and \( \phi : X \to X \) is a sliding block code of itself. Then \((X, \phi)\) is a symbolic dynamical system.

2.2 Probabilistic Deterministic Finite Automata (PDFA)

This subsection presents the framework and fundamental properties of finite state automata.

Definition 2.2.1. (Deterministic finite automaton) A deterministic finite automaton (DFA) [81] is a 5-tuple \( G = (\Sigma, Q, \delta, Q_I, Q_F) \) where:

1. \( \Sigma \) is a non-empty finite set, called the symbol alphabet;

2. \( Q \) is a non-empty finite set, called the set of states;

3. \( \delta : Q \times \Sigma \to Q \) is the injective transition function;

4. \( Q_I \) is a set of initial states \((Q_I \subset Q)\);
5. **Q_F** is a set of final states \( (Q_F \subset Q) \).

An automaton \( G \) is called trim [80] if any state is accessible from \( Q_I \) and can access \( Q_F \).

This paper deals with finite trim automaton \( (\Sigma, Q, \delta, Q_I, Q_F) \), in which all states belong to both \( Q_I \) and \( Q_F \). From now on, the deterministic finite automata is denoted as a 3-tuple \( G = (\Sigma, Q, \delta) \).

**Definition 2.2.2. (Extended Map)** The extended state transition map \( \delta^*: Q \times \Sigma^* \rightarrow Q \) transfers one word to another through finitely many transitions such that, for all \( q \in Q, \sigma \in \Sigma \) and \( w \in \Sigma^* \), \( \delta^*(q, w\sigma) = \delta(\delta^*(q, w), \sigma) \) where \( w\sigma \) is the suffixing of the symbol sequence \( w \) by the symbol \( \sigma \).

**Definition 2.2.3. (Irreducible DFA)** A deterministic DFA \( G \) is said to be irreducible if, for all \( q_1, q_2 \in Q \), there exists a symbol sequence \( w_{1,2} \in \Sigma^* \) such that \( q_1 = \delta^*(q_2, w_{1,2}) \).

**Definition 2.2.4. (PDFA [82])** A probabilistic deterministic finite-state automaton (PDFA) is constructed on the algebraic structure of deterministic finite-state automata (DFA) \( G = (\Sigma, Q, \delta) \) as a pair \( K = (G, \Pi) \), i.e., the PDFA \( K \) is a 4-tuple \( K = (\Sigma, Q, \delta, \Pi) \), where:

1. \( \Sigma \) is a non-empty finite set, called the symbol alphabet, with cardinality \( |\Sigma| < \infty \);
2. \( Q \) is a non-empty finite set, called the set of states, with cardinality \( |Q| < \infty \);
3. \( \delta: Q \times \Sigma \rightarrow Q \) is the state transition map;
4. \( \Pi: Q \times \Sigma \rightarrow [0, 1] \) is the symbol generation matrix (also called probability morph matrix) that satisfies the condition \( \sum_{j=1}^{||\Sigma||} \Pi_{ij} = 1 \) \( \forall q_i \in Q \), and \( \Pi_{ij} \triangleq \Pi(q_i, \sigma_j) \) is the probability of emission of the symbol \( \sigma_j \in \Sigma \) when the state \( q_i \in Q \) is observed.

**Definition 2.2.5. (PNFA [82])** A probabilistic nondeterministic finite-state automaton (PNFA) is a probabilistic deterministic finite-state automaton (PDFA) with a state transition as \( \delta: Q \times \Sigma \rightarrow P(Q) \), where \( P(Q) \) denotes the power set of \( Q \).

**Proposition 2.2.1.** PDFA \( \subseteq \) PNFA, i.e., probabilistic deterministic finite-state automaton (PDFA) is a proper subset of probabilistic nondeterministic finite-state automaton (PNFA).

**Proof.** The proof of Proposition 2.2.1 can be found in [83].
Definition 2.2.6. (Edge shift) Let $\mathcal{K} = (\Sigma, Q, \delta, \Pi)$ be a PDFA, where each existing transition $\delta(p, \sigma) = q$ is represented as a labelled edge $e \in Q \times \Sigma \times Q$, with an initial state $i(e) = p \in Q$, a label $l(e) = \sigma \in \Sigma$ and a terminal state $t(e) = q \in Q$. Then, the shift space $X_\mathcal{K}$, called edge shift, is the shift space over the alphabet $\Sigma$ specified by:

$$X_\mathcal{K} = \{(l(e_k))_{k \in \mathbb{Z}} \in \Sigma^\mathbb{Z} : t(e_k) = i(e_{k+1}) \text{ for all } k \in \mathbb{Z}\}$$

Next the notion of shift of finite type is brought in, which can be represented as finite state automata.

Definition 2.2.7. (Shift of finite type) A shift space $X$ with a finite set of forbidden words is called a shift of finite type (SFT). An SFT is $M$-step if it can be described by a collection of forbidden words, each of which has length $M + 1$.

Some of the properties of shifts of finite type are stated below.

1. If $X$ is a shift of finite type, then there is an $M \geq 0$ such that $X$ is $M$-step.

2. A shift space $X$ that is conjugate to a shift of finite type $Y$ is itself a shift of finite type.

The following theorem [78] shows that any shift of finite type can be recorded to an edge shift, which is generated from a deterministic finite automaton.

Theorem 2.2.1. If $X$ is an $M$-step shift of finite type over alphabet $\Sigma$, then there is a deterministic finite automaton $\mathcal{K} = (\Sigma, Q, \delta, \Pi)$ such that $X^{[M+1]} = X_\mathcal{K}$, where states are defined as $Q = B_M(X)$ and transition map is defined as $\delta(q[k] = x_{[k-M+1,k]}, x_{k+1}) = q[k+1] = x_{[k-M+2,k+1]}$ for all $k \in \mathbb{Z}$ and $x \in X$.

Proof. There exists a sliding block code $\Phi : B_M(X) \rightarrow Q$ with memory $m = M - 1$ and anticipation $n = 0$, in which any allowed word of length $M$ in $X$ is mapped to a state in $\mathcal{K}$. \qed

2.3 Hidden Markov Models

This subsection briefly introduces the concept of hidden Markov models (HMM).

Definition 2.3.1. (Markov chain) A Markov chain on a deterministic finite automaton $\mathcal{G} = (\Sigma, Q, \delta)$ is an assignment of initial state probabilities $\mu(q_i) \geq 0$ for all $q_i \in Q$ and conditional probabilities $p(\sigma_j|q_i)$ for all $\sigma_j \in \Sigma$ and $q_i \in Q$ such that

$$\sum_{q_i \in Q} \mu(q_i) = 1 \text{ and } \sum_{\sigma_j \in \Sigma} p(\sigma_j|q_i) = 1$$
Definition 2.3.2. *(Hidden Markov Model [83])* A discrete HMM is a 5-tuple $M = (\Sigma, Q, A, B, P_0)$ where

1. $\Sigma$ is an alphabet;
2. $Q$ is a set of states;
3. $A : Q \times Q \rightarrow [0, 1]$ is a mapping defining the probability of each transition;
4. $B : Q \times \Sigma \rightarrow [0, 1]$ is a mapping defining the emission probability of each symbol on each state;
5. $P_0 : Q \rightarrow [0, 1]$ is a mapping defining the initial probability of each state.
6. The following constraints must be satisfied for all $q \in Q$:
   \[
   \sum_{q' \in Q} A(q, q') = 1; \quad \sum_{\sigma \in \Sigma} B(q, \sigma) = 1; \quad \sum_{q \in Q} P_0(q) = 1
   \]

Definition 2.3.3. *(HMMT [83])* A discrete HMM with transition emission is a 5-tuple $M = (\Sigma, Q, A, B, P_0)$ where

1. $\Sigma$ is an alphabet;
2. $Q$ is a set of states;
3. $A : Q \times Q \rightarrow [0, 1]$ is a mapping defining the probability of each transition;
4. $B : Q \times \Sigma \times Q \rightarrow [0, 1]$ is a mapping defining the emission probability of each symbol on each state;
5. $P_0 : Q \rightarrow [0, 1]$ is a mapping defining the initial probability of each state.
6. Following constraints must be satisfied:
   \[
   \forall q \in Q, \quad \sum_{q' \in Q} A(q, q') = 1 \quad (2.1)
   \]
   \[
   \forall q, q' \in Q, \quad \sum_{\sigma \in \Sigma} B(q, \sigma, q') = \begin{cases} 1 & \text{if } A(q, q') > 0, \\ 0 & \text{otherwise} \end{cases} \quad (2.2)
   \]
   \[
   \sum_{q \in Q} P_0(q) = 1 \quad (2.3)
   \]

Remark 2.3.1. Alternative definitions of HMMs (or HMMTs) include a single non-emitting initial state $q_0$, also called a silent state. The use of a single initial state $q_0$ with initial probability $P_0(q_0) = 1$, results in models equivalent to HMMs (or HMMTs) described with 4-tuple $M = (\Sigma, Q, A, B)$. 

We present an important lemma states the relation between HMMs, HMMTs and PNFA as below.

**Lemma 2.3.1.** Let $K = (\Sigma, Q, \delta, \Pi)$ be a PNFA, there exists an equivalent HMMT $M = (\Sigma, Q, A, B)$.

**Lemma 2.3.2.** Let $M = (\Sigma, Q, A, B)$ be a HMMT, there exists an equivalent HMM $M' = (\Sigma, Q', A', B')$.

The proof of Lemma 2.3.2 and 2.3.1 can be found in [83].

Figure 2.3 depicts the relation among M-step shift of finite type (M-SFT) in symbolic dynamics, probabilistic deterministic finite-state automaton (PDFA) in theory of computation, and Hidden Markov Model (HMM) in stochastic process.

![Figure 2.3: Relationship among M-step shifts of finite type (M-SFT) in symbolic dynamics, probabilistic deterministic finite-state automata (PDFA) in the theory of computation, and hidden Markov models (HMM) in stochastic processes.](image)

We introduce a special class of PDFA, called D-Markov machines, belong to the class of shifts of finite type (SFT). This class of model has a simple algebraic structure and is computationally efficient to construct and implement.

**Definition 2.3.4.** (D-Markov Machine [8, 11]) Let $S = \{s_1, s_2, \ldots \}$ be a discrete symbol sequence that is assumed to be a statistically stationary stochastic process. The probability of occurrence of a new symbol depends only on the most recent $D$ consecutive symbols, where $D$ is a positive integer, i.e.,

$$P[s_k | s_{k-D} \cdots s_{k-1}] = P[s_k | s_{k-D} \cdots s_{k-1}]$$ (2.4)

where $D$ is called the depth of the D-Markov machine. Then, a D-Markov machine (modeled by an HMM and presented as a PDFA in which each state is represented by a finite history of $D$ symbols) is defined as a 4-tuple $\mathcal{M} \triangleq (Q, \Sigma, \delta, \pi)$ such that:

1. $Q = \{q_1, q_2, \ldots, q_{|Q|}\}$ is the state set corresponding to symbol sequence $S$, where $|Q|$ is the (finite) cardinality of the set of states.
2. $\Sigma = \{s_1, ..., s_{|\Sigma|}\}$ is the alphabet set of symbol sequence $S$, where $|\Sigma|$ is the (finite) cardinality of the alphabet $\Sigma$ of symbols.

3. $\delta : Q \times \Sigma \rightarrow Q$ is the state transition mapping. It is noted that the PDFA structure is built on $S$ and thus, the transition map explains the same symbol sequence;

4. $\pi : Q \times \Sigma \rightarrow [0, 1]$ is the morph matrix of size $|Q| \times |\Sigma|$; the $ij$th element $\pi(q_i, \sigma_j)$ of $\Pi$ denotes the probability of finding the symbol $\sigma_j$ at next time step while making a transition from the state $q_i$.

Thus, the symbol sequence is compressed as a PDFA by approximating the states by words of finite length from the symbol sequence. The PDFA induces a Markov chain of finite order where the parameters of the Markov chain i.e., the stochastic matrix is estimated from data by following a maximum a priori probability (MAP)) approach under the assumptions of sufficiently long data [11]. Once the parameters are estimated, they are used as temporal representation for the underlying data, which can be used for different machine learning applications (e.g., pattern matching and clustering).

Next the notion of $\times$D-Markov machine is introduced, and the underlying concept is illustrated in Figure 2.4.

![Figure 2.4: $\times$D-Markov (pronounced as cross D-Markov) modeling and feature extraction](image)

**Definition 2.3.5. ($\times$D-Markov Machine [84])** Let $U \triangleq \ldots u_{k-1}u_ku_{k+1} \ldots$ and $Y \triangleq \ldots y_{k-1}y_ky_{k+1} \ldots$ be two (time-synchronized) stochastic symbol sequences. Then, a $\times$D-Markov machine, where the Markov assumption holds for $Y$ with respect to $U$, is defined as a 5-tuple $M_{1\rightarrow2} \triangleq (Q_1, \Sigma_1, \Sigma_2, \delta_1, \Pi_{1\rightarrow2})$ such that:

1. $\Sigma_1 = \{s_1, ..., s_{|\Sigma_1|}\}$ is the alphabet of symbols in $U$.

2. $\Sigma_2 = \{\sigma_1, ..., \sigma_{|\Sigma_2|}\}$ is the alphabet of symbols in $Y$.  

18
3. $Q_1 = \{q_1, q_2, \ldots, q_{|Q_1|}\}$ is the set of states corresponding to $U$.

4. $\delta_1: Q_1 \times \Sigma_1 \rightarrow Q_1$ is the state transition mapping. It is noted that the PDFA structure is built on $U$ and thus, the transition map $\delta_1$ generates a sequence of symbols that belong to $\Sigma_1$; however the Markov assumption holds for $Y$ on the states inferred from $U$.

5. $\Pi_{1\rightarrow 2}: Q_1 \times \Sigma_2 \rightarrow [0, 1]$ is the $\times$-morph (pronounced as cross morph) matrix of dimension $|Q_1| \times |\Sigma_2|$; the $ij^{th}$ element $\Pi(i, j)$ of the $\times$-morph matrix $\Pi$ denotes the probability of finding the symbol $\sigma_j$ in the symbol string $Y$ at next time step while making a transition from the state $q_i$ of the DPFA constructed from the symbol sequence $U$.

Similarly, a 5-tuple $M_{2\rightarrow 1} \triangleq (Q_2, \Sigma_2, \Sigma_1, \delta_2, \Pi_{2\rightarrow 1})$ yields symbols in $\Sigma_1$ from states in $Q_2$.

In a $D$-Markov machine model that is capable of capturing the temporal transition behavior embedded in the input-current time series, the underlying PDFA model could represent different users’ driving styles for electric vehicles. On the other hand, $\times D$-Markov machine models can represent the causal dependency between two stochastic symbol sequences. Both $D$-Markov machines and $\times D$-Markov machines are used as temporal representations of the underlying time series data, which can be used for different machine learning applications (e.g., feature extraction and clustering for pattern classification).

### 2.4 Information Theory

**Definition 2.4.1. (Shannon Entropy [85])** The Shannon entropy $H$ of a discrete random variable $X$ with possible values $\{x_1, x_2, \ldots, x_n\}$ and probability mass function $p_X$ is defined as:

$$H(X) = -\sum_i p_X(x_i) \log p_X(x_i) \quad (2.5)$$

The Shannon entropy of $X$ describes the uncertainty or unpredictability of the information content in $X$.

**Definition 2.4.2. (Conditional Entropy [85])** The conditional entropy of the random variable $Y$ given a random variable $X$ is defined as:

$$H(Y|X) = \sum_{x \in X} p_X(x) H(Y|X = x)$$

$$= -\sum_{x \in X} p_X(x) \sum_{y \in Y} p_{Y|X}(y|x) \log p_{Y|X}(y|x) \quad (2.6)$$
where \( P_{Y|X}(y|x) \) is the conditional probability of \( y \in Y \) given that \( x \in X \) is observed.

**Definition 2.4.3. (Entropy rate of a state)** The entropy rate (also called conditional entropy) of symbols \( \Sigma \) conditioned on a state \( q \in Q \) is defined as:

\[
H(\Sigma|q) = \sum_{\sigma \in \Sigma} P(\sigma|q) \log P(\sigma|q)
\]  

(2.7)

where \( P(\sigma|q) \) is the conditional probability of a symbol \( \sigma \in \Sigma \) given that a state (i.e., a relevant symbol block) \( q \) has been observed.

**Definition 2.4.4. (Entropy rate)** The entropy rate (conditional entropy) over a state set \( Q \) is defined as:

\[
H(\Sigma|Q) = \sum_{q \in Q} P(q) H(\Sigma|q)
\]

\[
= - \sum_{q \in Q} \sum_{\sigma \in \Sigma} P(q) P(\sigma|q) \log P(\sigma|q)
\]  

(2.8)

Mukherjee and Ray [11] have claimed that lower is the entropy rate, more predictable is the next observed symbol conditioned on the current state. As entropy rate approaches 0, the next observed symbol tends to be deterministic.

**Definition 2.4.5. (Mutual Information [85])** Formally, the mutual information of two discrete random variables \( X \) and \( Y \) is defined as:

\[
I(X;Y) = \sum_{x \in X, y \in Y} p_{XY}(x,y) \log \left( \frac{p_{XY}(x,y)}{p_X(x)p_Y(y)} \right)
\]  

(2.9)

The mutual information is a measure of the mutual dependence between two random variables and it can be equivalently expressed as:

\[
I(X;Y) = H(X) - H(X|Y)
\]

(2.10)

\[
= H(Y) - H(Y|X)
\]  

(2.11)

**Definition 2.4.6. (Transfer Entropy [86, 87])** Transfer entropy from random process \( X_t \) to random process \( Y_t \) \((t \in \mathbb{N})\) is measured by the mutual information between \( Y_t \) and history of variable \( X_{t-D}^{t-1} = [x_{t-D} \ldots x_{t-1}] \) given the history of the influenced variable \( Y_{t-D}^{t-1} = [y_{t-D} \ldots y_{t-1}] \) in the condition:

\[
T_{X \rightarrow Y} = I(Y_t; X_{t-D}^{t-1}|Y_{t-D}^{t-1})
\]

\[
= H(Y_t|X_{t-D}^{t-1}, Y_{t-D}^{t-1}) - H(Y_t|X_{t-D}^{t-1})
\]  

(2.12)

Transfer entropy quantifies the uncertainty reduced in future values of \( Y \) by knowing the past values of \( X \) given past values of \( Y \). It is a measure of the amount of directed information transfer between two random processes.
2.5 Existing framework for PDFA Construction

This section presents the exiting time series symbolization in the framework of information-theoretic Markov modeling. There are three basic problems of interest that must be solved to extract the embedded temporal behavior in the (possibly processed) time series signal under consideration. These three problems are delineated below.

2.5.1 Selection of Alphabetize

- **Problem #1:** Determination of the alphabet size $|\Sigma|$, which is the number of different symbols in the alphabet $\Sigma$ to represent the time series as a sequence of symbols. The possible range of $|\Sigma|$ is from 2 to the length of the time series signal. The latter case is trivial and involves no information loss due to symbolization.

- **Problem #2:** Identification of the partitioning boundary locations, $P$, which quantizes the signal space into a finite number of mutually exclusive and exhaustive regions. Each region is associated with a unique symbol, to which the time series signal in that region is mapped.

- **Problem #3:** Identification of the state set $Q$, which consists of meaningful (i.e., relevant) symbol patterns that could reduce the uncertainty of the temporal transition in the symbol sequence [6].

![Figure 2.5: An example of time series symbolization: First, a partition is applied to the original time series; then, each signal is mapped to the symbol which is associated to the region; finally, the state is generated by combine symbol blocks.](image)

The procedure of time series symbolization is elucidated by an example in Figure 2.5. While systematic suboptimal procedures for alphabet size selection has been reported in literature (e.g., [88]), the alphabet size in the example of Figure 2.5
is selected ad-hoc and is application-dependent. This practice is often followed because the computation complexity of the symbolic time series is approximately proportional to the alphabet size, while the information gain for an increase in alphabet size is not. From this perspective, the Shannon entropy excess is defined for \( n \) symbols as a measure of the increment in information gain from \((n - 1)\) symbols.

**Definition 2.5.1. (Shannon Entropy Excess [89])** The Shannon entropy excess \( h(n) \) for \( n \) symbols is the increment in maximum uncertainty as the a new symbol is added to the existing \( n \) symbols in the alphabet.

\[
h(n + 1) = H_{\text{max}}(n + 1) - H_{\text{max}}(n), \quad n = 1, 2, 3, \cdots
\]

where \( H_{\text{max}}(n) = -\sum_{k=1}^{n} \frac{1}{n} \log \frac{1}{n} = \log n \) is defined by the uniform distribution of the symbols to achieve the maximum uncertainty, and \( H_{\text{max}}(1) \triangleq 0 \).

![Figure 2.6: Maximum Shannon entropy and entropy excess at different alphabet sizes.](image)

Figure 2.6: Maximum Shannon entropy and entropy excess at different alphabet sizes.

Figure 2.6 depicts the profiles of maximum Shannon entropy and entropy excess as functions of the alphabet size \( |\Sigma| \). As \( |\Sigma| \) is increased, the information gain (i.e., Shannon entropy excess) is decreased. So, it is logical to make a trade-off between the computational complexity and the information loss when determining the alphabet size during time series symbolization [11].

### 2.5.2 Determination of Partitioning locations

Once the alphabet size is determined, the next step is identification of partitioning boundary locations. The objective here is to obtain a symbol sequence that will retain temporal transition behavior embedded in the time series signal. As mentioned earlier in chapter 1, a generating partition would be desirable from
the perspectives of information retention; however, the necessary conditions for existence of generating partition include: noise-free and infinitely long time series, which cannot be guaranteed in real-life situations. It is noted that there exist a few ad-hoc unsupervised algorithms that can generate partition locations for a given (one-dimensional) finite length time series in special cases (e.g., (one-dimensional) logistic map).

Rajagopalan et al. [15] have reported a maximum entropy partitioning (MEP) algorithm (also known as equal frequency partition), where the key idea is to uniformly distribute the occurrence probability of each symbol in the generated symbol sequence such that the information-rich regions of the time series are partitioned finer and those with sparse information are partitioned coarser. In general, Shannon entropy of a symbol sequence is expressed as:

\[
H(\Sigma) = -\sum_{k=1}^{|
\Sigma\n|} P_k \log P_k
\]

where \( P_k \) is the probability of occurrence of the symbol \( \sigma_k \). Algorithm 1 explains the procedure of maximum entropy partitioning (MEP), which is adopted in this paper; the entropy of the symbol sequence is maximized because \( P_k \approx \frac{1}{|
\Sigma\n|} \forall k \).

**Algorithm 1 Maximum entropy partitioning**

**Require:** Time series \( X = [x_1, x_2, \ldots, x_N] \) and alphabet size \( |
\Sigma\n| \).

**Ensure:** Partition position vector \( P \in \mathbb{R}^{|
\Sigma\n|+1} \)

1: Sort the time series \( X \) in the ascending order;
2: Let \( K = \text{length}(X) \);
3: Assign \( P(1) = X(1) \), i.e., the minimum element of \( X \)
4: for \( i = 2 \) to \( |
\Sigma\n| \) do
5: \( P(i) = X\left(\text{ceil}\left(\frac{(i - 1) \times K}{|
\Sigma\n|}\right)\right) \)
6: end for
7: Assign \( P(|\n\Sigma\n| + 1) = X(K) \), i.e., the maximum element of \( X \)

The next task is to discover the temporal patterns from the generated symbol sequence, where the PDFA states are identified by selecting relevant strings of symbols in a temporal order. There are two basic criteria to evaluate the significance of a possible symbol block with a finite length \( D \):

- **Criterion#1**: The frequency of the symbol block occurring in the symbol sequence.
- **Criterion#2**: Influence of the symbol block on the next observed symbol in temporal order.
If a certain symbol block appears frequently in the symbol sequence while the
distribution of the symbol observed next to it is statistically relevant (e.g., not
randomly occurring), then the symbol block is qualified to be a state. It is noted
that the relevant symbol blocks for a given symbol sequence may not be of equal
to identify states with different lengths by making use of an information-theoretic
measure between states and symbols in terms of the conditional entropy.

2.6 Symbolic Dynamic Filtering (SDF)

This section briefly describes the underlying concept of symbolic dynamic filtering
(SDF) upon which the proposed dynamic-data-driven tool of battery parameter
identification is constructed; SDF encodes the behavior of (possibly nonlinear) dy-
namical systems from the observed time series by symbolization and construction of
state machines (i.e., probabilistic deterministic finite state automata (PDFA)) [90].
This is followed by computation of the state emission matrices that are represen-
tatives of the evolving statistical characteristics of the battery dynamics.

The core assumption in the SDF analysis for construction of probabilistic de-
terministic finite state automata (PDFA) from symbol strings is that the symbolic
process under both nominal and off-nominal conditions can be approximated as
a Markov chain of order \(D\), called the \(D\)-Markov machine, where \(D\) is a positive
integer.

The construction of a \(D\)-Markov machine is based on: (i) state splitting that
generates symbol blocks of different lengths according to their relative importance;
and (ii) state merging that assimilates histories from symbol blocks leading to the
same symbolic behavior. Words of length \(D\) on a symbol string are treated as
the states of the \(D\)-Markov machine before any state-merging is executed. Thus,
on an alphabet \(\Sigma\), the total number of possible states becomes less than or equal
to \(|\Sigma|^D\); and operations of state merging may significantly reduce the number of
states [91].

2.6.1 Signal Pre-processing

This subsection addresses the procedure for time series symbolization. First, bat-
tery input current and output voltage are individually normalized into zero-mean
and unit-variance time series as:

\[
\tilde{\theta}_i = \frac{\theta_i - \mu_{\theta}}{\sigma_{\theta}} \text{ for } i = 1, 2, \ldots, L
\] (2.15)

where \(\mu_{\theta}\) is the mean and \(\sigma_{\theta}\) is the standard deviation of the unprocessed time
series \(\theta_1, \theta_2, \ldots, \theta_L\) of finite length \(L\). The objective of preprocessing is to remove
the undesirable effects due to mean shifting (e.g., due to measurement drifts)
and variance fluctuation (e.g., due to different noise level) from time series before symbolization.

Then a independent and identical zero-mean gaussian noise \( Z \sim N(0, \sigma_Z^2) \) is uniformly added to the time series \( r \). The pre-processed signal for the \( i^{th} \) sensor at each time instant \( k \) can be expressed as:

\[
\hat{y}_i[k] = y_i[k] + Z
\]  
(2.16)

To avoid losing too much information and undermining the PDFA feature, the standard deviation of the added gaussian noise is chosen as \( \sigma_Z = 1/|\Sigma| \) in this paper.

### 2.6.2 Time Series Symbolization

Symbolization requires partitioning (also known as quantization) of the normalized time series, where the signal space is partitioned into a finite number of cells that are labeled as symbols, i.e., the number of cells is identically equal to the cardinality \( |\Sigma| \) of the (symbol) alphabet \( \Sigma \). As an example, for the one-dimensional time series in Figure 2.4, the alphabet \( \Sigma = \{0, 1, 2, 3\} \), i.e., \( |\Sigma| = 4 \), and three partitioning lines divide the ordinate (i.e., y-axis) of the time series profile into four mutually exclusive and exhaustive regions that form a partition, where each region is labeled with one symbol from the alphabet \( \Sigma \). If the value of time series at a given instant is located in a particular cell, then it is coded with the symbol associated with that cell. In this way, a symbol string is generated from the (finite-length) time series. Details are reported in [90].

The ensemble of time series data is partitioned by using a partitioning tool, called maximum entropy partitioning (MEP) [92], that maximizes the entropy of the generated symbols; therefore, the information-rich cells of a data set are partitioned finer and those with sparse information are partitioned coarser (i.e., each cell contains approximately equal number of data points). The choice of alphabet size \( |\Sigma| \) largely depends on the specific data set and the allowable loss of information [91].

### 2.6.3 Feature Extraction by a D-Markov Machine

The PDFA states represent different combinations of blocks of symbols on the symbol string. In the graph of a PDFA, the directional edge (i.e., the emitted event) that interconnects a state (i.e. a node) to another state represents the transition probability between these states. Therefore, the “states” denote all possible symbol blocks (i.e., words) within a window of certain length, and the set of all states is denoted as \( Q = \{q_1, q_2, \ldots, q_Q\} \) and \(|Q| \) is the number of (finitely many) states. The procedure for estimation of the emission probabilities is presented next.
Given a (finite-length) symbol string $S$ over a (finite) alphabet $\Sigma$, there exist several PDFA construction algorithms to discover the underlying irreducible PDFA model $K$ of $S$, such as causal-state splitting reconstruction (CSSR) [93], D-Markov [90][92], and Compression via Recursive Identification of Self-Similar Semantics (CRISSiS [94]). All these algorithms start with identifying the structure of the PDFA $K \triangleq (Q, \Sigma, \delta, \pi)$. To estimate the state emission matrix, a $|Q| \times |\Sigma|$ count matrix $C$ is constructed and each element $c_{kj}$ of $C$ is computed as:

$$c_{kj} \triangleq 1 + N(q_k, \sigma_j)$$  \hspace{1cm} \text{(2.17)}$$

where $N(q_k, \sigma_j)$ denotes the number of times that a symbol $\sigma_j$ is generated from the state $q_k$ upon observing the symbol string $S$. The maximum a posteriori probability (MAP) estimates of emission probabilities for the PDFA $K$ are computed by frequency counting as

$$\hat{\pi}(\sigma_j | q_k) \triangleq \frac{c_{kj}}{\sum \ell c_{k\ell}} = \frac{1 + N(q_k, \sigma_j)}{|\Sigma| + \sum \ell N(q_k, \sigma_\ell)}$$  \hspace{1cm} \text{(2.18)}$$

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 $\hat{\pi}(\sigma | q) = \frac{1}{|\Sigma|}$ $\forall \sigma \in \Sigma$, i.e., the uniform distribution of event generation at the state $q$. The above procedure guarantees that the PDFA, constructed from a (finite-length) symbol string, must have an (elementwise) strictly positive morph map $\Pi$ and that the state transition map $\delta$ in Definitions 2.2.4 and 2.3.4 is a total function.

Having computed the emission probabilities $\hat{\pi}(\sigma_j | q_k)$ for $j \in \{1, 2, \cdots, |\Sigma|\}$ and $k \in \{1, 2, \cdots, |Q|\}$, the estimated emission probability matrix of the PDFA is obtained as:

$$\hat{\Pi} \triangleq \begin{bmatrix}
\hat{\pi}(\sigma_1 | q_1) & \cdots & \hat{\pi}(\sigma_{|\Sigma|} | q_1) \\
\vdots & \ddots & \vdots \\
\hat{\pi}(\sigma_1 | q_{|Q|}) & \cdots & \hat{\pi}(\sigma_{|\Sigma|} | q_{|Q|})
\end{bmatrix}.$$  \hspace{1cm} \text{(2.19)}$$

The procedure of symbolic dynamic filtering for feature extraction, which makes use of the estimated emission probability matrix $\hat{\Pi}$, is presented as Algorithm 3 in Appendix A.

### 2.6.4 Feature Extraction by a $\times D$-Markov Machine

For both algorithmic simplicity and computational efficiency, $\times D$-Markov machines have been adopted for modeling cross-dependence between two PDFAs. The states of PDFAs, $K_1$ and $K_2$ in Definition 2.3.5, are constructed in terms of their respective symbols when those individual PDFAs are represented as $D$-Markov machines. A typical state, generated from $X$ with depth of $D$, is represented by a symbol string $s_j s_{j+1} \cdots s_{j+D-1}$. 
Let $N(\sigma_j, q_k)$ denote the number of times that a symbol $\sigma_k$ is generated by PDFA $K_2$ when the state $q_j$ is observed (as a symbol string) by PDFA $K_1$. The maximum a posteriori probability (MAP) estimate of the symbol emission probability of the PDFA $K_2$ from PDFA $K_1$ is obtained by frequency counting [91] as:

$$\Pi_{1\rightarrow2}(q_k, \sigma_j) = \frac{1 + N(\sigma_j, q_k)}{|\Sigma_2| + \sum_{\ell} N(\sigma_\ell, q_k)} \quad (2.20)$$

If no event is generated at a combination of symbol $\sigma_j \in \Sigma_2$ and state $q_k \in Q_1$, then there should be no preference to any particular symbol and it is logical to have $\Pi_{1\rightarrow2}(k, j) = \frac{1}{|\Sigma_2|}$. The above procedure guarantees that $\times$-Markov machines, constructed from two (finite-length) symbol strings, will have an (elementwise) strictly positive $\times$-morph matrix $\Pi_{1\rightarrow2}$ that is constructed by estimating the individual emission probabilities $\Pi_{1\rightarrow2}(k, j)$ for $k \in \{1, 2, \cdots, |Q_1|\}$ and $j \in \{1, 2, \cdots, |\Sigma_2|\}$. Similar procedures hold for the $\times$-morph matrix $\Pi_{2\rightarrow1}$.

With an appropriate choice of partitioning, it is ensured that the resulting Markov chain model satisfies the ergodicity conditions [95]; in other words, under statistically stationary conditions, the probability of every state being reachable from any other state within finitely many transitions must be strictly positive. The statistics of the time series at an epoch $t^k$, represented by the estimated emission matrix $\hat{\Pi}^k$, change to $\hat{\Pi}^t$ at another epoch $t^t$. Accordingly, the estimated emission matrices $\hat{\Pi}^k$ and $\hat{\Pi}^t$ are treated as feature vectors that are postulated to have the imbedded information on the dynamical system at the epochs $t^k$ and $t^t$, respectively. Thus, the (quasi-)stationary emission matrix $\hat{\Pi}$ serves as the “feature” vector extracted from the time series and is used for pattern classification in the sequel. The evolution of the battery dynamics is captured as the divergence of the feature vector as defined below.

### 2.6.5 Distance of Symbol Sequences & PDFA Features

**Definition 2.6.1. (Hamming Distance)** Let $X = \{x_1 \cdots x_L\}$ and $Y = \{y_1 \cdots y_L\}$ be two symbol strings of the same (finite) length $L$. Then, the (scalar) Hamming distance between these two symbol strings is defined as:

$$d_H(X, Y) = \frac{1}{L} \sum_{i=1}^{L} h(x_i, y_i) \quad (2.21)$$

where

$$h(x, y) = \begin{cases} 1 & \text{if } x \neq y \\ 0 & \text{if } x = y \end{cases} \quad (2.22)$$

**Definition 2.6.2. (Feature Divergence)** Let $\hat{\Pi}_i$ be the feature extracted for the PDFA $G_i$ corresponding to the sensor $\mathcal{L}_i$, and let $\hat{\Pi}_j$ be the feature extracted for
the PDFA $G_j$ corresponding to the sensor $L_j$. Then, the (scalar) feature divergence between these two sensors is expressed as:

$$m(L_i, L_j) \triangleq d(\hat{\Pi}_i, \hat{\Pi}_j)$$

(2.23)

where $d$ is a suitable metric and popular choices of a metric (e.g., [90] and [96]) are

- Euclidean distance, $d_2(a, b) \triangleq (a - b)(a - b)'$
- City block distance, $d_1(a, b) \triangleq \sum_{j=1}^{n} |a_j - b_j|
- Cosine distance, $d_c(a, b) \triangleq 1 - \frac{ab'}{\sqrt{(aa')(bb')}}$

where $a, b$ are two row vectors having the same dimension and the column vector $a'$ is the transpose of $a$.

**Definition 2.6.3. (Proximity Matrix)** Let $C_r$ and $C_s$ be two clusters, each of which contains at least one symbol string and all symbol strings therein have the same length. Let $n_r$ be the number of sensors in cluster $C_r$, and let $S_{ri}$ and $S_{rj}$ be the $i^{th}$ and $j^{th}$ symbol strings, respectively, in the cluster $C_r$.

Then, the proximity matrix $A^D = [a^D_{rs}]$ between any two clusters is expressed elementwise as:

$$a^D_{rs} \triangleq D(C_r, C_s)$$

(2.24)

where $D(C_r, C_s)$, the cluster distance measure, is defined as:

$$D(C_r, C_s) \triangleq \frac{1}{n_r n_s} \sum_{i=1}^{n_r} \sum_{j=1}^{n_s} d_H(S_{ri}, S_{sj})$$

(2.25)

If the feature divergence $m$ is intended to be used as an indicator of a battery condition, then it is highly desirable to have a linear relationship between $m$ and $\theta$. From this perspective, the range of linearity between these two entities is obtained as a statistical model in terms of the coefficient of determination as defined below.

**Definition 2.6.4. (Coefficient of Determination)** As a measure of how well the linear least squares fit, $\hat{\theta}_k = a + b m_k$, performs as a predictor of an output $\theta$ in terms of the input $m$, where the scalars $a$ and $b$ are the intercept and slope of the linear fit, the coefficient of determination (COD) is defined as:

$$COD = 1 - \frac{\sum_{i=1}^{n}(\theta_i - \hat{\theta}_i)^2}{\sum_{i=1}^{n}(\theta_i - \bar{\theta})^2}$$

(2.26)

where $\bar{\theta} \triangleq \frac{1}{n} \sum_{i=1}^{n} \theta_i$ is the average of the output data. In this context, $\{m_k\}$ represents a sequence of the feature divergence and $\{\theta_k\}$ is the corresponding sequence of $(1 - SOH)$. The coefficient of determination COD ranges from 0 to 1 and COD = 1 implies a perfect linear fit.
2.7 Summary

This chapter summaries the existing framework for Markov-based time series symbolization and modeling. The underlying concepts can be formulated as the hidden Markov models (HMMs) in stochastic process, probabilistic deterministic finite automata (PDFA) in Turing machines, as well as shift of finite type in symbolic dynamics. The main tasks in symbolic dynamic filtering (SDF), a recently reported symbolic time series analysis tool, consists of two challenges: 1) determine the optimal alphabet size for the partitioning along with their locations in the signal space; 2) identify the optimal depth and state set for the Markov modeling when constructing PDFAs. The optimization problem for both tasks in general is to maximizing the representability of the obtained Markov model to the original time series. Although these two tasks should be under investigated together, the works reported in the open literature are commonly treat them as sequential steps in the solution and solve them independently.
Chapter 3

Battery State-of-Health Identification via Markov Modeling

This chapter addresses real-time identification of the state-of-health (SOH) parameter over the life span of a battery that is subjected to approximately repeated cycles of discharging/recharging current. In the proposed method, finite-length data of interest are selected via wavelet-based segmentation from the time series of synchronized input-output (i.e., current-voltage) pairs in the respective two-dimensional space. Then, symbol strings are generated by partitioning the selected segments of the input-output time series to construct a special class of probabilistic deterministic finite state automata (PDFA), called D-Markov machines. Pertinent features of the statistics of battery dynamics are extracted as the state emission matrices of these PDFA. This real-time method of SOH parameter identification relies on the divergence between extracted features. The underlying concept has been validated on (approximately periodic) experimental data, generated from a commercial-scale lead-acid battery. It is demonstrated by real-time analysis of the acquired current-voltage data on in-situ computational platforms that the proposed method is capable of distinguishing battery current-voltage dynamics at different aging stages, as an alternative to computation-intensive and electrochemistry-dependent analysis via physics-based modeling.

3.1 Introduction

Batteries are being increasingly used for reusable and safe energy storage in various application systems (e.g., electric & hybrid vehicles, renewable energy, power backup systems, and smart grids). A majority of these applications may require large battery packs that contain hundreds and even thousands of battery cells, to satisfy large and dynamic power demands; examples are plug-in electric vehicles and hybrid locomotives. The state-of-health (SOH) parameter is a measure of the
battery system’s ability to store and deliver electrical energy. Knowledge of the SOH parameter enhances preventive maintenance and reduction of life cycle cost through timely recharging and/or replacement of battery cells. Thus, accurate estimation of the SOH parameter is crucial for efficient operation of battery systems, including allocation of power and energy within and among the cells.

As an alternative to model-based analysis, Li et al. [97] have reported a dynamic data-driven method for identification of SOH by using the time series of the battery output voltage at different aging stages. The underlying concept is built upon the theory of symbolic dynamic filtering (SDF) [90][91] that extracts the dynamic information from symbolized time series of signals as probabilistic deterministic finite state automata (PDFA).

This chapter is an extension of the work reported by Li et al. [97] that solely used the time series of battery output (i.e., voltage) for parameter identification. In contrast to the work of Li et al. [97], the present work makes use of an ensemble of time series pairs of synchronized battery inputs (i.e., charging/discharging current) and battery outputs (i.e., voltage) for identification of the SOH parameter. These synchronized input-output pairs of time series are simultaneously analyzed to generate symbol strings that, in turn, are used to construct PDFA for information compression and feature extraction. The proposed dynamic data-driven method for the SOH parameter identification has been validated using experimental data of a (commercial-scale) lead-acid battery under varying input-output (e.g., current-voltage) conditions. Furthermore, the underlying software can be executed in real time on in-situ computational platforms and its implementation does not require any detailed knowledge of the battery system’s electrochemistry.

The chapter is organized into four sections, including the present section. Section 3.2 presents the wavelet-based segmentation algorithm for dynamic time series extraction. Section 3.3 presents the framework for time series symbolization in 2-D space. Section 3.5 presents the experimental data collection and results of SOH estimation based on the time series of synchronized input-output pairs. Section 3.6 summarizes this chapter.

### 3.2 Wavelet-based Time Series Segmentation

This subsection presents segment extraction of dynamic input-output pairs by conducting wavelet-based segmentation from the database of synchronized system input-outputs. Specifically, selection of the scale range of wavelet basis and the thresholding on wavelet coefficients are addressed. The wavelet-based analysis provides pertinent information of the signal simultaneously in the time domain and the frequency domain [98].

For a given wavelet basis function $\psi(t)$, the scaled and translated child wavelets
are defined as [99]:

\[ \psi_{\alpha,\tau}(t) = \sqrt{\frac{1}{\alpha}} \psi\left(\frac{t - \tau}{\alpha}\right) \]  

(3.1)

where \( \alpha \in (0, \infty) \) is the scale of wavelet transform and \( \tau \in (-\infty, \infty) \) is the time shift, and \( \psi \in L_2(\mathbb{R}) \) is such that \( \int_{-\infty}^{\infty} \psi(t)dt = 0 \) and the norm \( \|\psi\| = 1 \).

The continuous wavelet transform (CWT) of a function \( x(t) \) at a scale \( \alpha \) is represented as

\[ \tilde{x}(\alpha, \tau) = \int_{-\infty}^{\infty} \bar{\psi}_{\alpha,\tau}(t) x(t)dt \]  

(3.2)

where \( \bar{\psi} \) indicates the complex conjugate of \( \psi \); and the distinction between \( \bar{\psi} \) and \( \psi \) vanishes for real-valued wavelet basis functions. It is also noted that the time series of a continuous-time signal \( x(t) \) is its representation in the discrete-time domain and hence the integral in Eq. (3.2) is replaced by summation over a (finite) time series \( \{x[n]\}, \ n = 1, 2, \cdots, N \) to obtain the discrete wavelet transform (DWT) at a scale \( \alpha_m \) and time shift \( \tau_\ell \) as:

\[ \tilde{x}[\alpha_m, \tau_\ell] = \sum_{n=1}^{N} \bar{\psi}_{\alpha_m,\tau_\ell}[n] \ x[n] \]  

(3.3)

Every wavelet basis function can be associated with a purely periodic signal of frequency, called center frequency \( f_c \), which maximizes the Fourier transform of the wavelet modulus [100]. Consequently, the relationship among frequency points \( \varphi_m \), the associated scales \( \alpha_m \) for a given wavelet basis function and a sampling period \( \Delta \) of the time series is given as:

\[ \alpha_m = \frac{f_c}{\varphi_m \Delta} \]  

(3.4)

where \( \Delta \) is the sampling period of the analyzed signal. The wavelet basis function for analysis is selected based on the time-frequency characteristics of analyzed signals.

For a time series \( \{x[n]\}, \ n = 1, 2, \cdots, N \), the discrete Fourier transform is obtained as:

\[ \hat{x}[k] = \sum_{n=1}^{N} e^{-2\pi i k n / N} x[n] \text{ for } k = 1, 2, \cdots, N \]  

(3.5)

**Remark 3.2.1.** In general, the Fourier transform \( \hat{x}[k] \) of a signal is a complex number but its power spectral density \( |\hat{x}[k]|^2 \) is a non-negative real number.

Following Parseval’s theorem, the energy of the time series \( x[n] \) can be expressed in the discrete-time and discrete-frequency domains [101] as:

\[ \sum_{n=1}^{N} |x[n]|^2 = \sum_{k=1}^{N} |\hat{x}[k]|^2 \]  

(3.6)
Since the summand on the right hand side of Eq. (3.6) can be interpreted as a probability histogram describing the energy distribution of the signal at frequency points \( k \), the energy spectral density of a signal \( x[n] \) is defined as:

\[
S_x[k] = |\hat{x}[k]|^2 \quad \forall k \in \{1, 2, \cdots, N\}
\]  

(3.7)

The steps of the wavelet-based segmentation procedure, depicted in Fig. 3.1, are as follows.

- **Step 1:** Collection of (finite) time series data \( x[n], \ n = 1, 2, \cdots, N \) for a given sampling interval \( \Delta \). It is noted that the length \( N \) of the time series is user-selectable.

- **Step 2:** Computation of the discrete Fourier transform \( \hat{x}[k], \ k = 1, 2, \cdots, N \) and the corresponding PSD \( S_x[k], \ k = 1, 2, \cdots, N \). This step follows Eq. 3.5 and Eq. 3.7.
Step 3: Identification of frequency points of interest, $\varphi_m$, $m = 1, 2, \cdots, M$. In this step, the set $\{S_x[\varphi_m]\}$ is formed in terms of the $M$ points with highest values of the PSD $S_x[k]$, $k = 1, 2, \cdots, N$. [Note: $M$ is usually significantly less than $N$].

Step 4: Computation of the corresponding wavelet scales $\alpha_m$, $m = 1, 2, \cdots, M$. This step follows Eq. (3.4) with the corresponding frequency points $\varphi_m$ and the central frequency $f_c$ of the chosen wavelet basis function.

Step 5: Identification of the set of segmented time indices $\Gamma^m$ for each scale $\alpha_m$. In this step, the set of time shift indices $\Gamma^m \subset \{\tau_m^\ell\}_{\ell=1}^N$ are selected as the top $\xi_T$ fraction of the values of the wavelet coefficients $\tilde{x}_{\alpha_m}(\tau_m^\ell) = \tilde{x}[\alpha_m, \tau_m^\ell]$, $\ell = 1, 2, \cdots, N$ at a scale $\alpha_m$.

Step 6: Identification of the set, $T$, of all segmented time indices. This step is completed by taking the union of the sets of $\Gamma^m$’s among all scales $\alpha_m$, $m = 1, 2, \cdots, M$ as $T = \bigcup_{m=1}^M \Gamma^m$.

### 3.3 System Input-Output Pairs Symbolization

This subsection addresses the partitioning in the input-output space of single-input single-output (SISO) systems for symbolization of 2-dimensional time series data. Time series of input-output pairs are partitioned into a mutually exclusive and exhaustive set of finitely many segments, where a symbol string is generated by assigning a unique symbol to each segment of the input-output space.

Quasi-stationarity of SISO systems is assumed such that the system behavior is statistically stationary at the fast time scale of the process dynamics, while there exists observable non-stationary behavior evolving at a slow time scale. The notion of two time scales and their significance are discussed in the context of symbolization in a recent publication [91].

In this context, the maximum-entropy partitioning (MEP) [92] of the time-series data has been adopted to construct the symbol alphabet $\Sigma$ and to generate symbol strings. A pair of time series that represent input and output data is partitioned in the associated two-dimensional space to construct a symbolic string. Four alternative types of partitions have been used in the input-output space:

- **Partition Type 1** (Cartesian coordinates): First partition in the input axis (e.g., abscissa), and then partition in the output axis (e.g., ordinate) at individual input segments.

- **Partition Type 2** (Cartesian coordinates): First partition in the output axis, and then partition in the input axis at individual output segments.
• Partition Type 3 (Polar coordinates): First partition in the magnitude, and then partition in the phase at individual magnitude segments.

• Partition Type 4 (Polar coordinates): First partition in the phase, and then partition in the magnitude at individual phase segments.

![Diagram of System input-output pairs and Partition Boundaries](image)

**Figure 3.2: Construction of finite state automata (FSA) from time series**

Figure 3.2 depicts the underlying concept of symbolization of a 2-dimensional time series, where each segment in the left half plot is labeled by a unique symbol and Σ denotes the alphabet of all these symbols. The segment, visited by the time series plot takes a symbol value from the alphabet Σ. For example, having Σ = {α, β, γ, δ} in Fig. 3.2, a time-series \( x_0x_1x_2 \cdots \) generates a string of symbols in the symbol space as: \( s_0s_1s_2 \cdots \), where each \( s_i, i = 0, 1, 2, \cdots \), takes a symbol value from the alphabet Σ. This mapping is called symbolic dynamics as it attributes a (physically admissible) symbol string to the dynamical system starting from an initial state (for example, see the symbol string at the top right half plot of Fig. 3.2).

### 3.4 Proposed Method for Feature Extraction

The flow chart in Fig. 3.3 summarizes the basic procedures of the proposed information compression and feature extraction method for two-dimensional time series of synchronized input-output data. First, the raw data are normalized such that they have the properties of zero-mean and unit-variance. Second, the dynamic information of interest is extracted from the normalized data via wavelet-based segmentation in the time-frequency domain. Third, the concept of maximum entropy partitioning (MEP) [92] is adopted to partition the segmented data set into multiple states in the 2-dimensional input-output data space. Then, symbol strings are generated from the partitioned data sets and probabilistic deterministic finite state automata (PDFA) are constructed from the symbol strings, where a feature is represented as the emission matrix of a PDFA.
Figure 3.3: The flow chart for the proposed method of feature extraction

3.5 Experimental Validation

This section validates the algorithms of battery parameter identification with an ensemble of experimental data that have been collected from a lead-acid battery. The results of SOH identification are presented.

3.5.1 Data Acquisition and Processing

A brand new (12V AGM VRLA with 56Ah capacity) lead-acid battery has been used in the experiments. During the experiment, the battery has been charged or discharged according to given input (current) profiles at the room temperature and an ensemble of synchronized time-series of the input charge/discharge current and output voltage responses has been collected at the sampling frequency of 1Hz. A typical input current profile for this experiment is depicted in Fig. 3.4. The duration of a input profile is \(\sim 150\) hours, which consists of three capacity measure cycles that are followed by 25 duty cycles.

A capacity measures cycle is a slow, full discharge/dicharge cycle, where the battery is fully discharged followed by a full charge. In this experiment, the battery is discharged by a constant current of \(-20\)A for \(\sim 2\) hours at first. Then it is charged first by constant current of 20A until it reach voltage of 13.8V and kept charging at this voltage level for 3 hours. The maximum capacity at that time can be measured by integrating the current during the charge period. Three computed battery maximum capacities are obtained from these capacity measurement cycles, the mean value of them is considered as the nominal maximum capacity for that particular time. Since there are five similar (i.e., same pattern) input profiles in total are applied to the battery during the whole experiment. The degradation of battery SOH (i.e., the ratio of maximum capacity between ”now” and when it is brand new) is obtained and depicted in Fig. 3.5.

Total 25 duty cycles are divided into groups of five. The transition time between two consecutive groups is \(\sim 6\) hours with charging to full capacity, while the transition time between two consecutive duty cycles in one group is \(\sim 1.2\)
hours with inadequate recharging. Each duty cycle last \( \sim 2.7 \) hours, which is composed of \( \sim 75 \) ”Hotel-Pulse” cycles, as depicted in Fig. 3.4(b). Each individual ”Hotel-Pulses” cycle (i.e., duration of 120s) consists of a ”hotel” load (i.e., relatively steady discharge due to ”hotel” needs like lighting and other electrical equipments) and a discharge pulse followed by a charge (i.e., regeneration) pulse, as shown in Fig. 3.4(c). The amplitude of the ”hotel” load and the discharging & charging pulses are not usually monotonically increasing in a duty cycle, which made each duty cycle slightly different from others. This pattern of input cycles largely simulates a real-time working condition for an electric locomotive.

The raw time series of input current and output voltage are pre-processed by individual normalization, followed by wavelet-based segmentation based on the time series of output voltage. While segmentation extracts the relevant segments of normalized data based on the information of their frequency contents, normalization involves time translation and (down or up)-scaling of the raw data for conversion into zero-mean unit variance time series as:
where $\mu_N[n]$ is the mean value and $\sigma_N[n]$ is the standard deviation of the time series over a time span of $N$ data points centered at the time index $n$. The experimental data of each duty cycle (i.e., both input current and output voltage) are normalized in this moving average fashion. First, the data set is smoothed under moving average with a shift window of 240s (i.e., 240 consecutive data points). Then, each element of that data set is divided by its own standard deviation. Finally, the normalized data turn out to be zero-mean and unit-variance time series. Figure 3.8(b) and (e) provide two examples of normalized current and voltage data under this proposed shift normalization manner.

Figure 3.6: Wavelet-based segmentation for normalized input-output.

Figure 3.5: Degradation of battery capacity

$$x[n] = \frac{x_{raw}[n] - \mu_N[n]}{\sigma_N[n]}$$  \hspace{1cm} (3.8)

Figure 3.6 presents the segmentation index for small window of normalized input and output as an example. A time series with a segmentation index of 1 is of interest and a time series with a segmentation index of 0 represents a static system that is not useful for feature extraction. Since a "Hotel-Pusles" cycle has length of
120s (i.e., 120 consecutive data points), the dynamic characteristics of the tested battery are identified from the segments of discharging & charging pulses for a time span of 45s in each small cycle. The information-relevant segments are extracted from the normalized time series by using the wavelet-based segmentation algorithm (see Section 3.2). Figure 3.8(c) and (f) give the example result of segmentation based on voltage data for synchronized both input current and output voltage. The data length after wavelet-based segmentation is much shorter than the original data length.

Figure 3.7: Normalized input-output pairs of synchronized data for one duty cycle.

Figure 3.7 exemplifies the typical profiles of normalized (synchronous) and segmented input-output pairs from the ensemble of one duty-cycle data set for SOH identification, where the charging transitions between two consecutive duty cycles are excluded during data pre-processing.

3.5.2 Results and Discussion

The SOH parameter is identified based on the SDF features extracted from pre-processed input-output (i.e., current-voltage) time-series pairs that were collected during the duty cycles. Particularly, SOH identification relies on the divergence between the extracted features (see Eq. (2.23)). The parameters for SDF analysis (see Section 5.3.3) of time series data have been chosen as follows.

- The alphabet size $|\Sigma|$ is chosen to be the same for both dimensions of maximum entropy partitioning in the input-output space. As depicted in Fig. 3.9, partition types 1 and 2 are obtained in Cartesian coordinates and the difference between them lies in the partitioning order of the abscissa (x-axis) and the ordinate (y-axis). Partition types 3 and 4 are obtained in polar coordinates, where magnitude and phase are the used.

- The depth in the $D$-Markov machine is set at $D = 1$, which implies that $|\Sigma| = |Q|$.
The distance function $d(\cdot, \cdot)$ for computation of the feature divergence in Eq. (2.23) for SOH identification is chosen to be the City Block distance (i.e., the absolute sum or 1-norm) for as defined below.

$$d(\hat{\Pi}^k, \hat{\Pi}^0) \triangleq \sum_{j=1}^{\mid Q \mid} \sum_{i=1}^{\mid \Sigma \mid} |\hat{\Pi}^k(i, j) - \hat{\Pi}^0(i, j)|$$

(3.9)

The estimated SOH for real-time data is measured by feature divergence (see Eq. (2.23) and Eq. (3.9)) between the reference feature (i.e., extracted from the
Figure 3.10: Results of SOH estimation with alphabet size $|\Sigma| = 4$ for both input and output

(a) SOH over a time span of 25 duty cycles
(b) Profiles of $\theta$ vs feature divergence for 25 duty cycles
(c) Coefficients of determination for different partition types

Table 3.1: Coefficient of Determination (COD) for SOH estimation at alphabet size $|\Sigma| = |\Sigma_1| \times |\Sigma_2|$
Table 3.1 presents the CODs between normalized feature divergence and the parameter $\theta = 1 - SOH$ for SOH estimation in the analysis of 25 duty periods. These measurements of linearity show that the feature divergences over the battery life are largely linear (i.e., COD values are very close to one) in terms of the corresponding battery SOH parameter $\theta$ for different choices of alphabet size and partition type. Therefore, it is reasonable to conclude that the feature divergence (that is computed in real time) consistently and accurately represents the SOH dynamics during the entire battery life; this representation is valid for different parameter settings of the algorithm.

### 3.5.3 Computational Cost

This subsection presents a statement of the computational costs (i.e., execution time and memory requirement) of the SOH parameter estimation algorithm. In this context, all results have been generated on a single core 3.6 Ghz CPU with 12 GB RAM.

Table 3.2: Execution time and required memory for different analyzed data length

<table>
<thead>
<tr>
<th>Operating duration in analysis</th>
<th>Execution time (sec)</th>
<th>Memory requirement (MB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>5 duty periods</td>
<td>7 ~ 8</td>
<td>10 ~ 13</td>
</tr>
<tr>
<td>10 duty periods</td>
<td>14 ~ 15</td>
<td>15 ~ 18</td>
</tr>
<tr>
<td>15 duty periods</td>
<td>20 ~ 22</td>
<td>21 ~ 23</td>
</tr>
<tr>
<td>20 duty periods</td>
<td>27 ~ 28</td>
<td>26 ~ 30</td>
</tr>
<tr>
<td>25 duty periods</td>
<td>33 ~ 35</td>
<td>33 ~ 36</td>
</tr>
</tbody>
</table>

Table 3.2 presents the execution time and memory requirement of the proposed method for SOH estimation for different durations of operation and analysis. In each case, the CPU time and memory requirement have been obtained for the feature extraction process at all stages of battery aging under the assigned duty periods. The variations in the computational cost for different choices of $|\Sigma| = |\Sigma_1| \times |\Sigma_2|$ and partition types are much less compared to those for lengths of time series data. The execution time and memory requirement of the proposed software for battery SOH estimation is apparently insignificant even on a small computational platform (e.g., commercially available laptop computers).

As pointed out in the previous subsection, the performance of the proposed algorithm could be improved with longer time series data. Therefore, an appropriate choice of the data length depends on the battery system dynamics that also determine the sampling frequency of data acquisition. In real-life applications, such as hybrid electric vehicles, it is expected that larger data sets would be available for SOH estimation within the same testing period, because the sampling frequency of
current/voltage should be larger than 1 Hz, which has been used in the experiments reported this work.

3.6 Summary

This chapter presents an alternative SOH identification method which is built upon the concept of Symbolic Dynamic Filtering (SDF) [90], where time series of synchronized battery input (current) and output (voltage) data have been analyzed for information compression and feature extraction. The procedure of SOH identification relies on the divergence between the features extracted from long-term observations. Performance of the proposed SDF-based SOH identification method has been validated on experimental data of a commercial-scale lead-acid battery for operations over a long time span.
Chapter 4

Battery State-of-Charge Estimation via Recursive Bayes Filter

This chapter addresses estimation of battery state-of-charge (SOC) from the joint perspectives of dynamic data-driven and model-based recursive analyses. The proposed SOC estimation algorithm is built upon the concepts of symbolic time series analysis (STSA) and recursive Bayesian filtering. A special class of Markov models, called $\times D$-Markov (pronounced as cross $D$-Markov) machine, is constructed from a symbolized time-series pair of input current and output voltage. A measurement model of SOC is developed based on the features obtained from the $\times D$-Markov machine. Then, a combination of this measurement model and a low-order model of the SOC process dynamics is used for recursive Bayesian filtering, which is structurally similar to conventional Kalman filtering. The proposed algorithm of SOC estimation has been validated on (approximately periodic) experimental data of (synchronized) current-voltage time series, generated from a commercial-scale lead-acid battery system.

4.1 Introduction

The state-of-charge (SOC) indicates the maximum charge that can be drawn from a battery at a given instant of time; SOC is a critical system state for battery operation from both safety and performance perspectives. Since the state of the art in battery technology does not permit direct measurements of SOC, the challenge is to perform accurate and reliable estimation of SOC in real time based on the time series of input current and output voltage data. A large ensemble of model-based filtering techniques has been reported on the topic of SOC estimation. In general, state-space (e.g., equivalent circuit [102] and electrochemical [103]) models are constructed based on the understanding of physics and chemistry of battery cells. In these models, the input is the charging/discharging current, the output is
the battery voltage, and SOC is a model state.

Li et al. [104] reported a dynamic data-driven method for SOC estimation via Markov modeling on the battery voltage time series. The underlying concept was built upon the theory of symbolic analysis [90, 91] that extracts the dynamic information from symbolized time series. In [40], Li et al. have extended the method by considering synchronized inputs (i.e., current) and outputs (i.e., voltage) together in the setting of a Markov model to incorporate input-adaptive features. Later, Sarkar et al. [84] investigated the cross-dependency between symbolized input strings and symbolized output strings using the $\times D$-Markov (pronounced as cross D-Markov) machine for SOC estimation. Recently, Chattopadhyay et al. [105] constructed a sequential SOC estimation framework based on symbolic Markov modelling [40].

This work is a significant extension of the authors’ earlier work [84, 105] and proposes a framework for recursive estimation of SOC by combining the underlying concepts of $\times D$-Markov machines and recursive Bayesian filtering. Especially, the causal cross-dependence between the battery input current and output voltage is modeled and represented as a $\times D$-Markov machine, which is analogous to transfer function representation in the sense that the dynamic behavior of a system is modeled as the symbolic output in response to the symbolic input via Markov modeling. Then, the SOC measurement model is formulated by a kernel based conditional probability computation of symbolic Markov models, which is constructed from $\times D$-Markov features obtained in the training phase. Finally, a model-based recursive Bayesian estimator is constructed, which filters out the outliers in SOC estimation to reduce the estimation error. The proposed method has been validated using experimental data of a (commercial-scale) lead-acid battery under varying input conditions. Although the work reported in this chapter focuses on lead-acid batteries, the proposed methodology can be extended to other types of batteries (e.g., Li-ion and Ni-MH).

This chapter is organized in five sections including the present one. Section 4.2 proposes a hybrid SOC estimation algorithm as a combination of the data-driven SOC measurement model and model-based recursive Bayesian filtering. Section 4.3 briefly presents the details of the experiment for acquisition of battery system data as well as the results of experimental validation of the proposed estimation algorithm. Finally, this chapter is summarized and concluded in Section 4.4 along with recommendations for future research.

### 4.2 Hybrid recursive SOC estimation

This section presents a data-driven sequential method for estimation of the SOC parameter, where $\times D$-Markov matrices are extracted from (time-synchronized) time series pairs of battery input and output for analytical measurements of SOC. The algorithm of SOC estimation, which combines a state-space model of SOC
dynamics with SOC measurements, consists of the following two steps:

1. Analytical measurements of SOC by statistical conditional generating computation, based on $\times D$-Markov machine features constructed from symbolic time series of battery input and output.


In other words, the SOC state is first predicted from the previously estimated (or initially provided) state by using a dynamic model of SOC, and then the current SOC measurement is used to update the SOC estimate.

Figure 4.1: Schematic description of the hybrid recursive Bayesian filtering SOC estimation algorithm.

4.2.1 Data-driven analytical measurement of SOC

This subsection formulates a data-driven procedure for analytical measurement of SOC via statistical regression in a Bayesian framework. Each data set consists of a (time-synchronized) finite-length time series of input $U(t)$ and output $Y(t)$ with the associated SOC label $x$. A $\times D$-Markov machine is constructed from each data set by the procedure described in Section 2.6.4. Then, the $\times D$-Markov morph matrices are used as features in the algorithm as described below.

Let $M$ data sets be available for training the algorithm, where each data set $i$ consists of a symbolic input string $U_i$, a symbolic output string $Y_i$, the PDFA state string $Q_i$ generated from the symbolic input string $U_i$, the $\times D$-Markov morph matrix $\Pi_i$, and a labeled value $x_i$ of SOC. The structure of cross dependency
between the current (input) and voltage (output) is expected to vary for different values of SOC.

Given the input state string \( Q_i \) and the cross dependency specified by \( \Pi^j \) (generated from the \( j^{th} \) training data), the probability of observing a symbolic output string \( Y_i \) is:

\[
p(Y_i | U_i, \Pi^j) = P(Y_i | Q_i, \Pi^j) = \prod_{k=1}^{L-D+1} \Pi^j(q_k, y_{k+D-1})
\]

(4.1)

where \( L \) is the length of the symbol string and \( D \) is the depth for the Markov modeling.

During the testing phase, it is provided that \( U_k \) and \( Y_k \) are the symbolized time series of the input current \( U(t) \) and output voltage \( Y(t) \) during the \( k^{th} \) time window respectively. The probability of observation \( Y_k \) conditioned on the system state \( x_k \) and input \( U_k \) can be formulated as:

\[
p\left(Y_k | x_k; U_k\right) = \sum_{i=1}^{M} W(x_k, \Pi^i)p(Y_k | U_k, \Pi^i)
\]

(4.2)

where the weighting function \( W \) is formulated as a (Gaussian) radial basis function kernel:

\[
W(x_k, \Pi^i, \sigma) = \frac{\exp\left(\gamma(x_k - x_i)^2\right)}{\sum_{j=1}^{M} \exp\left(\gamma(x_k - x_j)^2\right)}
\]

(4.3)

where \( \gamma < 0 \) is a free parameter and \( x_i \) is the label associated with the \( \times \)-morph matrix \( \Pi^i \).

### 4.2.2 Recursive Bayesian filtering for SOC estimation

This subsection combines the data-driven (analytical) measurement of SOC with its model-based recursive estimation in the sense of Bayesian filtering. For this purpose, a linear state-space model of SOC dynamics is formulated as:

State equation: \( x_k = x_{k-1} - \frac{1}{C_k} u_k + w_k \)

(4.4)

where \( C_k \) is the maximum capacity of the battery during the \( k^{th} \) time window. \( w_k \) is the process noise which is assumed to be drawn from a zero-mean Gaussian distribution with standard deviation of \( q_k \). The \( u_k \) is the depth of discharge (DOD) from the beginning to the end of \( k^{th} \) time window, which is obtained as:

\[
u_k = \int_{T_h}^{T_{k+1}} U(t) dt
\]

(4.5)
Then, the probability of the current true state $x_k$ given the immediately previous one $X_{k-1}$ and the input $u_k$ can be expressed as a Gaussian distribution $N(x_{k-1} - \frac{u_k}{C_k}, q_k^2)$:

$$p(x_k|x_{k-1}; u_k) = f(x_k|x_{k-1} - \frac{u_k}{C_k}, q_k^2) = \frac{1}{q_k\sqrt{2\pi}} \exp\left(-\frac{(x_k - (x_{k-1} - \frac{u_k}{C_k}))^2}{2q_k^2}\right)$$  \hspace{1cm} (4.6)

The algorithm of the recursive Bayesian filter is formulated as follows:

- **Initialization**: the distribution of system state is:

  $$p(x_0|Y_0) = \text{unif}(a, b)$$  \hspace{1cm} (4.7)

  where $a = x_{\text{min}}$ and $b = x_{\text{max}}$ are the minimum and maximum operating range of system state $x$.

- **State prediction for $k \geq 1$**

  $$p\left(x_k|Y_{1:k-1}; \{U_{k-1}, u_k\}\right) = \int_a^b p(x_k|x_{k-1}; u_k)p(x_{k-1}|Y_{1:k-1}; U_{k-1})dx_{k-1}$$  \hspace{1cm} (4.8)

- **State update for $k \geq 1$** The measurement update stage for $(k \geq 1)$ is formulated as:

  $$p\left(x_k|Y_{1:k}; \{U_{k-1:k}, u_k\}\right) = \frac{p(Y_k|x_k; U_k)p(x_k|Y_{1:k-1}; \{U_{k-1}, u_k\})}{p(Y_k|Y_{1:k-1})}$$  \hspace{1cm} (4.9)

  where

  $$p\left(Y_k|Y_{1:k-1}; \{U_{k-1:k}, u_k\}\right) = \int_a^b p(Y_k|x_k; U_k)p(x_k|Y_{1:k-1}; \{U_{k-1}, u_k\})dx_k$$  \hspace{1cm} (4.10)

The optimal state estimation can be achieved by different criterion. For examples:
• Minimum mean-square error (MMSE)
\[ \hat{x}_{k|k}^{\text{MMSE}} = E[x_k|Y_{1:k}] \]
\[ = \int_{a}^{b} x_k p(x_k|Y_{1:k}; \{U_{k-1:k}, u_k\}) dx_k \quad (4.11) \]

• Maximum a-posteriori (MAP)
\[ \hat{x}_{k|k}^{\text{MAP}} = \arg \max_{x_k} p(x_k|Y_{1:k}; \{U_{k-1:k}, u_k\}) \quad (4.12) \]

**Remark 4.2.1.** The measurement model in Eq. (4.2) possibly involves a nonlinear nonparametric discrete-valued conditional probability distribution. Therefore, instead of using the commonly used Kalman filter, the general recursive Bayes filter (RBF) might be the statistically correct way to compute the posterior distribution of the unobserved state [106]. Although the resulting computational cost would be larger to implement this RBF, there are benefits from using such an analytical measurement model as stated at the end of Subsection 4.2.1. Performance comparison of the proposed method and the conventional Kalman filter with a linear (or linearized) model with additive Gaussian measurement noise is recommended as a topic of future research as stated in Chapter 9.

### 4.3 Experimental validation

This section validates the proposed sequential estimation algorithm with an ensemble of experimental data that have been collected from a commercial-scale lead-acid battery.

#### 4.3.1 Experimental Data Collection

The fresh (12V AGM VRLA with 56Ah capacity) lead-acid battery, which has been used in the experiments, was charged/discharged according to the specified input (current) profiles at the room temperature. An ensemble of synchronized time-series of the input charge/discharge current and output voltage responses has been collected at the sampling frequency of 1Hz. Figure 4.2 exhibits typical profiles of long-term experimental data, namely, input current, output voltage, and battery SOC; and Figure 4.3 presents short-term profiles of the corresponding experimental data. The input data set consists of fluctuating (with repeated “Hotel-Pulse” cycles) of battery discharging/charging current, where each individual “Hotel-Pulses” cycle (with a duration of ~ 120s) consists of a “hotel” load (i.e., relatively steady discharge due to “hotel” needs like lighting and other electrical equipments) and a discharge pulse followed by a charge (i.e., regeneration)
pulse. The amplitude of the “hotel” load and the discharging & charging pulses are fluctuating in the experiment, which made each cycle different from others. This pattern of input cycles largely simulates a real-time working condition for an electric vehicle.

In this context, the proposed data-driven estimation algorithm is built upon the concept of recursive Bayesian filtering that makes use of the dynamic information embedded in the system input and output combined with the predictive capability of a dynamic model of SOC. The time series of interest is located at dynamic transient sections, such as a step input in the form of short-duration pulses and their corresponding responses. In Figure 4.3, the highlighted dynamic zones are selected as analysis windows for the data-driven estimation algorithm. The data length for one dynamic zone (i.e., analysis window) is 50 on the fast time-scale in the unit of seconds.
Figure 4.3: Examples of dynamic zone and steady-state zone in the experimental data

4.3.2 Results and Discussion

There are, in total, 25 experiments that have been conducted for this research, where the nominal battery maximum capacity is assumed to be the same for all experiment data. To evaluate the performance robustness of the proposed method, a cross validation process is applied in this work as described below.

For each test run, 5 sets of experimental data are first randomly selected; then, the proposed data-drive regression algorithm is trained by the data from the remaining 20 sets of experimental data. Finally, the performance of the proposed recursive estimation algorithm is tested based on those selected 5 testing data sets. In total, 100 test runs have been conducted for this research.

The parameters for data-driven measurement model and system state-space model are chosen as following:
Figure 4.4: Examples of recursive SOC estimation for different alphabet size $|\Sigma|$.

- The parameter for (Gaussian) radial basis function kernel:
  \[
  \gamma = \frac{1}{2 \ast (0.03)^2}
  \]
  \hfill (4.13)

- The maximum capacity of the battery during the $k^{th}$ window is assigned as the initial battery capacity for all analysis windows:
  \[
  C_k = C_{\text{initial}};
  \]
  \hfill (4.14)

- The standard deviation for process noise $w_k$ is assigned as 10% of the ratio between input $u_k$ and capacity $C_k$:
  \[
  q_k = 0.1 \times \frac{u_k}{C_k}
  \]
  \hfill (4.15)

In this section, we compare the performance between hybrid recursive Bayesian filtering and solo data-driven estimation via $\times$-Markov modeling. The minimum mean-square error (MMSE) estimation is chosen as the criterion for the recursive Bayesian filtering at each step; while the maximum a-posteriori (MAP) estimation is applied for the solo data-driven estimation:

- SOC estimation by Hybrid recursive Bayesian filtering:
  \[
  \hat{x}_{k|k}^{\text{MMSE}} = E[x_k|Y_{1:k}]
  \]
  \hfill (4.16)
• SOC estimation by solo data-driven $\times$-Markov modeling:

$$\hat{x}_k^{\text{MAP}} = \arg\max_{x_k} p(\mathbf{y}_k|x_k; \mathbf{u}_k)$$ (4.17)

Figure 4.4 depicts three typical examples of SOC estimation results under different alphabet size: $|\Sigma| = 4, 6$ and 8. For $|\Sigma| = 4$, the $\times D$-Markov morph-matrix features experience difficulties to capture the dynamic characteristics of the battery in the high range (e.g., $\sim 75\%$ to $80\%$) and low range (e.g., $\sim 60\%$ to $65\%$) of SOC. The execution time of the recursive estimation algorithm for convergence is in the order of one hour, which is in excess of 30 analysis windows. However, as the alphabet size is increased to $|\Sigma| = 6$, the $\times D$-Markov morph-matrix features improve their distinguishing capability at low SOC range (e.g., $\sim 60\%$ to $65\%$), while the convergence time is significantly reduced to about 20 analysis windows on the average. The rationale for this performance enhancement is explained below.

As the alphabet size is modestly increased from $|\Sigma| = 4$ to $|\Sigma| = 6$ and then to $|\Sigma| = 8$, the resulting $\times D$-Markov machines capture more detailed information on the dynamics embedded in the time series; consequently, the corresponding morph-matrix features obtained under different operating conditions are more distinctive (i.e., more apart from each other); the effects are observed in Figure 4.4. However, due to the finite length of the time series (i.e., finite length of the analysis window) under consideration, $|\Sigma|$ cannot be increased beyond a certain limit as explained by Wen et al. [107] and experimentally observed by Mukherjee and Ray [91].

![Figure 4.5: Root-mean-square (RMS) error of SOC estimation for different alphabet size $|\Sigma|$](image)

Figure 4.5 depicts the root-mean-square (RMS) error of SOC estimation for solely data-driven symbolic $\times D$-Markov machine and its combination with model-based recursive Bayes filter. For solely $\times D$-Markov machine, the RMS error is around 3% for the alphabet size $|\Sigma|$ ranging from 4 to 10. Due to the length limit
of the time-series data at hand, the estimation accuracy of the $\times D$-Markov machine approach may not be improved any further by increasing the alphabet size [107]. On the other hand, the RMS error for combined data-driven and model-based recursive Bayes filter is significantly reduced compared with the same time series data, as seen in Figure 4.5.

4.4 Summary

This chapter presents a combination of dynamic data-driven and model-based recursive methods for real-time estimation of a critical battery parameter, namely, state of charge (SOC). The underlying concept is built upon a measurement model and prior history of estimation, which rely on synchronized time series of repeated-pattern and fluctuating input (i.e., charging/discharging current) and output (i.e., battery voltage) data. The SOC estimation algorithm has been validated on experimental data of a commercial-scale lead-acid battery.
Chapter 5

Dynamic Modeling of Combustion Process via Video Data

This chapter proposes a framework for analyzing video of physical processes (i.e., lean and ultra-lean combustors) under a variety of operating conditions. The main challenge here is to develop feature extraction and information compression algorithms with low computational complexity such that they can be applied to real-time analysis of video (e.g., a sequence of image frames) captured by a high-speed camera. In the proposed method, the first stage of the algorithm extracts low-dimensional image features from each frame of the video, which is compressed into a sequence of image features. In the second stage, these image features are mapped to symbols by a partitioning of the feature space, thereby the sequence of images is converted to a sequence of symbols. In the third and final stage, a special class of probabilistic deterministic finite state automata (PDFA), called D-Markov machines, are constructed from the symbol strings to extract pertinent features representing the embedded dynamic characteristics of the physical process. This chapter compares the performance and efficiency of three image feature extraction algorithms: Histogram of Oriented Gradients, Gabor Wavelets, and Fractal Dimension. The k-means clustering algorithm has been used for feature space partitioning. The proposed algorithm has been validated on experimental data in a laboratory environment from a lean, premixed, swirl-stabilized, variable-length combustor with a single fuel-injector.

5.1 Introduction

Lean and ultra-lean combustors have been proposed in industrial research settings and they appear to be very promising for mitigating environmental pollution, especially for reducing Nitrogen Oxides (NOx). However, the main challenges remaining are to prevent thermo-acoustic instabilities and lean blowout (LBO) of
Many factors influence combustion stability, which include variable combustor length, equivalence (i.e., fuel-air) ratio, and inlet flow velocity and pressure. The acoustic oscillations of the flame are the main consequences from unstable combustion, which may lead to serious issues such as inefficient combustion and acoustic resonance. These instabilities are sources of many detrimental effects; for example, LBO for an aircraft gas turbine engine may cause loss of thrust leading to flame shutdown and re-ignition.

This chapter proposes a dynamic data-driven applications system (DDDAS) approach [108] to capture and model the high-speed video of combustion flame. The overall framework is constructed from three underlying stages, namely, feature extraction from single images, dimensional reduction in the feature space, and finally D-Markov modeling of the discretized symbol strings. The symbol strings modeling is based on the reported concepts of symbolic dynamic filtering (SDF) [90], which extracts the dynamic information from a symbolized time series as a probabilistic deterministic finite state automata (PDFA). The proposed method has been validated on experimental data obtained from a lean, premixed, swirl-stabilized, variable length research combustion system with a single fuel-injector.

This chapter is organized in five sections including the present one. Section 5.2 presents three standard computer vision algorithm for feature extraction from image data. Section 5.3 formulate the proposed dimensionality reduction framework for image video data. Section 5.4 proposed a training and testing framework based on the Bayesian network and symbolic dynamic filtering (SDF). Section 5.5 briefly presents the details of the experiment for acquisition of battery system data as well as the results of the proposed algorithm. Finally, this chapter is summarized in Section 5.6.

5.2 Image processing algorithms

This section briefly presents the three standard image processing techniques that are used in this chapter, namely, the Gabor wavelet [109] for multi-resolution analysis, the Histogram of Oriented Gradients (HOG) [110], and box counting for finding the fractal dimension [111]. These three techniques have been chosen because they are capable of extracting different types of image features and yet fit well into the framework of the proposed research. More detailed explanations of these algorithms can be found in the Appendix A.

Gabor wavelets are used for general computer-vision tasks such as HMAX [112]. They correspond to low-level image filters (e.g. edge and texture detectors over multiple orientations), and the collection of these filters defines a filter bank. These low level edge and texture filters are convolved to extract the filter-bank responses. As is done in HMAX and convolutional neural networks [113], max-pooling has
been performed over the responses to ensure: (i) invariance to local image variations and (ii) for dimensionality reduction.

The histogram of oriented gradients (HOG) algorithm [110] was originally developed for pedestrian detection, but ended up setting the state of the art in other more general object recognition tasks. The procedure for finding HOG descriptors consists of the following steps: Step 1: Local gradients are obtained over the image by applying central difference operators; Step 2: The image is partitioned by defining local cells and then the gradients are collected in each cell to define a histogram of the oriented gradients; and Step 3: Local cells are combined to form blocks, followed by normalization of the histogram over the entire block. This final procedure yields better invariance to local changes in lighting, and these blocks are concatenated as the feature descriptor.

The fractal-dimension [111] has been used in the analysis of a wide range of applications, which include stock market analysis, noise corrupted time-series signals, measuring the coast-line of Britain, and analyzing flame geometries. Due to the fractal nature of many real world scenarios, it is important to develop a mathematical formalism that can well define these scenarios. The fractal dimension, often referred to as the box dimension, provides an intuitive concept for understanding non-integer dimensions.

5.3 Symbolic Dynamic Modeling of Video

The sequence of image features, extracted from the video via the algorithms in the previous section are further compressed into symbols in this section. Then from these symbol sequences, $D$-Markov models [90, 91] are constructed to capture the dynamic behavior of flame patterns in the combustion process.

5.3.1 Principal Component Analysis as Dimensionality Reduction

Principal component analysis (PCA) [32] is a standard technique that takes the original data and maps it to a new equivalent space defined in a rotated basis such that the greatest variance of the data is contained along the first basis dimension, the second greatest variance is contained along the second basis dimension, and so on. The transformation is defined by taking the eigenvectors of the covariance matrix. The eigenvalues of the diagonalized covariance matrix tell you how much discriminative information is contained along each dimension, and thus one can use only a subset of the original dimensions while maintaining class separability. More information can be found in Appendix C.
5.3.2 K-Means Clustering for Feature Symbolization

The $k$-means clustering [114, 115, 116] has been used to group the PCA-projected features into $k$ mutually exclusive clusters. After initiating the positions of cluster centroids, each feature is assigned to the closest cluster based on the $l_2$ Euclidean distance. The cluster centroids are then updated by minimizing the sum of the distance measures between every feature and its corresponding cluster centroid. The cluster assignment and distance minimization processes are iteratively repeated until the algorithm converges and there is no further change in assignment of the feature to the clusters. The positions of cluster centroids are initialized by randomizing from the given features. To avoid the local minimum due to a bad initialization, 10 replicates with individual random initial starts have been used in parallel to ensure that the algorithm returns the lowest sum of distance measures. The outputs of $k$-means clustering are symbol strings of cluster indices for the feature sequence, which itself corresponds to the raw image sequence of the combustion video.

5.3.3 Symbolization of Time Series and $D$-Markov Modeling

This subsection briefly describes the underlying concepts of Symbolic Dynamic Filtering (SDF), with a more detailed explanation found in [90]. It is a dynamic-data-driven tool that captures the behavior of (possibly non-linear) dynamical systems from symbol strings, and from this constructs state machines (i.e. probabilistic deterministic finite state automata (PDFA)) [90].

When given a sequence of symbols, the goal of symbolic dynamics is to characterize the statistical transitions between states and symbols, where states are defined as string combinations of symbols. A Markov machine is constructed to find the probability that, given you are in a certain state, what is the probability you will see a certain symbol. A $D$-Markov machine allows the construction of Markov machines where the states can be composed of at most $D$ symbols [90, 91].

![Diagram](image-url)  
**Figure 5.1:** Proposed approach for high-speed video symbolization
5.4 Proposed Approach

This section presents the proposed method for symbolic dynamic video analysis of the flame patterns for a combustion process. First, the framework of symbolic dynamic modeling for video data is summarized. Then, the problem of flame pattern classification for a combustion process is formulated into training and testing phases.

5.4.1 Symbolic Analysis on Video

The computer vision techniques usually deal with static images as opposed to dynamic video data. In order to process the dynamic video data, a discriminative model is constructed over the features identified by static image processing algorithms. Framing the discriminative aspect of the model as probabilistic deterministic finite state automata (PDFA) allows one to use the tools of symbolic dynamics to analyze the temporal aspects of the video.

The general procedure can be described in two stages. In the first stage, as seen in Figure 5.1, an image processing technique is used to extract feature descriptors on a frame-by-frame basis, and then if necessary the dimension is further reduced. In this example, HOG features were extracted over raw high-speed combustion images and the dimensions were reduced using PCA.

Figure 5.1 also describes the second stage, in which this lower dimensional feature space is partitioned by clustering the lower dimensional features using a clustering technique (such as k-means or a Gaussian Mixture Model [32]). Each partition in the model is identified with a unique symbol, and since each feature is identified with a unique partition, the feature can be identified with a unique symbol. Video is a sequence of images, that within this framework is mapped to a sequence of features, which itself is mapped to a sequence of symbols. This symbol sequence is like any other symbol sequence [90], and can be analyzed using the tools of symbolic dynamics to define a PDFA.

In this chapter, this method of video processing has been used to study combustion systems equipped with a high-speed camera. Flames at different operating conditions exhibit different visual effects, such as color, curvature, flickering, regional extinction & re-ignition, and plumes. These features of flame geometry are mapped as symbols to generate a symbol sequence, which is useful for understanding and characterizing the flame states. Currently high-speed video of flames are largely studied by a human on a frame-by-frame basis, such as in [117]. Instead, analysis of the video by the proposed algorithm would allow the analysis of these transitions on a much larger scale and thereby develop more statistically robust results. For example, flame instability is qualitatively described as the sequential extinction and re-ignition of the flame at various spatial locations. By identifying each symbol as a physically interpretable flame state, one can quite naturally study
flame state transitions in a statistical sense to quantitatively describe sequential extinction and re-ignition at various spatial locations.

Figure 5.2: Flow chart of the proposed method training and testing phases

5.4.2 Formulation of Training & Testing Procedure

This subsection presents a general framework of the online pattern classification problem in the symbolic domain; the patterns are constructed from symbol strings as probabilistic deterministic finite state automata (PDFA) with (possibly) diverse algebraic structures. The uncertainties due to the finite length of the symbol string in both training and testing phases, which would influence the final classification decision, are formulated as Dirichlet and multinomial distributions. While the details are reported by Wen et al. [107], the essential concepts and procedures of the proposed method are outlined below.

The sequential online identification method consists of an off-line training phase and an online testing phase, as depicted in Figure 5.2. All training and testing data sets are symbolized independently and, in the same way, the classification process is conducted based on the generated symbol strings. In the training phase, probabilistic deterministic finite state automata (PDFA) $K^i$, $i = 1, 2, \cdots, L$, are constructed from the symbol strings $S^i$ for each of the $L$ a priori specified classes. Then, the probability density function $f_{\Pi_i|S^i}$ of the emission probability matrix $\Pi^i$, conditioned on the symbol string $S^i$, is fitted as a Dirichlet distribution [107] based on the emission counts $\{N^i_{mn}\}$. In the testing phase, a symbol string $\tilde{S}$ of finite length is fitted into every PDFA $K^i$ and the associated emission counts $\{\tilde{N}^i_{mn}\}$ are computed accordingly for each class. Then, the probability $\Pr(\tilde{N}^i|\Pi^i)$ of observing the test symbol string $\tilde{S}$, conditioned on a given state $Q^i$ and emission matrix $\Pi^i$, is modeled as a multinomial distribution [107]. Thus, given a symbol string $S^i$ in the training phase, the probability $\Pr(S^i|S^i)$ of observing a symbol string $\tilde{S}$ in the testing phase is computed by combining $\Pr(\tilde{N}^i|\Pi^i)$ and $f_{\Pi_i|S^i}$. A classification decision is made by choosing the class, corresponding to the maximum of posterior probability $\Pr(C_i|\tilde{S})$ [118].
5.5 Experimental Validation

The proposed method has been applied to the high speed video collected from an experimental combustor. This section briefly presents the experimental apparatus for data collection and the results of flame pattern classification under different working conditions.

![Schematic view of the combustor apparatus (fluid flow from left to right)](image)

5.5.1 Description of the Experimental Apparatus

This section describes the experimental apparatus and the data analyzed in this chapter. The apparatus is a laboratory-scale combustor that is composed of the following major components: an inlet, an injector, a variable length combustion chamber and an exhaust system, as shown in Figure 5.3. Compressed air is supplied to the combustor at 180 psig which is set by a dome pressure regulator, passing through a filter to remove any particulates and water, and finally preheated to 250°C by an 88 kW electric heater. The fuel used is natural gas with a composition of approximately 95% methane and is compressed to 200 psig before being delivered to the system. The air and natural gas flow rates are measured with thermal mass flow meters. Equivalence ratios and mean inlet velocities are controlled by adjusting flow rates with needle valves. Fuel is injected near the swirler for technically pre-mixed experiments, which introduces the fluctuation of the pressure and velocity to the combustion process.

The experiments have been performed with self-excited flames, where the length of the combustor was changed to excite unstable combustion modes. Instabilities in flame dynamics are caused by couplings of unsteady heat release rates, system acoustics, inlet velocities and equivalence ratios. The choke plate creates a large pressure drop which helps attenuate pressure fluctuations from propagating upstream from the combustor to the injectors, thus helps minimize equivalence ratio fluctuations.
The combustion chamber has a quartz window so that the flame can be viewed during operation. The flame video records CH* chemiluminescence emissions by an intensified charged-coupled device (ICCD camera) with a 430±5 nm narrow bandpass filter (which corresponds with the CH* chemiluminescence emissions).

In these experiments, the flame has been self-excited and the inlet velocity was set to 40m/s. The experiments were conducted with various combinations of the apparatus settings. For example the combustor length was set to 25, 33, 35, 38, 41, 45 and 58 inches which correspond to different oscillation modes, and the equivalence ratio to 0.55 and 0.60.

5.5.2 Experimental Results and Discussion

In these experiments, real-time classification of flame patterns under different combustion working conditions is based on the PDFA features (see Section 5.3) extracted from symbolic modeling of the collected video. The parameters for feature extraction and symbolization of video data have been chosen as follows.
• Each image is initially cropped around the flame and downscaled by a factor of 2 so that the final size is $240 \times 300$.

• For HOG, the image is divided into cells of size $21 \times 64$, a block size of $2 \times 2$, and 9 gradients are used.

• For Gabor wavelets, the filter bank is taken to be kernels of size 1/5 the dimensions of the image, $\lambda \in \{1/2, 1/4\}, \theta \in \{0, \pi/4, \pi/2, 3\pi/4\}, \psi \in \{0\}, \sigma \in \{1\}, \gamma \in \{1\}$ and max pooling was performed over the disjoint regions of size 1/10 the filter response size. Greek letters are defined in Appendix 1.

• For Principal Component Analysis (PCA), the top three principal components (that capture 95% of total signal energy) are picked for dimensional reduction of the space of features extracted from images via multiple image processing algorithms.

• For $k$-means clustering in PCA feature symbolization, the number of cluster is chosen to be $k = 5$.

• The size of the symbol alphabet is chosen to be $|\Sigma_1| = 5$, corresponding to number of clusters from the $k$-means clustering algorithm.

• The depth in the $D$-Markov machine is set at $D = 1$.

• The length of both training and testing data is 100.

Figure 5.4 depicts four testing examples for different combustor lengths under the equivalence ratio of 0.55 with fractal dimension measurement. The results demonstrate that the algorithm is able to correctly identify the combustor length after observing test data that are of very short length, i.e. within 80 image frames.

Figure 5.5: Posterior probabilities as functions of test data length at different equivalence ratios and combustor lengths

63
Two plots in Figure 5.5 depict the profiles of posterior probabilities as functions of the data length for different equivalence ratios under combustor lengths of 25 inches and 35 inches, using HOG features. The results show that while the proposed method is sensitive to short data lengths (e.g. 20 frames and below), it is stable for data lengths over 30 for different equivalence ratios.

![Figure 5.6: Average misclassification rate for different combustor lengths](image)

Each of the two plots in Figure 5.6 depict the average misclassification rate for different combustor lengths of three image processing algorithms, HOG, Hausdorff, and Gabor. Two classification cases have been conducted separately, where the first case at equivalence ratio of 0.55 consists of four classes with respective combustor lengths at 25, 35, 41, 58 inches and the second case at equivalence ratio of 0.6 consists of five classes with combustor lengths at 25, 33, 38, 45, 58 inches. For each class in both cases, there is one training data set of 100 image frames and four testing data sets with the same length (i.e., 100 image frames) for each class. Each profile of misclassification rate, presented in Figure 5.6, has been generated as the average of the results from these two cases (i.e., equivalence ratios, 0.55 and 0.60). The results show that the algorithm with symbolic dynamic analysis using HOG feature descriptors provide the best performance when compared to Hausdorff dimension measurements and Gabor filters. Nevertheless, the error for all three image processing algorithms converges within 100 image frames.

Figure 5.7 depicts the average misclassification rate for different equivalence ratios for the three image processing algorithms, HOG, Hausdorff, and Gabor. The data have been collected at combustor lengths of 25 inches and 58 inches for each of the equivalence ratios of 0.55 and 0.60. Similar to what is done for Figure 5.6, there is one training data set of 100 image frames and four testing data sets with the same length of 100 image frames for each class. The misclassification rate presented in the profile of Figure 5.7 is the average of the results from two combustor lengths. The results shows that symbolic dynamic analysis with both HOG and Fractal dimension measurement are highly sensitive to flame patterns.
at different equivalence ratios. Although the error converges to zero for both, the convergence time for HOG to zero is shorter. On the other hand, while Gabor filter has limited ability for distinguishing the flame pattern of equivalence ratio, the error rate of the Gabor filter is much higher than that of the other two algorithms within 100 frames.

5.6 Summary

This chapter has developed a video processing framework for analyzing high-speed physical processes. There are three stages of dimensionality reduction in that processing: 1) feature extraction from raw image data via standard computer vision techniques; 2) a further dimensionality reduction in feature space via linear Principal Component Analysis (PCA); 3) symbolization on low dimensional feature via K-means clustering. Finally, a symbolic Markov model is constructed from the symbolized image video to extract dynamic temporal pattern embedded in the symbol sequence. The underlying algorithms have been validated on the experimental data generated from a laboratory-scale combustor apparatus.
This chapter addresses the problem of target detection in dynamic environments in a semi-supervised data-driven setting with low-cost passive sensors. A key challenge here is to simultaneously achieve high probabilities of correct detection with low probabilities of false alarm under the constraints of limited computation and communication resources. In general, the changes in a dynamic environment may significantly affect the performance of target detection due to limited training scenarios and the assumptions made on signal behavior under a static environment. To this end, an algorithm of binary hypothesis testing is proposed based on clustering of features extracted from multiple sensors that may observe the target. First, the features are extracted individually from time-series signals of different sensors by using a recently reported feature extraction tool, called symbolic dynamic filtering (SDF). Then, these features are grouped as clusters in the feature space to evaluate homogeneity of the sensor responses. Finally, a decision for target detection is made based on the distance measurements between pairs of sensor clusters. The proposed procedure has been experimentally validated in a laboratory setting for mobile target detection. In the experiments, multiple homogeneous infrared sensors have been used with different orientations in the presence of changing ambient illumination intensities. The experimental results show that the proposed target detection procedure with feature-level sensor fusion is robust and that it outperforms those with decision-level and data-level sensor fusion.

6.1 Introduction

Detection of moving targets in uncertain and dynamic environments is of utmost importance for intelligence, surveillance and reconnaissance (ISR) systems (e.g., Secure Border Initiative (SBI) by the Department of Homeland Security). To
this end, large distributed networks of passive sensors are often attractive because they can provide coverage of wide areas at a moderate running cost. Every sensor in the network is provided with communication and computing capabilities for the collective intelligent behavior. A key requirement for cost effective operation of such distributed sensor networks is to maintain reliable sensing performance, while limiting the amount of communications required for reliable decision fusion and situation awareness across the network. In addition, limited computational capabilities of the individual sensors require low-complexity data-processing algorithms for information extraction from the time series of sensed signals. Decision-making with passive sensing in a dynamic environment is an algorithmically challenging problem as these sensors (e.g. acoustics, seismic, and infrared) are often sensitive to the environmental uncertainties and these effects, in general, are difficult to model using the fundamental principles of physics.

While maintaining reliable sensing performance, the following research challenges need to be simultaneously addressed as key requirements for operating distributed sensor networks in a cost-effective way,

1. **Constrained communication at sensing nodes**, which would require a feasible framework for information compression and distributed information fusion to reduce the communication overhead.

2. **Limited computational capabilities at sensing nodes**, which would require low-complexity algorithms for information extraction and decision analysis.

3. **Operation under dynamic environments**, which may significantly compromise the performance of supervised algorithms for object and situation assessment.

A distributed framework for feature-level fusion of sensor time series has been proposed for target detection in dynamic environments [119]; the underlying theory is built upon a recently reported tool of feature extraction, called symbolic dynamic filtering (SDF) [90][91]. The compressed information in the extracted features is then clustered in an unsupervised fashion for target detection via statistical hypothesis testing. Because of their different spatial locations and orientations, the sensors have different degrees of influence from an event (e.g., a mobile target) of interest. The key idea here is to capture different levels of the target’s influence on the individual sensors for fusion by clustering of the collected information. A decision rule is then formulated for target detection based on the behavior of a pair of clusters in the feature space. Note that the clusters are formed in the feature space and do not have any direct relation with the _cluster head_ mentioned earlier.

The proposed approach of feature-level collaborative decision making in the presence of dynamic background environment has been experimental validation of multi-sensor target detection problem in a laboratory setting. The results suggest that the proposed fusion of information at the feature level is robust to a dynamic
training environment and performs significantly better than an equivalent decision-
level and data-level fusion schemes.

The chapter is organized in seven sections including the current section. Section 6.2 presents the statement of the addressed problem along with a list of major assumptions. Section 6.3 elaborates the proposed approach adopted in this chapter. Section 6.4 formulates the target detection problem as a hypothesis testing framework. Section 6.5 briefly presents the details of the experimental facility and procedure. Section 6.6 presents the results of experimental validation of the proposed concept. Finally the chapter is summarized in Section 6.7.

6.2 Problem Statement

This section formulates the problem of target detection in a dynamic environment. The presence of a target is determined by analyzing the ensemble of time series data generated from a number of sensors on a network. To this end, a set of sensors is deployed in a local region of the sensor network. The observation ensemble \( Y = \{Y_1, Y_2, \ldots, Y_n\} \) represents the set of time-series data generated by the ensemble of sensors \( L = \{L_1, L_2, \ldots, L_n\} \).

Target detection in this chapter is based on a binary hypothesis test of \( H_0 \) versus \( H_1 \) that represent the hypotheses corresponding to “environmental disturbances only” (i.e., absence of a target) and “presence of a target,” respectively. That is, \( H_0 \) is referred to as the null hypothesis and \( H_1 \) as the alternate hypothesis. In this setting, the problem of target detection in a dynamic environment is to find a decision rule for \( H_0 \) versus \( H_1 \) by a synergistic combination of the data collected by the sensor set \( L \) under the following assumptions.

1. The sensors are homogeneous, i.e., all sensors are of the same modality.

2. Even though the sensors have the same modality, their deployment is heterogeneous in the sense that they are not co-located and that their orientations are different; therefore, different sensors may yield different outputs in the presence of a single target.

3. Within a given time span of observation, there is at most one (moving) target. In other words, the observed time series outputs of the sensors correspond to the same event (i.e., the same target if it is present).

4. The time scale of environmental changes is largely similar to that of event occurrence (e.g., target motion). In other words, the environment is dynamically changing during the observation of a local event.

The assumptions 1 to 4, stated above, are often encountered in distributed sensing applications. The usage of homogeneous sensors is cost-effective and placing them non-co-located with different orientations provides heterogeneity that
allows a variety of views of the target and environment. Furthermore, the sensors are usually looking for hard-to-find targets, which generally means only one will appear in a field-of-view within a short time span, or else the field of view would be too large to detect a faint target. Finally, the time scale of encountering hard-to-find targets is often much slower than that of environmental changes; if this is not the case, then standard adaptive filtering techniques can usually suppress the environmental effects and make target detection an obvious process. Thus, the above assumptions are consequences of phenomena that may naturally occur at a node in a sensor network node in practical applications.

For the constraints of low-complexity fusion for decision-making under requirements to limit communication, this problem requires a collective assessment of the features extracted from the individual sensors for a collaborative decision rule. More importantly, the algorithms must make a trade-off between the complexity of collaborative decision and the accuracy of detection of the events of interest.

6.3 Proposed Technical Approach

For collaborative decision-making, it is necessary to construct a framework for information fusion at appropriate levels of granularity or details. From this perspective, the decisions are mostly characterized in terms of information levels at which the fusion takes place. Based on the level of information details, sensor fusion models are classified as the well known triplet: data-level, feature-level and decision-level [120].

Even though data-level fusion entails maximum details, communication of raw data (i.e., symbols in the proposed approach) may require tight clock synchronization in individual network nodes and high communication overhead, which are expensive to implement. On the other hand, decision-level fusion requires relatively less communication overhead at the expense of loss of vital information, which may lead to incorrect interpretations in a dynamic operating environment. Therefore, the focus of this work is to formulate an algorithm for feature-level information fusion for target detection.

Each time series $Y_i$ of the observation ensemble $Y$ is first individually processed (at the sensor site) and then compressed into a low-dimensional feature. The symbolic dynamics-based feature extraction method is used to create generative models of the sensed data and also to serve as compact representations of the events that generate the data. Hence, each time-series $Y_i$ for the sensor $L_i$ is now statistically represented by a unique PDFA.

The set of PDFA, corresponding to the set of sensors $L$, is denoted as $\mathbb{G} \equiv \{G_1, G_2, \ldots, G_n\}$. The cardinality of the set of states in each PDFA $G_i$ in $\mathbb{G}$ is constrained to be identical (so that they can be compared in the same space). This is further facilitated by having an identical alphabet size $|\Sigma|$ for all PDFA $G_i \in \mathbb{G}$. 69
In the rest of this section, we present three different methods that will be used to investigate the structure of the PDFA set $G$. Based on the structure of $G$ in the environment-only and target-present scenarios, a decision rule is made for target detection. While the individual sensors in a local group have different locations and orientations and thus are subjected to heterogeneous excitation due to a local event of interest (e.g., a single target), distributed dynamics of the environmental changes will have relatively more homogeneous (i.e., less heterogeneous) effects on these sensors. Therefore, the PDFA $G_i$’s in the set $G$ will have largely similar algebraic structures (i.e., the $\delta$ maps and $\hat{\Pi}$ matrices for PDFA $G_i$’s) under the null hypothesis $H_0$ (i.e., the absence of a target in the environment-only scenario). In contrast, the algebraic structures of the individual PDFA $G_i$’s will be different under the alternate hypothesis $H_1$ (i.e., the presence of a target) so that the PDFA set $G$ can be grouped into two structurally different clusters. The classification boundaries learned during training, however, could yield misleading interpretations during testing if a PDFA representing a dynamic environment with no target has an algebraic structure similar to a PDFA representing a target scenario.

![Feature Space of all sensors](image)

Figure 6.1: Feature-space illustration in a dynamic background environment

Figure 6.1 presents a qualitative illustration of the feature evolution in a dynamic environment for a small sensor network. Sensors with local environment-only information (i.e., in the absence of a target) should have similar features at any particular instant of time; however, this information at a given location might be different at two different time instants as well as at different locations at the same time instant. Hence, when compared in the feature space, the sensors with environment-only information should be lumped together as a single cluster. If a target is present, response of sensors and thus, the features extracted from the same sensors with target information should be significantly different from the features of sensors with environment-only information.
6.4 Hypothesis Testing Formulation

The two most dissimilar clusters in the PDFA set $G$ are obtained for each clustering algorithm; for example, see the two clusters at the top of the hierarchical trees in Figure B.1. A decision on the presence of target is made based on the divergence (or gap) between these two clusters. In the presence of a target, the PDFA set $G$ should be more heterogeneously partitioned into two clusters, one carrying largely the information on the environment-only scenario and the other carrying largely the information on the target-present scenario. It is noted that sensors with environment-only information are always present because the presence of a target is spatially sparse. For the environment-only scenario, even the two maximally separated clusters (e.g., the ones with maximum cluster divergence in the hierarchical cluster tree) should be close to each other in comparison to the target-present scenario. Hence, the maximum cluster divergence for the cluster set $C_1$ would be higher as compared to that for the set $C_0$, where the subscripts 0 and 1 represent the null hypothesis $H_0$ of environment-only (i.e., no target) scenario and the alternate hypothesis $H_1$ of target-present scenario, respectively. It is concluded from these observations that the maximum cluster divergence, $\Delta(C_r, C_s)$, which is achieved for the top two clusters in the hierarchical clustering tree, should be strongly influenced by the presence or absence of a target. Spatial sparsity of the event implies that the cluster divergence should be larger in the presence of a target, i.e.,

$$\max_{C_r, C_s \in C_1} \Delta(C_r, C_s) > \max_{C_r, C_s \in C_0} \Delta(C_r, C_s) \quad (6.1)$$

A decision for target presence could be made with appropriate boundaries to distinguish the cluster gap statistics in the absence and presence of targets. To this end, the distribution of cluster divergence $\Delta(C_r, C_s)$ (e.g., divergence between the two clusters as seen at the top of the hierarchical tree in Figure B.1) for each hypothesis could be obtained during a training process. Then, the target detection problem is formulated as a binary hypothesis test in terms of the hypothesis pair as

$$\begin{cases} 
H_0 : X \sim P_0 \\
H_1 : X \sim P_1
\end{cases} \quad (6.2)$$

where $X$ is the range space of the random variable representing the divergence between the two clusters in the PDFA set $G$; and the probability measures $P_0$ and $P_1$ represent the cluster divergence under the null hypothesis $H_0$ (i.e., environment-only scenario) and the alternate hypotheses $H_1$ (i.e., target-present scenario), respectively. Decision boundaries can then be obtained by choosing a threshold $\eta$ on the likelihood ratio $\Lambda$, which is obtained for Eq. (6.2) as follows.

$$\Lambda(x) = \frac{dP_1(x)}{dP_0(x)} \quad \text{for} \ x \in X \quad (6.3)$$
where it is assumed that the probability measure $\mathcal{P}_1$ is absolutely continuous [60] with respect to the probability measure $\mathcal{P}_0$, i.e., for any any event $E$ belonging to the $\sigma$-algebra of $X$, if $\mathcal{P}_0(E) = 0$, then it is implied that $\mathcal{P}_1(E) = 0$. Under this condition, the likelihood ratio $\Lambda$ in Eq. (6.3) is the Radon-Nikodym derivative of $\mathcal{P}_1$ with respect to $\mathcal{P}_0$ and is denoted as $\frac{d\mathcal{P}_1}{d\mathcal{P}_0}$.

Sensor networks are designed to satisfy certain performance requirements (e.g., not exceeding a maximum allowable false alarm probability). Under such constraints, the Neyman-Pearson hypothesis testing procedure [60] maximizes the probability of correct detection $P_D$ while the false alarm probabilities $P_F$ are not allowed to exceed a specified bound $\alpha$, where $\alpha \in (0, 1)$ is called the significance level. From this perspective, an appropriate choice of the decision rule $\delta$ is made to maximize the detection probability as

$$\max_{\delta} P_D(\delta) \text{ subject to } P_F(\delta) \leq \alpha \quad (6.4)$$

A threshold $\eta$ for rejecting the null hypothesis $H_0$ in favor of the alternate hypothesis $H_1$ could be identified by using the following equation.

$$\mathcal{P}_0(\{x \in X : \Lambda(x) > \eta\}) = \alpha \quad (6.5)$$

It follows from Eq. (6.4), Eq. (6.5) and Neyman-Pearson lemma [60] that there exists a unique optimal decision rule $\delta^*$ with an associated threshold parameter $\eta > 0$, which is dependent on the significance level $\alpha$, such that

$$\delta^*(x) = \begin{cases} 
\text{Select } H_1 & \text{if } \Lambda(x) > \eta(\alpha) \\
\text{Select } H_0 & \text{if } \Lambda(x) \leq \eta(\alpha)
\end{cases} \quad \text{for almost all } x \in X \quad (6.6)$$

6.5 Description of the Experiments

The experimental apparatus consists of two computer-instrumented and computer-controlled Khepera III mobile robots [121]: one serves as a single moving target and the other serves as a stationary multi-sensor network with 9 infrared sensors and a pair of navigational sensors. The dynamic environment and the associated disturbances are emulated as variations in the daylight intensities on partially cloudy days.

The experimental apparatus consists of a small sensor network, a moving target, and ambient light. The sensor network is a ring of 9 TCRT5000™ infrared sensors. A two computer-instrumented and computer-controlled Khepera III mobile robots [121] serves as a single moving target. the dynamic environment and the associated disturbances are emulated as variations in the daylight intensities on partially cloudy days.

These sensors embed an infrared light emitter and a receiver. The current experiments make use of the ambient light measurements and the performance of
environmental surveillance is inherently dependent on the maximum sensor updating rates. The update time between measurements of all 9 sensors is 33 ms. During this 33 ms interval, the 9 sensors are read in a sequential way every 3 ms. In this experimental apparatus, the central workstation is linked with sensor nodes via Bluetooth. After receiving each new measurement, the station converts and stores the sensor readings before collecting new data. Due to the processing and communication delays, the average interval between two readings is $\sim 65.3$ ms with standard deviation of 5.2 ms, which makes the average updating frequency as $\sim 18.5$ Hz with standard deviation of $\sim 2.9$ Hz.

This chapter considers two types of scenarios for multi-sensor target detection under dynamic environments. Figure 6.2 depicts the layout of experimentation. The sensor network is placed at the center of a square room in which only one wall has open windows that are exposed to the sun. During the experiments, the moving target travels at a constant speed in straight lines between the sensor network and the ambient light source. The infrared readings of sensors oriented toward the moving target are subjected to disturbances when the target is moving in and out, which causes intermittent blocking of the ambient light source. Figure 6.3 presents examples of sensor readings for both environment-only and target-present scenarios.

![Figure 6.2: Experimental setup for target detection in a dynamic environment](image)

All experiments have been conducted during days under partially cloudy conditions, where the sunlight is intermittently blocked by clouds. This situation affects the readings of infrared sensors, as depicted in Figure 6.3, specially for those sensors that are oriented toward the windows. Sensor nos. 3, 4, and 5 (that are subjected to stronger ambient illumination) and the remaining sensors, i.e., nos. 1, 2, 6, 7, 8 and 9 (that are subjected to weaker ambient illumination) are respectively grouped together as two single clusters (see the top rows in two plates of
6.6 Results and Discussion

This section presents the experimental results of multi-sensor fusion at both decision and feature levels for target detection in a dynamic environment. For decision-level fusion, decisions of target detection are made by voting based on the time series of individual sensors. On the other hand, target detection by feature-level fusion is based on the hypothesis testing of the proposed clustering algorithms. Features are generated from sensor time series based on Symbolic Dynamic Filtering (see Algorithm 2) with the choice of alphabet size $|\Sigma| = 10$ and PDFA model.
depth (i.e., memory) $D = 1$.

Time series data, used for target detection, are first normalized by subtracting the mean and then dividing by the standard deviation; this step corresponds to bias removal and variance normalization. This step ensures that the algorithm is not just picking up spikes in mean or variance for decision making. For robust performance evaluation of both decision-level and feature-level fusion, cross validation is performed by using multiple combinations of training and test data. For each iteration, the experimental data of both environment-only and target-present cases are randomly divided into training and test data sets: 80% of each scenario (i.e., 40 sets) are assigned to the training set while the remaining 20% (i.e., 10 sets) are assigned to the test set. The results are generated as the average over 100 such combinations of training and test sets.

In this chapter, two different pattern classifiers have been used: Support Vector Machine (SVM) and k-Nearest Neighbors (k-NN) that are among the well-known and most frequently used standard tools [32]. The parameter of neighborhood size for k-NN algorithm is chosen as $k = 3$; and the linear kernel function is used for SVM classification. The results obtained by usage of these two classifiers could be different due to the inherent structure of the extracted features.

### 6.6.1 Decision-level Fusion

Decision-level fusion consists of learning and test phases. During the learning phase, each sensor is trained by features extracted from labeled training data. In the testing phase, each sensor reports the results obtained from pattern classification of the features extracted from its test readings. Accordingly, a central workstation declares a positive target detection if the total number of sensors reporting the presence of a target satisfies the detection criterion.

<table>
<thead>
<tr>
<th>$N_{tg} \geq 1$</th>
<th>Target</th>
<th>No Target</th>
<th>Target</th>
<th>No Target</th>
</tr>
</thead>
<tbody>
<tr>
<td>Target</td>
<td>87.4%</td>
<td>12.6%</td>
<td>92.0%</td>
<td>8.0%</td>
</tr>
<tr>
<td>No Target</td>
<td>29.2%</td>
<td>70.8%</td>
<td>50.8%</td>
<td>49.2%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$N_{tg} \geq 2$</th>
<th>Target</th>
<th>No Target</th>
<th>Target</th>
<th>No Target</th>
</tr>
</thead>
<tbody>
<tr>
<td>Target</td>
<td>64.9%</td>
<td>35.1%</td>
<td>63.8%</td>
<td>36.2%</td>
</tr>
<tr>
<td>No Target</td>
<td>16.6%</td>
<td>83.4%</td>
<td>30.8%</td>
<td>69.2%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$N_{tg} \geq 3$</th>
<th>Target</th>
<th>No Target</th>
<th>Target</th>
<th>No Target</th>
</tr>
</thead>
<tbody>
<tr>
<td>Target</td>
<td>35.0%</td>
<td>65.0%</td>
<td>26.0%</td>
<td>74.0%</td>
</tr>
<tr>
<td>No Target</td>
<td>9.8%</td>
<td>90.2%</td>
<td>19.4%</td>
<td>80.6%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$N_{tg} \geq 4$</th>
<th>Target</th>
<th>No Target</th>
<th>Target</th>
<th>No Target</th>
</tr>
</thead>
<tbody>
<tr>
<td>Target</td>
<td>1.2%</td>
<td>98.8%</td>
<td>3.1%</td>
<td>96.9%</td>
</tr>
<tr>
<td>No Target</td>
<td>5.2%</td>
<td>94.8%</td>
<td>14.2%</td>
<td>85.8%</td>
</tr>
</tbody>
</table>
The results of target detection with decision-level fusion and using SVM and k-NN are shown as confusion matrices in Table 6.1, which shows that both classifiers yield decreasing false alarm rates as the requirement on the total number \( N_{tg} \) of sensors reporting the presence of a target is increased. However, the positive detection is severely compromised when requirement on \( N_{tg} \) is greater than 2. As the requirement for correct detection becomes more critical (e.g., \( N_{tg} \geq 4 \)), the detection performance suffers from low successful detection, even though the zero false alarm rate is achieved. The target detection problem, addressed in this chapter, is not sensitive to these two different classifiers, SVM and k-NN. It appears that the poor performance of decision-level fusion is not dependent on the choice of classifiers.

### 6.6.2 Data-level Fusion

This subsection presents the technical approach and results of a data-level fusion scheme for target detection. The data sets used in the training and testing phases are the same as those used for decision-level fusion. However, features are extracted from the ensemble of all sensors instead of being generated individually.

![Symbolization procedure for multi-dimensional time series](image-url)

**Figure 6.4:** Symbolization procedure for multi-dimensional time series

To apply feature extraction on the time series of multiple sensor nodes, a multi-dimensional symbolization method has been developed as depicted in Figure 6.4. The steps of the feature extraction algorithm are as follows.

To apply feature extraction on the time series of multiple sensor nodes, a multi-dimensional symbolization method has been developed. The steps of the feature extraction algorithm are as follows.
• Step 1: Preprocessing – the time series data of each sensor is transformed as a zero-mean and unit-variance manner (i.e., subtract its mean and then divide by its standard deviation).

• Step 2: Maximum Entropy Partitioning (MEP) (Section 2.6.2 – individual processed time series data are partitioned to obtain the symbol sequence for each sensor.

• Step 3: Application of Principal Component Analysis (PCA) [32] – the time series data from multiple sensors are treated as a multi-dimensional observation vector that is decomposed into linearly independent components using PCA.

• Step 4: k-means clustering [32] – applied on the reduced-dimension PCA feature vectors based on a pre-defined set of clusters.

The symbol sequences are compressed representations of the information contents in the ensemble of time series from all sensor nodes, where the PCA serves the role of dimension reduction. In the analysis based on the original high-dimensional time series ensemble over all experimental data sets, top three principal components are chosen for PCA; these three principal components contribute to more than 95% of the variance (i.e., total energy of all principal components).

Table 6.2: Average confusion matrices of cross validation for data-level target detection with # of clusters \( N_C = 3 \)

<table>
<thead>
<tr>
<th>Event Description</th>
<th>( k)-NN</th>
<th>SVM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Target Present</td>
<td>Target No Target</td>
</tr>
<tr>
<td>Target Present</td>
<td>90.64%</td>
<td>9.36%</td>
</tr>
<tr>
<td>No Target</td>
<td>10.56%</td>
<td>89.44%</td>
</tr>
</tbody>
</table>

Figure 6.5: Distribution of cluster distances for environment-only \( (H_0) \) and target-present \( (H_1) \) scenarios in the feature space with cosine distance (similarity).
A major role of data-level fusion is to obtain the correlation among time series data from different sensor nodes in two different scenarios: no target and target present. The data-level fusion yields better performance relative to the decision-level fusion, because more information is available. In data-level fusion, there are three information compression steps during the symbolization process: partition for individual time series from each sensor, projection on top PCA principle components, and final symbolization from clustering of PCA feature sequence. This information compression could be helpful for communication and computation efficiency, and the reported results are promising for this particular application of target detection under dynamic environments.

The results of target detection have been optimized with appropriate choices of cluster number in the k-means clustering algorithm for PCA feature sequence. In this chapter, the results are obtained with the number of cluster chosen to be 3. Similar to decision-level fusion, two pattern classification tools, namely, SVM and k-NN [32], are applied and Table 6.2 lists the results data-level fusion.

### 6.6.3 Feature-level Fusion

This subsection presents performance evaluation of target detection by feature-level fusion using the three different techniques of clustering, namely, k-means, agglomerative hierarchical, and spectral, discussed earlier. In each of these approaches, a decision on target detection is made based on a threshold on the divergence (or gap) between the two most dissimilar (i.e., maximally separated) clusters in the PDFA set G. Since the attributes (i.e., features) of multi-sensor data are, on the average, more similar to each other in the environment-only scenario than in the target-present scenario, a target is declared to be present if the divergence between the two clusters of interest is higher than a specified discriminating threshold $T$.

Figure 6.6 depicts the Receiver Operating Characteristic (ROC) curves to illustrate the performance of hypothesis testing for target detection using the entire data set. For all three proposed clustering algorithms, the performance of cosine distance (or similarity) measure significantly exceeds those of other feature divergence measures. For Euclidean and city block divergences in k-means and hierarchical clustering, the former provides better true positive rates for the same false positive rate along the entire ROC curve. From the ROC curve obtained for spectral clustering, it is seen that the spectral properties of the Laplacian matrices (with cosine similarity measure) obtained for the features can separate detect targets without any confusion (or error). To see this more clearly, the statistics of cluster divergence (or gap) which are obtained for all three approaches by using the different metrics are approximately fitted with the most-likely Gaussian distribution. The separability of the distributions with cosine distance (similarity) function for the two scenarios are visualized in Figure 6.5.

To obtain the probability distributions, first the characteristics of the diver-
Table 6.3: Average confusion matrices for feature-level fusion target detection I

<table>
<thead>
<tr>
<th>Distance</th>
<th>Target</th>
<th>No Target</th>
<th>Target</th>
<th>No Target</th>
</tr>
</thead>
<tbody>
<tr>
<td>Euclidean</td>
<td>Target</td>
<td>20.1%</td>
<td>79.9%</td>
<td>20.7%</td>
</tr>
<tr>
<td></td>
<td>No Target</td>
<td>24.9%</td>
<td>75.1%</td>
<td>20.0%</td>
</tr>
<tr>
<td>City Block</td>
<td>Target</td>
<td>60.0%</td>
<td>40.0%</td>
<td>69.6%</td>
</tr>
<tr>
<td></td>
<td>No Target</td>
<td>1.5%</td>
<td>98.5%</td>
<td>4.9%</td>
</tr>
<tr>
<td>Cosine</td>
<td>Target</td>
<td>100%</td>
<td>0%</td>
<td>100%</td>
</tr>
<tr>
<td></td>
<td>No Target</td>
<td>10%</td>
<td>99.0%</td>
<td>0%</td>
</tr>
</tbody>
</table>

Table 6.4: Average confusion matrices for feature-level fusion target detection II

<table>
<thead>
<tr>
<th>Similarity</th>
<th>Target</th>
<th>No Target</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spectral</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Gaussian Similarity</td>
<td>Target</td>
<td>46.3%</td>
</tr>
<tr>
<td></td>
<td>No Target</td>
<td>11.6%</td>
</tr>
<tr>
<td>Cosine Similarity</td>
<td>Target</td>
<td>100%</td>
</tr>
<tr>
<td></td>
<td>No Target</td>
<td>0%</td>
</tr>
</tbody>
</table>

gence (or gap) among the members of the PDFA set $G$ are generated for all the 100 cases using the three clustering techniques. Then, the gap statistics between the two most dissimilar members of $G$ are obtained and fitted with a Gaussian distribution for both environment-only and target-present scenarios. The distribution obtained for cosine distance (or similarity) for all three clustering methods are distinctly separable. The difference in the mean cluster divergence of the two events is greater than at least twice the standard deviation of the respective distributions, as seen in the histogram and estimated distributions of Figure 6.5 for cosine measurement cases.

Both k-means and hierarchical clustering are applied with distance measurements, while spectral clustering uses similarity measurements. Hence, the cluster distance for spectral clustering represents the negative similarity measurements, which are presented as negative real values.

It is observed that the variance parameters in the probability distribution of cluster divergence for both environment-only and the target-present scenarios are approximately similar in most cases (except for the cosine distance function under k-means clustering), and that the mean value for target-present is always larger than that for environment-only. With Gaussian distribution models for both hypotheses $H_0$ and $H_1$, it follows from Eq. (6.2) that
\[ H_0 : X \sim \mathcal{N}(\mu_0, \sigma^2) \]
\[ H_1 : X \sim \mathcal{N}(\mu_1, \sigma^2) \] (6.7)

where the hypotheses \( H_0 \) and \( H_1 \) share same variance \( \sigma^2 \) and different expected values \( \mu_1 > \mu_0 \). It is noted that the variance \( \sigma^2 \) is obtained by averaging the variances of the distributions corresponding to the two hypotheses. Then, the likelihood ratio \( \Lambda \) in Eq. (6.3) is expressed in a closed form as
\[
\Lambda(x) = \exp \left( \left( \frac{\mu_1 - \mu_0}{\sigma^2} \right) \left( x - \frac{\mu_1 + \mu_0}{2} \right) \right) \quad \forall x \in \mathbb{R} \] (6.8)

Since \( \Lambda \) in Eq. (6.8) is a continuous and strictly increasing function, the decision threshold \( \eta \) for likelihood ratio \( \Lambda \) in Eq. (6.6) can be converted to the decision threshold \( T \) for observation \( x \) as:
\[ T = \Lambda^{-1}(\eta) \] (6.9)

Accordingly, Eq. (6.5) can be reformulated for this particular application as
\[ P_0(\{x > T\}) = \alpha \] (6.10)

and the parameter \( T \) is computed for Gaussian distribution \( \mathcal{N}(\mu_0, \sigma^2) \) as a function of the significance level \( \alpha \) as
\[ T(\alpha) = \sigma Q^{-1}(\alpha) + \mu_0 \quad \text{for} \quad \alpha \in (0,1) \] (6.11)

where \( Q(\theta) \triangleq \frac{1}{\sqrt{2\pi}} \int_{\theta}^{\infty} dt \ e^{-\frac{t^2}{2}} \quad \forall \theta \in \mathbb{R} \).

It follows from Eq. (6.10) that an \( \alpha \)-level Neyman-Pearson test for this case yields the optimal decision rule as
\[
\delta^*(x) = \begin{cases} 
\text{Select } H_1 & \text{if } x > T(\alpha) \\
\text{Select } H_0 & \text{if } x \leq T(\alpha)
\end{cases} \] (6.12)

To evaluate the robustness of the feature-level fusion, cross validation is performed by different combinations of training and test data. In the training phase, a ROC curve is constructed from 80 cases (40 from each of the environment-only and target-present scenarios) of training data. The threshold \( T \) is obtained from Eq. (6.11) so that the false-alarm probability does not exceed the assigned significance level \( \alpha = 0.05 \) for Neyman-Pearson criterion [60]. Then, this threshold value is used for decision-making in the remaining 20 cases for both scenarios to compute the confusion matrices in the test phase. In total, 100 different combinations of training and test data are used and the results are summarized in Table 6.3 and 6.4.
Figure 6.6: Receiver operating characteristic (ROC) curves for feature-level target detection

While an arbitrary classifier could have chosen wildly different decision boundaries in a dynamic environment, the proposed method achieves stable decision boundaries with multi-sensor fusion. The results of cross validation show that the performance of cosine distance (similarity) measure is consistently good regardless of the choice of the clustering algorithm. The false positive rates of cosine similarity in the test phase are all below 5% as required, while other measures fail to meet this criterion. The true positive rate for cosine measures are above 95% for all clustering algorithms, while the other metrics show a rather poor performance for all clustering algorithms.

Table 6.5: Cluster purity for target-present scenarios

<table>
<thead>
<tr>
<th>Feature Measure</th>
<th>k-means</th>
<th>Hierarchical</th>
<th>Spectral</th>
</tr>
</thead>
<tbody>
<tr>
<td>Euclidean Distance</td>
<td>84.44%</td>
<td>80.00%</td>
<td></td>
</tr>
<tr>
<td>City Block Distance</td>
<td>85.11%</td>
<td>86.44%</td>
<td></td>
</tr>
<tr>
<td>Cosine Distance</td>
<td>86.67%</td>
<td>87.11%</td>
<td></td>
</tr>
<tr>
<td>Gaussian Similarity</td>
<td></td>
<td></td>
<td>84.22%</td>
</tr>
<tr>
<td>Cosine Similarity</td>
<td></td>
<td></td>
<td>88.00%</td>
</tr>
</tbody>
</table>

Table 6.5 presents the results of cluster purity [122] for 50 cases with a moving target. Due to spatial sparsity of the moving target, only a few sensors can detect targets while the rest collect environment-only data. The nine sensors are clustered into two groups as: (a) environment-only and (b) target-present. Then, the clustering purity is measured by calculating the correct grouping of sensors in the two clusters. Among all metrics (i.e, distance and similarity measures) used, the cosine distance (similarity) measure provides the highest cluster purity for all three clustering algorithms. On an average, all three algorithms misclassify only one or two sensor(s) out of nine for each event. However, the performance difference is not that profound under different combinations of similarity measures and
clustering algorithms.

The comparison between the cosine similarity measure and a typical distance metric (e.g., Euclidean) has been reported in the open literature [96, 123]. Although the characteristics of feature vectors could be more favored by one measure than the other [124], cosine measures usually outperform Euclidean measure [125, 126] in clustering problems. The reason is that distance metrics (e.g., Euclidean and City Block) are measures of the distance between a pair of points in the space, while cosine similarity is a measure of the angle between the pair of vectors. For the clustering purpose, cosine similarity provides the capability of distinguishing changes in the direction of the feature vector in contrast to the absolute numerical difference of individual elements of the feature vector.

6.7 Summary

This chapter presents a generalized framework for robust multi-sensor target detection in dynamic environments, where low-dimensional features are extracted from different sensors by making use of Symbolic Dynamic Filtering (SDF) [90][91]. A data-driven method is proposed for sensor fusion, which does not rely on availability of sensor and process models. The tool of symbolic dynamic filtering (SDF) [90] is applied for feature extraction from time-series data, collected from different sensors. The structure of the set of features is then assessed using three different clustering methods while using different metrics for them. Due to a dynamic background environment, the structure of each of the clustered features changes at different time instants; however, within the time span of an event of interest, all sensors share the same environment dynamics. Under this observation, a decision on the presence of a target is made by first using the divergence between the two most dissimilar clusters and then by learning the cluster gap statistics for the binary hypotheses during the training process. It is also shown how to use different criteria (e.g., maximum false-alarm probabilities) to calculate a threshold for the deciding rule.

Performance of the proposed target detection method is experimentally evaluated by hypothesis testing in a laboratory setting for a single moving target, where detection is made by multiple infrared sensors under fluctuating ambient light. Results of experimentation show that the proposed feature-level method outperforms a supervised decision-level fusion scheme in uncertain dynamic environments. The robustness of the performance is evaluated by a cross validation process and the results show perfect positive detection with low false alarm probabilities. The decision boundaries have been obtained under the constraints of maximum allowable false alarm probabilities by applying Neyman-Pearson decision criterion [60]. The results (i.e., the decision boundaries) are found to be stable with (almost) perfect accuracy for detection of targets with appropriate choices of the distance function and similarity matrix (i.e., cosine distance and cosine similarity).
This chapter proposes an approach to analyze the contribution of sensors of a passive sensor network in the information level, and demonstrates its application to detection of moving targets in a dynamic background environment. The proposed information contribution measures is formulated under a information-theoretic symbolic dynamics framework: D-Markov machines modelled as Probabilistic Deterministic Finite State Automata (PDFA). For event detection in passive sensor network, the overall information state of the network is presented by the top principal component of the data collected across the network. To quantify the contribution of each sensor in the information content, standard D-Markov models of the individual sensors as well as cross D-Markov models of the sensors conditioned on the network information state are created. The difference of the conditional entropies between these machines is treated as the information contribution measurement for each sensor. The proposed method is applied to experimental data collected from a small local passive sensor network for target detection under dynamic environment, where characteristics of environmental signal is similar to the target signal in the sense of time scale and texture. Various results are presented regarding the correlation between the sensors across the network.

7.1 Introduction

This chapter addresses the problem of dynamic sensor selection for event detection, where a major challenge is that, besides high-frequency measurement noise, the spectral (i.e., frequency) characteristics of environmental and event signals are largely similar in both amplitude and time scales. Therefore, the underlying algorithm must be able to distinguish between and carefully choose the sensors that would detect target events and the sensors that only possess the environmental
information. As stated earlier, another challenge is that the structure of dynamic environment may be partially or completely unknown and (possibly) always changing. Therefore, the sensor selection algorithm for target detection must be executed in an unsupervised fashion to adapt to different environmental conditions. The paper investigates how a symbolic dynamics-based, information-theoretic technique would overcome the difficulties of making accurate decisions for selecting sensors that may not provide unique target information by filtering out ambiguous information. To this end, a measure of information is constructed for event detection under a dynamically changing environment; this measure represents the information contribution of each sensor.

This chapter is a significant extension of previous chapter as well as the authors’ earlier paper [127] that was presented in a conference as a preliminary version. The network data are first decomposed into respective orthogonal components by using the standard Principal Component Analysis (PCA) [32], and the network information state is then represented as the component that embeds the maximum variance in the network data. The time series at each sensor node is discretized into a symbol sequence for information compression. Eventually, two different types of Markov machine models are constructed; one based on the sensor symbol sequence itself [90, 91], and, in the other one, the individual sensor data are conditioned on the approximate network state [128]. Finally, the contribution of each sensor to the network information is computed based on the difference between conditional entropies measurements of these two types of Markov machines, which is inspired from the concept of transfer entropy [129]. While the earlier work by the authors in [119] presented a feature-level fusion algorithm for passive networks, this paper has developed new strategies for improving network performance by modeling the symbol-level correlations across the network by using $\times D$-Markov machines [128]. The proposed method has been experimentally validated on a multi-sensor target detection problem in a laboratory setting that was reported in a recent publication [119]. The results suggest that the proposed information-theoretic, entropy-rate based sensor fusion algorithm is robust to the dynamic environment and it able to achieve near-perfect performance at very small alphabet size used for discretization of data (this is especially beneficial for computation and communication of data).

The chapter is organized in five sections including the current section. Section 7.2 elaborates the proposed approach adopted in this chapter for the addressed problem along with a list of major assumptions. Section 7.3 briefly presents the details of the experiment and the results of experimental validation of the proposed concept. Finally the chapter is summarized in Section 7.4.
7.2 Proposed Approach

This section presents the proposed method of measuring the information contribution of each sensor in networks under dynamic environment. First, a general sensor sensing model adopted in this chapter is formulated and basic assumptions are provided with consideration of practical applications purposes. Then, the development of proposed measurement is stated based on the information provided in Section 2.4. Finally, the target detection problem is formulated as a hypothesis test under the proposed framework.

7.2.1 Sensor and Target Model

Consider a sensor network consisting of $N$ sensors for surveillance of an area of interest. The $i^{th}$ sensor generates a real-valued time series $y_i = \{y_i[1], y_i[2], \ldots, y_i[T]\}$ for $T \in \mathbb{N}$. The sensor signal at each instant $y_i[k]$ is a (non-linear) function of dynamic environmental information $S_E[k]$, target information $S_T[k]$, and measurement noise $N[k]$. The structures of both environmental and target information are stochastic and unknown. However, there is assumed knowledge of their distribution. The sensor signal for both “target absent” (i.e., environmental disturbances only) and “target present” cases during time span $k \in [0 \ T]$ are modeled as follows, respectively:

\[
\begin{align*}
\text{Target Absent} : y_i[k] &= f(S_E[k], N[k]) \\
\text{Target Present} : y_i[k] &= g(S_E[k], S_T[k], N[k])
\end{align*}
\]  
(7.1)

In this context, the overall information from the sensor network is represented by the first principal component $x$ obtained from all sensor time series matrix $Y = [y_1 y_2 \ldots y_N]$. Let $v_i$ be the normalized eigenvectors of the real symmetric matrix $YY^T$ corresponding to the (real positive) eigenvalues $\lambda_i$ that are arranged in decreasing order of magnitude, i.e., $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_N$. The corresponding (normalized) eigenvectors $u_i$ in the original data space are obtained in terms of $v_i$ and $\lambda_i$ as follows [32][130]:

\[
u_i = \frac{v_i}{\sqrt{N \lambda_i}} Y^T, \quad i = 1, 2, \ldots, N.
\]  
(7.2)

Then, the first principal component $x$ of all sensor time series matrix can be expressed as:

\[
x = Y u_i
\]

It is noted that the first principal component $x$ is a weighted linear combination of all the sensor readings, it contains both environment and target information under the “target present” cases.
7.2.2 Information Measures in Sensor Network

In this subsection, a new causality measure between the first principal component time series \( x \), as a representative of the overall information from the sensor network, and individual sensor reading time series \( y_i \) \((i \in [1, 2, \ldots, N])\) is proposed under the framework of ×D-Markov machines. The objective is to evaluate the contribution of each sensor to the overall information content of the network. All time series are pre-processed and discretized into symbol sequences before any operation for low communication and computation costs.

According to Definition. 2.4.6, the transfer entropy from \( x \) to \( y_i \) can be expressed as:

\[
T_{x \rightarrow y_i} = H(y_i[k] \mid y_i[k-D:k-1]) - H(y_i[k] \mid x[k-D:k-1], y_i[k-D:k-1])
\] (7.3)

which quantifies the amount that the uncertainty of predicting future value \( y_i[k] \) can be reduced by getting extra information from \( x[k-D:k-1] \). In this chapter, we introduce a new information measure that compares the predictability of future value \( y_i[k] \) given the past value of itself \( y_i[k-D:k-1] \) and the past value of the first principle component \( x[k-D:k-1] \) with a pre-defined memory \( D \). Then, this new information measure can be expressed as:

\[
\Delta I_{x \rightarrow y_i} = H(y_i[k] \mid y_i[k-D:k-1]) - H(y_i[k] \mid x[k-D:k-1])
\] (7.4)

As the transfer entropy, it is a causality measure between \( x \) and \( y_i \). If \( \Delta I_{x \rightarrow y_i} < 0 \), it shows that the conditional entropy of future value \( y_i[k] \) given the past value \( x[k-D:k-1] \) is higher than that given the past value \( y_i[k-D:k-1] \). In that case, it indicated that the predictability of \( y_i[k] \) by \( x[k-D:k-1] \) is less than \( y_i[k-D:k-1] \). Since \( x \) represents the overall information in the network, the sensor reading time series \( y_i \) is assumed to "drive" the \( x \) in time (i.e., it contains unique information which contributes to the network). By a similar argument, the first principle component \( x \) is "driving" the sensor reading time series (i.e., the information from sensor readings is commonly shared in the network) \( y_i \) when \( \Delta I_{x \rightarrow y_i} > 0 \).

To compute the proposed information measure under the framework of ×D-Markov machines, the set of states \( Q \) is assigned as the all the words \( w \) (i.e., symbol block) with the same length as the memory \( D \):

\[
Q = \bigcup_{|w|=D} w
\] (7.5)

With the conditional Entropy of ×D-Markov machine defined in Equ. 2.8, the proposed information measure can be reformulated as:
Figure 7.1: Flow chart to generate information contribution measurement for each sensor in the network

\[ \Delta I_{x\rightarrow y_i} = H(\Sigma_{y_i}|Q_{y_i}) - H(\Sigma_{y_i}|Q_x) \] (7.6)

Figure 7.1 presents the procedures to generate information contribution measure for each sensor in the sensor network.

### 7.3 Experimental Validation

This section validates proposed algorithm for target detection in a dynamic environment on a laboratory apparatus that is described below.

#### 7.3.1 Signal Pre-processing to enhance robustness

Since the orientation and location of sensors are diverse in the experiment, the time series readings from each sensor varies relative to each other, even for sensors under the same dynamic environment. To obtain robust symbolic dynamic features with improved signal-to-noise ratio.

Figure 7.2 presents an example of the proposed noise-adding technique applying on a "Target Absent" (i.e., dynamic environment only) event. Once symbol strings are obtained from the sensor reading time series via the Maximum Entropy Partition (MEP), PDFA features are generated accordingly via the symbolic dynamic filtering algorithm described in Section 2.6. To measure the pair distance between two PDFA features from different sensors, the cosine distance function \( D_C(\bullet, \bullet) \) is chosen in this chapter (the rationale can be found in previous chapter). It can be expressed as:
Figure 7.2: Example: (a) and (b) present the partition of sensor time series before/after adding noise with alphabet size of 4. Time series of sensors are marked in blue. The line for symbolization is marked in red. (c) and (d) depict the distance among PDFA features using cosine distance measurement.

\[
D_C(a, b) = 1 - \frac{\sum_{i=1}^{n} a_i b_i}{\sqrt{\sum_{i=1}^{n} a_i^2} \sqrt{\sum_{i=1}^{n} b_i^2}}
\]  

(7.7)

where \( a = [a_1 a_2 \cdots a_n] \) and \( b = [b_1 b_2 \cdots b_n] \) are two real numbered vectors with same dimension of \( n > 0 \).

Although the location and orientation of sensors are divergent from each other, they captured a similar environment information pattern under the "Target Absent" case. To evaluate the similarity among sensor features, agglomerative (i.e., bottom up) hierarchical clustering is adopted in this chapter. The top cluster divergence (as depicted in Fig. 7.2 (c) and (f)) is used to quantify the overall sensor feature similarity in the sensor network. Plots (b) and (d) in Fig. 7.2 present the feature clustering results for sensor time series with and without adding noise, respectively. There is a significant reduction on the maximum cluster divergence when a identical independent zero-mean gaussian noise is introduced to each sensor time series, since the effects from the sensor location and orientation are mitigated due to the added noise.

The objective of this time series pre-processing step is to increase the similarity
Table 7.1: Average feature divergence ratio between Target Present and Target Absent cases under different alphabet sizes

<table>
<thead>
<tr>
<th>Alphabet Size</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Feature Divergence Ratio: without noise</td>
<td>2.42</td>
<td>2.06</td>
<td>2.03</td>
<td>2.05</td>
<td>2.03</td>
<td>1.98</td>
<td>1.96</td>
</tr>
<tr>
<td>Target / No Target</td>
<td>3.28</td>
<td>3.90</td>
<td>4.33</td>
<td>4.33</td>
<td>4.34</td>
<td>3.88</td>
<td>3.58</td>
</tr>
</tbody>
</table>

of PDFA features among sensors under the "Target Absent" cases while maintain the dissimilarity of PDFA features between sensors with target detected and others under the "Target Present" cases. It is inevitable that certain information embedded in the time series would be compromised when any noise is introduced. To verify the influence of this technique on the sensor features under the "Target Present" cases, Table 7.1 presents the average ratio of top cluster divergence for "Target Present" to "Target Absent" cases with/without adding noise under different choice of the alphabet size for PDFA feature generation. It demonstrates that even with an extra noise in the sensor time series, the sensor feature cluster divergence for the "Target Present" cases are amplified compared to the measures under the "Target Present" cases. This indicates that the performance of using the PDFA feature to differentiate "Target Present" to "Target Absent" event is improved by this pre-processing procedure.

### 7.3.2 Information Causality Measures

As mentioned in the previous section, the sign of the information causality measure reflects the direction of causality between two time series. The positive value indicates that the first principle component, which represents overall network information, has the initiative compared to the sensor under examination. The negative value indicates that sensor has its own unique information (i.e., transition pattern in time series) which leads to a contribution to the network information content.

The proposed information causality measure is computed by the difference of the conditional entropy between two types of Markov machines constructed from sensor time series. The conditional entropy of a standard (self) $D$-Markov machine for each sensor time series represents the uncertainty of predicting the future reading based on its own past readings. The conditional entropy of $\times D$-Markov machines between sensor time series and the first principal component measures the uncertainty of predicting the future reading of sensor time series based on the first principal component.

Figure 7.3 depicts the average value of conditional entropies of both standard and $\times D$-Markov machines for both "Target Absent" and "Target Present" cases.
Figure 7.3: Average conditional entropies of Markov machines for both "Target Present" and "Target Absent" cases

under the alphabet size of $|\Sigma| = 4$. For the "Target Absent" case, the conditional entropy of the standard $D$-Markov machine is larger than that of $x$-$D$-Markov machine, except for sensor No. 3. It indicates that sensor No. 3 dominated the information contribution relative to the first principal component due to its location and orientation to the ambient light. On the other hand, sensors No. 3, 4, and 5 has significantly smaller conditional entropy for standard Markov machine compared to cross Markov machine. Since sensor No.3, 4, and 5 are top three sensor which may have target detected, they contribute the most to the first principal component compared to others, which only contain environment information.

Figure 7.4: Average information causality measures for both target present and target absent cases

Figure 7.4 depicts the average value of proposed information causality measures of each sensor under both "Target Present" and "Target Absent" cases. For the "Target Absent" case, the information causality measures are all positive and
similar among all sensors, except for sensor No. 3. On the other hand, information causality measures for "Target Present" presents a distinguishable pattern for sensors. It is noted that sensor No. 3, 4, and 5, which have target detected all the time, have negative mean information causality measure values, while the value of sensors with no target information are all positive and of small magnitude.

7.3.3 Alternative Network Information Representation

In this chapter, the first principal component is chosen as the representation of overall network information, since it maximizes the variance of the projection from sensor quantized symbol strings. As expressed in Eq. (7.3), the first principal component is a weighted linear combination of the original sensor strings.

![Figure 7.5: Average PCA coefficients both target present and target absent cases](image)

Figure 7.5 depicts the average PCA coefficients of each sensor for different scenarios. For "target absent" cases, the coefficients are numerically similar among each sensors, while sensor No.3, 4, 5, and 6 have significant more contribution to the first principal component for the "target present" cases. In average, the variance explained by the first principal component is 68.81% for "target absent" cases and 25.74% for "target present" cases. In all, the first principal component, as the representation of the overall network information, is linear combination among sensor symbol sequences with case-dependent numerical weights. To simplify the step of acquiring network information, we here propose an alternative representation for the information:

\[ X = \sum_{i=1}^{N} \frac{1}{N} y_i \]  

(7.8)

This information representation is a simple (equally weighted) linear combination of all the sensor symbol sequences. Compared with the first principal com-
ponent, this representation has a much lower computational cost to generate. As will be shown in the next subsection, it provides equivalent target detection performance as the first principal component.

### 7.3.4 Target Detection Results

![ROC curves for different values of \( \Sigma \)](image)

Figure 7.6: ROC curve for target detection problem under different choice of alphabet size

In previous chapter, the decision rule for target detection under a dynamic environment is based on the thresholding on the top clustering divergence of sensor standard Markov machine features constructed from time series without adding any noise. It shows (nearly) perfect performance with large alphabet size (i.e., \(|\Sigma|=10\)) with cosine distance measure between features. In this chapter, we compare results of feature clustering algorithm with and without adding a identical and independent zero-mean Gaussian noise; and also with the result of the proposed information causality measures of sensors.

Figure. 7.6 presents the ROC curves for target detection under dynamic environments for the different algorithms under different alphabet sizes. From the results, it is shown that the performance of thresholding on the top cluster divergence is improved as the alphabet size increases. In particular, the clustering algorithm with added a zero-mean Gaussian noise is generally superior than that with no noise added. On the other hand, the performance of the proposed information causality measure is the best among the three algorithms. More importantly, this algorithm obtains nearly prefect target detection at a small number choice of
alphabet size and it continues to obtain consistently good results as the alphabet size increases.

7.4 Summary

This chapter presents a generalized framework for information contribution measurements in sensor networks and apply the proposed method to event detection in dynamic environments. The information contribution of each sensor to the overall network information is measured by the difference between conditional entropy computed by the standard $D$-Markov machine constructed from sensor time series and the principal component of all sensors time series, which is the representation of the overall network information. The proposed method for multi-sensor target detection in a dynamic environment had been validated by utilizing the algorithm within an experimental setting.

While there are several issues that need to be resolved by further theoretical and experimental research, the authors suggest the following topics as potential topics of future research:

1. **Sequential hypothesis testing**: A framework is required for more complicated scenarios of target classification when multiple homogeneous and (or) heterogeneous targets are present.

2. **Experimental validation**: The test apparatus should be able to accommodate a large-scale sensor network.
Chapter 8

Time Series Symbolization for Information Extraction

This chapter formulates an unsupervised algorithm for symbolization of signal time series to capture the embedded dynamic behavior. The key idea is to convert time series of the digital signal into a sequence of (spatially discrete) symbols from which the embedded dynamic information can be extracted in an unsupervised manner (i.e., labels for time series are not required). The main challenges are: 1) selection of the symbol alphabet size; 2) identification of the partitioning segment locations in the signal space of time series; and 3) construction of probabilistic deterministic finite-state automata (PDFA) from the symbol sequences that contain temporal patterns. The reported work addresses these challenges by maximizing the mutual information measures between symbol strings and PDFA states. The proposed method has been validated by numerical simulation as well as on experimental data sets. Performance of the proposed algorithm is compared to those of two commonly used algorithms of time series partitioning.

8.1 Introduction

To the best of the authors knowledge, a majority of symbolic time series analysis approaches reported in literature treat symbol definition and word selection as two separate tasks. These two problems are solved in a fixed order: (i) partitioning to obtain symbols, followed by (ii) modeling of automata to obtain words as states. In contrast, the current paper proposes an algorithm that optimally chooses the symbols and words simultaneously by maximizing their mutual information measures, which is accomplished as follows. First, a set of potential partition locations is assumed as an initial condition; then, iteratively, one more segment is created in the existing partition to accommodate one more symbol in the alphabet. For each potential partition location, the corresponding optimal word set is obtained.
by maximizing the mutual information measure between the respective symbol and word. The potential partition location with the highest mutual information is chosen and added to the existing partition. In this way, new symbols are being kept added until the mutual information stops increasing. The proposed method has been validated on simulation and experimental data sets and its performance is compared to those of several commonly used methods of signal discretization. In the context of dynamic data-driven application systems [108],

The rest of the chapter is organized as follows. Section 8.2 elaborates the proposed approach of unsupervised time series symbolization adopted in this paper. Section 8.3 validates the proposed concepts on a number of artificial and realistic experimental data sets by comparison with two commonly used methods of time series partitioning. Section 8.4 summaries the chapter.

8.2 Proposed Method

This section proposes an unsupervised algorithm of time series symbolization (i.e., with no requirement for labeling of time series), based on information-theoretic Markov modeling of time series. The symbolization process identifies a suboptimal partition \( P_{\text{opt}} \) that maximizes the mutual information between the symbols in \( \Sigma \) and the states in \( Q \) as:

\[
I(\Sigma, Q) = H(\Sigma) - H(\Sigma|Q) \\
= - \sum_{\sigma \in \Sigma} P(\sigma) \log P(\sigma) - \sum_{q \in Q} H(\Sigma|q) \\
= - \sum_{\sigma \in \Sigma} P(\sigma) \log P(\sigma) + \sum_{q \in Q} \sum_{\sigma \in \Sigma} P(q)P(\sigma|q) \log P(\sigma|q) \tag{8.1}
\]

In Eq. (8.1), the mutual information \( I(\Sigma, Q) \) consists of two terms, where the first term is the entropy of the symbols in \( \Sigma \) and the second term is the entropy rate between the symbols in \( \Sigma \) and the states in \( Q \). The entropy of symbols in \( \Sigma \) is maximized (i.e., the uniform distribution of symbol occurrence in the symbol sequence) and the entropy rate is minimized (i.e., the uncertainty reduction for occurrence of the next observed symbol from a given state). The optimization task is divided in three stages as explained below.

- **Stage 1**: For a given time series, the optimal alphabet size \( |\Sigma| \) is determined as:

\[
|\Sigma|_{\text{opt}} = \min \left\{ n \left| \frac{f_{\text{max}}}{f_{\text{max}}(\Sigma, Q)} < \epsilon_{|\Sigma|} \right. \right\} \tag{8.2}
\]
where $\varepsilon|\Sigma| > 0$ is a pre-defined threshold and $I_{\text{max}}^n(\Sigma, Q)$ is the maximum mutual information for alphabet size of $|\Sigma| = n$. Then, the pair of optimal partitioning locations $P_{\text{opt}}$ and the corresponding optimal state set $Q_{\text{opt}}$ maximizes the mutual information $I(\Sigma, Q)$ in Eq. (8.1) as:

$$\{P_{\text{opt}}, Q_{\text{opt}}\} = \arg \max_{P, Q} I^n(\Sigma, Q)$$

(8.3)

It is noted that the optimization in Eq. (8.3) is nonconvex in general. A greedy search algorithm is proposed to circumvent this problem by selecting the initial condition from a finite number of candidate combinations.

**Algorithm 2** Partitioning by sequential selection based on an information-theoretic measure

**Require:** Time series $X = [x_1, x_2 \ldots x_N]$, Alphabet size $|\Sigma|$, and number of partition positioning locations candidate $m$

**Ensure:** Partition position vector $P_{\text{opt}} \in \mathbb{R}^{|\Sigma|+1}$ and the maximum mutual information $I_{\text{max}}^{\Sigma}(\Sigma, Q)$ for alphabet size $|\Sigma|$

1: Create the set of candidate partitioning locations $C = \{c_1, c_2, \ldots, c_m\}$ via MEP algorithm

2: Assign the partitioning location vector $P_{\text{opt}}(1) = \min(X)$ and $P_{\text{opt}}(|\Sigma| + 1) = \max(X)$

3: for $i = 2$ to $|\Sigma|$ do

4: Following Eq. (8.3)

$$P_{\text{opt}}(i) = \arg \max_{c \in C} I_i^{\Sigma^c}(\Sigma^c, Q_{\text{opt}}^c)$$

where $\Sigma^c$ is the corresponding alphabet based on partition $P_i^c = [P(1) \ldots P(i-1) c P(|\Sigma| + 1)]$ and $Q_{\text{opt}}^c$ is the state set that achieves maximum mutual information under this location.

5: Remove the selected partitioning location $P_{\text{opt}}(i)$ from the candidate set $C$

6: end for

- **Stage 2:** Identification of partitioning locations $P$ is formulated as a sequential selection process. First, a set of candidate partitioning locations $C = \{c_1, c_2, \ldots, c_m\}$ is obtained by MEP on the time series. The total number of candidate partitioning locations $m$ is pre-defined and must be a significant large number (e.g., $10^2 \sim 10^3$). The exact number depends on parameters such as the length of time series, computational capability, and the accuracy specification. In each iteration of the algorithm, all available candidate partitioning locations $C$ are individually added to the current selected partitioning locations $P$ and the corresponding maximum mutual information is calculated. The candidate achieving the highest maximum
mutual information is chosen and added to the current selected partitioning locations $P$. This process is repeated until the desired number of alphabet size is achieved. The details of the partitioning locations selection algorithm are presented in Algorithm 2.

Usage of Eq. (8.3) in Algorithm 2 yields $P_{\text{opt}}(i)$ for sequential selection of the partitioning locations. The mutual information $I^i(\Sigma_c, Q_c)$ at $i^{th}$ iteration is expressed as:

$$I^i(\Sigma^c, Q^c) = H(\Sigma^c) - H(\Sigma^c | Q^c)$$  (8.4)

The first term on the right side of Eq. (8.4) is the symbol entropy $H(\Sigma^c)$, which solely depends on the partition $P^c$. The second term is the entropy rate which depends on both alphabet $\Sigma^c$ and the state set $Q^c_{\text{opt}}$. This leads to the final stage of optimization process.

- **Stage 3:** For a given symbol sequence $\{s_1s_2...s_N\}$ over the alphabet $\Sigma$, the optimal $Q_{\text{opt}}$ is obtained by maximizing the mutual information. Since the symbol entropy $H(\Sigma_c)$ is independent of the choice of states, the optimization is carried out by minimizing the entropy rate $H(\Sigma | Q)$:

$$Q_{\text{opt}} = \arg\max_Q I(\Sigma, Q) = \arg\min_Q H(\Sigma | Q)$$  (8.5)

Identification of the state set starts off with the simplest set of states (i.e., $Q = \Sigma$) and subsequently splitting the current state results in the largest decrease of the entropy rate. The process of splitting a state $q \in Q$ is executed by replacing the symbol block for $q$ by its branches given by the set $\{sq : s \in \Sigma\}$. Then, the maximum reduction of the entropy rate of the PDFA $\mathcal{M}$ is the governing criterion for selecting the state to split. Figure 8.1 depicts an example to clarify the concept, based on the following two important criteria in state splitting:

- **Threshold for occurrence probability** $\varepsilon_p$, indicating whether a state contains meaningful statistical information or not.
- **Threshold for entropy rate reduction** $\varepsilon_q$, determining whether states of longer memory is necessary or not.

As a result, not all the states are split and a variable-structure PDFA is created. State splitting for inferring the PDFA is delineated as Algorithm 3. The parameters of the Markov model are estimated using maximum a posteriori (MAP) rule with uniform prior. Let $N(\sigma_j|q_i)$ denote the number of times that a symbol $\sigma_j$ is generated in $S_2$ when the state $q_i$ as the symbol string is observed in the PDFA $\mathcal{M}$. The maximum a posteriori probability (MAP) estimate of the emission probability of the symbols $\sigma \in \Sigma$ conditioned on $q_i \in Q$ is estimated by frequency counting [11] as Eq. 2.20.
Figure 8.1: An example of state splitting with alphabet $\Sigma = \{\sigma_1, \sigma_2, \sigma_3\}$. The state $q$ is first split as $\{\sigma_1 q, \sigma_2 q, \sigma_3 q\}$. The new state $\sigma_2 q$ is not further split because its occurrence probability $P(\sigma_2 q)$ is less than the threshold $\epsilon_p$. The new state $\sigma_3 q$ is not further split because the sum of entropy rate of generated states $\{\sigma_1 \sigma_3 q, \sigma_2 \sigma_3 q, \sigma_3 \sigma_3 q\}$ is not significantly reduced compared to that of state $\sigma_3 q$. The state $\sigma_1 q$ is split as both criteria are satisfied. States after splitting are $\{\sigma_1 \sigma_1 q, \sigma_2 \sigma_1 q, \sigma_3 \sigma_1 q, \sigma_2 q, \sigma_3 q\}$.

8.3 Performance Validation

In general, evaluation of an unsupervised algorithm is difficult due to lack of the ground truth. This chapter evaluates the performance of the proposed time series symbolization algorithm in terms of temporal behavior based on extensive data from the following two sources:

1. Numerical simulation of a PDFA model.
2. Laboratory-scale experiments on a local passive infrared sensor network for target detection under dynamic environment.

The following parameters have been chosen in the above evaluation procedure.

- The threshold for alphabet size selection $\epsilon_{|\Sigma|} = 1.1$ in Eq. (8.2).
- The threshold of occurrence probability $\epsilon_p = 0.05$ for state splitting algorithm.
- The threshold of entropy rate reduction $\epsilon_q = 0.85$ for state splitting algorithm.
Algorithm 3 State splitting for variable-depth $D$-Markov machine

**Require:** Symbol sequences $S = \{ \ldots s_1 s_2 s_3 \ldots | s_i \in \Sigma \}$

threshold for state splitting $\varepsilon_q > 0$ and $\varepsilon_p > 0$

**Ensure:** State set $Q$

1. **Initialize:** Create a 1-Markov machine $Q^* := \Sigma$
2. repeat
3. $Q := Q^*$
4. for $i = 1$ to $|Q|$ do
5. if $\frac{H(\Sigma \setminus \{s q_i : s \in \Sigma\})}{H(\Sigma | q_i)} < \varepsilon_q$ and $P(q_i) > \varepsilon_p$ then
6. $Q^* = Q \setminus q_i \cup \{s q_i : s \in \Sigma\}$
7. end if
8. end for
9. until $|Q| = |Q^*|$

---

Figure 8.2: Example of simulation validation via time series generated from a PDFA model

### 8.3.1 Numerical Simulation

The PDFA $K$ in Figure 8.2 is a variation of the even shift machine with three symbols, i.e., $\Sigma = \{ a, b, c \}$, which is not a shift of finite type [78, 83], because the minimal set of forbidden words has a subset $\{ ab^{(2k-1)}a, ac^{(2k-1)}a \}$, $k \in \mathbb{N}$, which does not have a finite cardinality. Hence, this PDFA $K$ is not a $D$-Markov machine.

A 2,000-long symbol string is generated from $K$, where each symbol corresponds to a uniform distribution as depicted in Figure 8.2. A time series of real numbers is obtained from the symbol sequence by assigning uniform distribution by injecting additive zero-mean Gaussian noise with a pre-defined standard deviation of 0.1.

Having 30 partitioning regions as the initial guess, the time series is segmented by 29 boundary locations as seen in Figure 8.3 that depicts the 29 candidate locations generated by the MEP algorithm. These locations are further tuned by the proposed symbolization algorithm, where the maximum depth for state splitting algorithm is assigned as $D = 3$. Figure 8.4 depicts three consecutive iterations in the selection of partitioning locations via maximization of mutual information at each iteration. At the first iteration, the maximum mutual information is attained at the 19th location with a value of 2.02 at depth $D = 3$. However, the maximum
Figure 8.3: The 29 candidate partitioning locations for the simulation time series generated from PDFA

mutual information at $D = 2$ for the same location is numerically very close to that at $D = 3$. The numerical difference between these two different depths is possibly due to the computational uncertainties resulting from the finite length of the time series. After the $19^{th}$ location is selected in the first iteration, the remaining 28 locations are analyzed in the second iteration with alphabet size of $|\Sigma| = 3$. At this stage, the values of maximum mutual information at $D = 2$ and $D = 3$ are very close to each other for each location, which are significantly larger than the corresponding values at $D = 1$. The maximum mutual information at $10^{th}$ location is found to be 0.1932, which is $\sim 164\%$ improvement of mutual information measurement compared to that for $|\Sigma| = 2$. Finally, at the third iteration with $|\Sigma| = 4$, the maximum mutual information is 0.1866, which is smaller than that at $|\Sigma| = 3$; this is possibly due to finite length of the time series. Therefore the partition selection algorithm is stopped at $|\Sigma| = 3$ and the resulting partition is obtained as $P_{opt} = [-0.21 1.03 2.02 3.18]$. Comparing it with the distribution assignment to each symbol, the selected partition correctly captures the temporal transition pattern from the generating process under the 10% zero-mean Gaussian noise.

Table 8.1 presents the occurrence probabilities of states generated from the state splitting algorithm and the corresponding emission probability for each symbol. There are three states that contain pertinent statistical information \{a, bb, cc\}. Those states reflect the basic structure of the PDFA model in this example. For state $ab$, the symbol emitted is highly focused on symbol $b$. A similar situation is for state $ac$ and symbol $c$. This also follows the symbol emission rule in the PDFA model.
Figure 8.4: Mutual information for each candidate partitioning locations for each iteration at different depth $D$ of modeling.

Table 8.1: The occurrence probabilities of State and Symbol emission for the simulation time series

<table>
<thead>
<tr>
<th>Alphabet</th>
<th>a</th>
<th>b</th>
<th>c</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>0.2996</td>
<td>0.4618</td>
<td>0.2757</td>
</tr>
<tr>
<td>ab</td>
<td>0.0822</td>
<td>0.0476</td>
<td><strong>0.8690</strong></td>
</tr>
<tr>
<td>bb</td>
<td><strong>0.2157</strong></td>
<td>0.3364</td>
<td>0.4470</td>
</tr>
<tr>
<td>State</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cb</td>
<td>0.0681</td>
<td>0.1511</td>
<td>0.6763</td>
</tr>
<tr>
<td>ac</td>
<td>0.0786</td>
<td>0.0063</td>
<td>0.1063</td>
</tr>
<tr>
<td>bc</td>
<td>0.0646</td>
<td>0.1136</td>
<td>0.1742</td>
</tr>
<tr>
<td>cc</td>
<td><strong>0.1912</strong></td>
<td>0.3532</td>
<td>0.2597</td>
</tr>
</tbody>
</table>

8.3.2 Experimental Validation

The experimental apparatus consists of a small sensor network and dynamic ambient light. The sensor network is a ring of nine TCRT5000$^{\text{TM}}$ infrared sensors. Details of the collected time-series data characteristics have been reported earlier chapters. The dynamic environment and the associated environmental disturbances are emulated as variations in the daylight intensities on partially cloudy days.

To overcome the different environmental impacts (i.e., noise level) to different sensors, we proposed a consensus partition position selection algorithm for multiple time series based on the Algorithm 2. During the sequential selection of partitioning portions, the criterion for optimal selection of partitioning locations is revised to choose the candidate that maximizes the average mutual information over the ensemble of time series. The only information needed to be exchanged at each iteration is the candidate index and the corresponding maximum mutual information values. The detailed procedure for consensus partition is described in
Algorithm 4.

Algorithm 4 Consensus partition position selection for multiple time series

Require: Multiple time series $X_1, X_2, \ldots, X_L$, Alphabet size $|\Sigma|$, and number of partition position candidate $m$

Ensure: Partition position vector $P_{opt} \in \mathbb{R}^{|\Sigma|+1}$ and the average maximum mutual information $I_{\Sigma}^{\text{max}}(\Sigma, Q)$ among all time series at alphabet size $|\Sigma|

1: Create the set of candidate partitioning locations $C = \{c_1, c_2, \ldots, c_m\}$ via MEP algorithm for each time series

2: for $i=2$ to $|\Sigma|$ do

3: $P_{opt}(i) = \arg \max_{c \in C} \sum_{k=1}^{L} I_{k, \text{max}}^i(c)$ (8.6)

where $I_{k, \text{max}}^i(c)$ is the mutual information value for $k^{th}$ time series at alphabet size $|\Sigma| = i$ for $c^{th}$ candidate partition position.

4: Remove the selected partition position $P_{opt}(i)$ from the candidate set $C$

5: end for

The results are presented for 50 experiments. For each experiment, the nine sensors in Figure 6.2 record synchronized data for about 65 seconds with an average sampling rate of $\sim 18.5$ Hz.

Since the orientation and location of sensors are diverse in the individual experiments, the time series from each sensor varies relative to each other, even for sensors under the same dynamic environment. Figure 8.5 presents examples of noise injection for three different types of partitioning of sensor time series. Performance of the proposed unsupervised time series symbolization algorithm in plate (a) of Figure 8.5 is compared with two other symbolization algorithms, namely, maximum entropy partitioning (MEP) in plate (b), and uniform space partitioning (USP) in plate (c). The alphabet size for all cases of partitioning is selected as $|\Sigma| = 4$. The distance between two symbol strings is measured by the Hamming distance defined in Eq. (2.21), and the distance between two clusters of symbol strings is measured by the cluster distance defined in Eq. (2.25). Figure. 8.6 depicts the average maximum mutual information over nine sensors for different choices of the alphabet size $|\Sigma|$. Since the ratio of maximum mutual information for $|\Sigma| = 4$ and that for $|\Sigma| = 5$ is larger than the predefined threshold of 0.85, $|\Sigma| = 4$ is chosen as the alphabet size.

The maximum Hamming distance among all nine sensors for the proposed unsupervised symbolization algorithm is 0.217, that for the MEP algorithm is 0.489, which is more than twice the distance compared to the proposed method. For the USP algorithm, the corresponding Hamming distance is 0.566, which is the largest among the three partitioning algorithms. That is, the proposed symbolization method achieves the smallest Hamming distance based on the symbol strings.
Figure 8.5: Examples: Three different types of partitions for sensor time series: (a) Partitioning for the proposed unsupervised time series symbolization algorithm which is compared with two other unsupervised symbolization algorithms: (b) Uniform partitioning and (c) Maximum entropy partitioning. Results in plates (a), (b), and (c) are for alphabet size $|\Sigma| = 4$, where the partitioning line segments for symbolization are shown by (red) dashed horizontal lines and time series of sensors are marked in (blue) solid-line profiles. Plates (d), (e), and (f) depict the Hamming distance among generated symbol strings the respective binary trees. The distance between two symbol strings is measured by the Hamming distance defined in Eq. (2.21). The distance between two clusters of symbol strings is measured by the cluster distance defined in Eq. (2.25).

generated under different environmental impacts. The proposed symbolization algorithm provides small alphabet size as the result of the formulated optimization process. Among the 50 experiments, there are 9 cases with $|\Sigma| = 3$, 39 cases with with $|\Sigma| = 4$, and 2 cases with with $|\Sigma| = 5$. The average Hamming distance among all 9 sensors for the proposed algorithm is 0.40, compared with 0.53 for MEP and 0.57 for USP. It shows that the proposed algorithm significantly reduced the difference among generated symbol sequences from sensor time series under different noise levels. The dominant temporal pattern of environment dynamics embedded in different time series is magnified.
8.4 Summary

This chapter has developed an unsupervised algorithm for symbolization of time series. The objective is to model the embedded temporal behavior as a probabilistic deterministic finite-state automaton (PDFA) in an information-theoretic Markov setting. The results are obtained by maximizing the mutual information between symbols and states; consequently, the uncertainty of the constructed model is reduced. In this context, three main issues are addressed:

1. Determination of the symbol alphabet size $|\Sigma|$.
2. Identification of the boundary locations, $P$ for partitioning of the time series.
3. Identification of the state set $Q$ in the PDFA model.

The symbolization algorithm has been validated on time series data generated by both numerical simulation and laboratory experimentation. It is observed that the proposed algorithm outperforms two standard time series symbolization algorithms, namely, maximum entropy partitioning (MEP) and uniform space partitioning (USP).
Conclusions and Future Research Directions

The research presented in this dissertation investigates adaptive information extraction approaches for system identification and network decision-making via symbolic time series analysis (STSA). As discussed in the introductory chapter, three types complex systems are presented in this work to demonstrate the high customizability of the proposed framework for different data formats, system structures, and application purposes. In detail, this dissertation focuses on pattern recognition problems in following aspects:

- *Adaptive system parameter and state estimation under varying inputs:* To identifying system parameters that varying in the slow time scale and states that varying in the fast time scale, a $\times D$-Markov modeling tool is introduced in this dissertation to extract the dynamic characteristics of system responses for varying or fluctuating input profile at different working conditions. Compared to traditional data-driven approaches which only consider system outputs, dynamic symbolic model constructed from the proposed approach can be adaptive to different input patterns, which is analogous to transfer functions in modern control theory. To overcome the shortcomings of solo data-driven or model-based approaches, a hybrid recursive Bayesian filtering for system state estimation is proposed in this dissertation. The proposed estimator consists of a data-driven measurement model which is obtained from a supervised training process via $\times D$-Markov modeling at different operating conditions, as well as a physical model-based prediction model to compensate potential over-fitting and under-fitting problems of data-driven pattern recognition step.

- *Adaptive dynamic modeling of physical process via video data:* Unlike numerical sensing data, Markov modeling of video (i.e., a sequence of images) data
for dynamic information extraction is counter-intuitive due to its high complexity. A three-stage dimensionality reduction process is proposed in this dissertation for such purpose. Combined standard image processing techniques, feature clustering algorithms, and symbolic Markov modeling, the dominant pattern transition pattern embedded in the video can be represented as a trajectory in low-dimensional feature space or even a symbol sequence.

- Adaptive information fusion in sensor networks under dynamic environments: For large scale sensor networks, energy management is a critical issue since both sensing communication and computation abilities are achieved at a cost of energy consumption while sensor nodes are most likely battery-powered. The reliability and capacity of communication channel between nodes also could has constrains in the network. In this dissertation, we compare the network performance under different fusion schemes (i.e., data, feature, and decision levels) via symbolic Markov modeling of sensing data for a multi-sensor target detection problem under dynamic environments. A novel information contribution measurement is proposed to evaluate and quantify the information content value of each sensor node to the network. The proposed method significantly reduces the computational complexity at each sensor node as well as communication load in the network comparing to conventional data fusion and aggregation techniques. More important, the potential useful information which is contaminated with the ambient noise can be identified and located in the network.

9.1 Contributions of the Dissertation

In this section, the major contributions of this dissertation are summarized from both theoretical and practical perspectives.

- Unsupervised time series symbolization for information extraction: The major theoretical contribution of this dissertation is the development a unsupervised symbolization algorithm for 1-Dimensional time series data with finite length. The proposed method is formulated as a optimization problem with a greedy search strategy to obtain a suboptimal PDFA model from the original time series based on a objective function which is to maximize the mutual information value between symbol set and state set of PDFA model. Compared to conventional symbolic time series analysis techniques, the proposed method requires minimum input from the users such as data labels and pre-defined parameters during the symbolization process. Results shows that the proposed method can capture temporal patterns embedded in time series which is contaminated with both noise and other ambient dynamics.
• **Battery system identification via symbolic Markov modeling on input-output pairs:** A symbolic Markov machine models the (possibly nonlinear) input-output characteristics, constructed from input-output trajectories in 2-D space, is analogous to a transfer function realization without the need for linearization of the battery system dynamics. This representation significantly reduces computational complexity relative to the commonly used physical dynamic models. Besides that, a data-driven system state measurement model is developed based on the features obtained from Markov modeling on symbolic time series of battery input (i.e., current) and output (i.e., voltage) and integrated within the structure of a physical model based recursive Bayesian filtering for real-time system state estimation.

• **Dynamic modeling of combustion flame video for state classification:** A highly customizable framework for dimensionality reduction and symbolic Markov Modeling of physical process is developed to extract the dynamic temporal pattern embedded in the video data. The raw video data is compressed to a symbol sequence while the dominant dynamic characteristics of video temporal pattern is preserved. To classify different working conditions of physical processes, a sequential testing procedure is formulated as a Dirichlet-multinomial Bayesian inference based on the symbolic Markov Models. This approach makes real-time analysis and decision making possible for complex physical process captured by high speed video due to the low-complexity nature of the proposed dimensionality reduction framework.

• **Information fusion and evaluation for sensor networks under dynamic environment:** Unsupervised clustering algorithms at both data-level and feature-level are proposed for identifying event information in sensor network with presence of disturbance from varying environment which shares similar dynamic characteristics to event of interest. Symbolic dynamic filtering (SDF) with a noise-adding technique which can amplify the common underlying dynamics among sensor nodes is applied in this work. Besides that, an information casualty measurement based on the concept of transfer entropy in information theory is also proposed to evaluate the contribution of each sensor node to the network as for the information content aspect. With appropriate choosing representation of the network information (e.g., the first principle of PCA or linear combination of sensor time series), the importance of each sensor node can be evaluated and quantified to select sensors which potentially contain the desired event information.
9.2 Future Work Directions

While there are several issues that need to be resolved by further theoretical and experimental research, the following topics of future research are suggested:

- **Unsupervised symbolization for high dimensional time series**: Generalization of the proposed symbolization algorithm from one dimensional time series to vector time series. Many existing methods for clustering (i.e., partitioning) high-dimensional data based on the distribution are reported in the open literature, there are only very few of them considering the temporal transition pattern embedded in the data itself. The main challenge is that there exists only one unique partitioning solution at 1-dimensional space for any given symbolization objective function, which is not case when it is in two or higher dimensions. As proposed in this dissertation, one feasible approach is to conduct a dimensionality reduction to vector time series before symbolization. However, there are two potential issues in this approach needed to be taken care of: 1) information loss is inevitable during dimensionality reduction, the temporal patterns of vector time series should be preserved as much as possible; 2) the resulted output may still in a low-dimensional feature space, where each dimension may represent different information content.

- **Investigation of the effects of sampling rate on time series symbolization processes**: The choices of symbol set and state set are strongly influenced by the sampling rate at data acquisition stage. A over-sampled time series would lead to a PDFA model which is insensitive to the embedded temporal pattern; while a under-sampled time series may end up with a PDFA model that cannot describe the embedded dynamics at all. It is necessary to establish an adaptive algorithm that will allow autonomous downsampling or upsampling of time series before symbolization. Although the over-sampling issue can be investigated by measurements such as autocorrelation, the potential under-sampling is very difficult to discover without the detail knowledge of the dynamic system of interest. However, it is feasible to obtain a (sub)optimal sampling rate for the symbolization process by comparing the information-theoretical evaluations (e.g., Morph matrices, mutual information, and entropy rate) of resulted PDFA models when the sampling rate is adjective at data collection stage.

- **Dynamic data-driven modeling for system identification**: There are two potential research areas in this topic. First, system input profiling promotes system management efficiency. As system identification by $\times D$-Markov modeling of system input-output pairs, a separate PDFA model can be constructed from the input symbol sequence itself. By identifying the modeled input pattern with pre-trained data set, a better control strategy can be made based
on the specified type of current input profile. For an example, the current input depends on the user’s driving style for electric vehicles. The PDFA model of input current could classify driver’s style into different categories such as aggressive, moderate, or conservative. A more sufficient battery management scheme can be chosen based on it. Second, Adaptive updating rule is desired to enhance the performance of hybrid recursive Bayesian filtering. The measurement model constructed from the dynamic data-driven approach may not be statistical stationary (i.e., the distribution of measurement conditional on system state is not in the fixed shape) compared to the linear model with Gaussian uncertainty assumption. To avoid outlier contaminate exiting estimation result, a updating rule is desired to decide whether the coming measurement should be considered in updating step. With this adaption, the estimation accuracy should be maintained at a specified level after certain iterations. Third, although the results presented in this dissertation show that the hybrid (i.e., combined data-driven and model-based) recursive Bayes filter outperforms the solo data-driven approach, a performance comparison of the proposed method and the conventional Kalman filter with a linear (or linearized) model with additive Gaussian measurement noise is in order. By comparing solo data-driven approach, model-based algorithm, and proposed hybrid solution, a better understanding of advantages and disadvantages of each choice can be interpreted.

- **Fusion of heterogeneous data with symbolic representation**: A important motivation for heterogeneous information fusion is to obtain more accurate information and make more robust decision by considering information from multiple resources. The general framework of symbolic time series analysis provides a possibility of information fusion for different data types in a unified symbolic space. From concepts from the information theory, there are many measurements can be introduced to evaluate the dependence and causality between different symbol sequences. These properties can provide unique information from which is unavailable from each individual resources.
Appendix A

Computer Vision Techniques

This appendix presents the computer vision techniques to find the image feature descriptors in the first stage of the algorithm.

A.1 Gabor Wavelet

Wavelet analysis \cite{98, 131} is a technique developed for signal decomposition and reconstruction in the time-frequency domain subject to the constraint of the Heisenberg Uncertainty Principle. A wavelet dictionary $\mathcal{D}$ is constructed from a zero-mean mother wavelet $\psi$ as functions of the scaling parameter $s$ and translating parameter $u$:

$$\mathcal{D} := \{ \psi_{u,s}(t) = \frac{1}{\sqrt{s}} \psi\left(\frac{t - u}{s}\right) \}_{u \in \mathbb{R}, s > 0}$$  \hspace{1cm} (A.1)

The continuous wavelet transform of a signal $f$ at any scale $s$ and translation $u$ is then defined as the inner-product of the corresponding time-scale dictionary element and the original signal:

$$Wf(u, s) := <\psi_{u,s}, f> = \int_{-\infty}^{+\infty} \frac{1}{\sqrt{s}} \psi^*(\frac{t - u}{s}) f(t) dt$$  \hspace{1cm} (A.2)

Unlike a windowed Fourier Transform, which has a fixed window size (i.e. scale), the wavelet can change scale freely. The intuition is that all types of Fourier analysis are subject to the Heisenberg Uncertainty Principle $\sigma_t \sigma_\omega \geq c/2, c > 0$, which states that you cannot arbitrarily well localize a signal in both the time and frequency domains. As you localize the signal in time, the frequency becomes less well localized, and vice versa. At high frequencies one can localize the signal over shorter scales, and at low frequencies one needs longer scales to localize the signal. Thus by constructing a basis set (wavelet dictionary) which expands
high frequency components of the signal differently than low frequency components inversely to the scale of the wavelet, one can achieve the lower bound of the Heisenberg Uncertainty Principle.

This construction works for a time series signal, an image or higher dimensional data such as a 3-dimensional MRI scan. For a time series $t$ is time and for an image $t$ is the $(x, y)$ position of the image.

The Gabor wavelet \[109\] is a wavelet composed of a carrier complex exponential multiplied by an envelope Gaussian window:

$$\psi(x, y; \lambda, \theta, \phi, \sigma, \gamma) := e^{-\frac{x'^2 + y'^2}{2\sigma^2}} e^{i(2\pi x'\lambda + \phi)} \quad (A.3)$$

where $x' = x \cos(\theta) + y \sin(\theta)$ and $y' = -x \sin(\theta) + y \cos(\theta)$, i.e. pure rotations of standard Euclidean space.

### A.2 Histogram of Oriented Gradients

The Histogram of Oriented Gradients (HOG) \[110\] is a simple, yet effective algorithm for a wide variety of recognition tasks, where edge information is spatially pooled to construct low level feature descriptors that are invariant to local transformations. The first step is to compute the gradient of the image by applying the first order derivative filters $[-1, 0, 1]$ and $[-1, 0, 1]^T$. This is analogous to central differences in finite difference methods to approximate a gradient. The second step is to divide the gradient image into disjoint cells of pixels, and for each cell map the gradient of each pixel to a histogram over predefined (usually 8) directions. The magnitude of the gradient direction is used to weight contributions to the histogram. The third step is to make the feature descriptor have better invariance to illumination and shadowing. To achieve this, spatially connected cell histograms are concatenated to form blocks, and the entire block is normalized together. These blocks are then concatenated to obtain the feature descriptor. Overlapping blocks mean that an individual cell’s histogram can contribute to the final feature more than once.

### A.3 Fractal Dimension

Mathematical analysis usually deals with functions that are continuous and have continuous derivatives. This is in contrast to many situations encountered in the real world, for example stock-market security values, noise corrupted time-series signals, the coast-line of Britain, etc. Because of this, it is important to develop a mathematical formalism which can well define these scenarios. The fractal dimension, also called the box dimension, gives an intuitive explanation for understanding non-integer dimensions. Given a set (e.g. Cantor set in 1D, unit sphere in 3D,
etc), one wants to find the growth rate required to cover the set with small \( \epsilon \)-balls as the dimension of the object grows.

In the usual mathematical analysis of \( C^1 \) functions, doubling the size of an object has no effect on the density of the object. If one is in \( d \) (embedding) dimensions, then the number of \( \epsilon \)-balls required to cover the set is inversely proportional to the volume of each \( \epsilon \)-ball, i.e. \( r(\epsilon) \sim \epsilon^D \), where \( r(\epsilon) \) is the number of boxes required to cover the set and \( D(0) = d_f \) is the box-counting fractal dimension. Solving for \( D \) and taking the limit yields the box dimension:

\[
D := \lim_{\epsilon \to 0^+} \frac{\log r(\epsilon)}{\log 1/\epsilon}
\]  
(A.4)

This definition assumes all cells covering the set to be equal and the limit is independent of the cell shape. The Hausdorff dimension provides a more general definition applicable to more complex geometries. Assume the set is covered by cells \( \sigma_k \) each of size \( \epsilon_k \) (supremum of distance between any two points in cell \( \sigma_k \) is \( \epsilon_k \)). The Hausdorff content is then defined to be:

\[
m(\beta, \epsilon) := \inf_{\sigma_k} \sum_k (\epsilon_k)^\beta
\]

(A.5)

In the limit as \( \epsilon \to 0^+ \), the \( \beta \) parameter takes a unique value \( \beta_0 \) such that \( m(\beta) = \lim_{\epsilon \to 0^+} m(\beta, \epsilon) \) is both non-zero and finite. The Hausdorff dimension is then defined to be this, \( D_H = \beta_0 \). Numerically, the Hausdorff dimension and the Box-counting dimension give the same result for nearly all data sets, thus the algorithm described below to find the set dimension is based on box-counting.

Given a binary representation of an image, the following procedure is performed. Uniform grids (boxes) are generated over the image with each grid element (box) having pixel size \( 2^k \times 2^k \) for \( k = 1, 2, \ldots, 10 \). For each uniform grid, the number of grid elements (boxes) are counted which contain non-zero pixels. This defines the number of boxes required to cover the set for multiple sized boxes. Then \( D \) is interpreted as the slope of the line \( \log r(\epsilon) = D \log 1/\epsilon \) for \( k = 1, 2, \ldots, 10 \). For different scales \( k \) the gradient approximation is different, and the vector of gradient approximations was used as the feature.
Unsupervised Clustering Algorithms

In the following, we present details of three different clustering methods, namely, k-means, agglomerative hierarchical, and spectral, that have been used to make collaborative target detection. Each of these three clustering methods has been used to analyze the topological similarity of the PDFA set $\mathcal{G}$ corresponding to the ensemble of observation $\mathcal{Y}$. In all these three methods, we assess the structure of the set $\mathcal{G}$ under the two possible hypotheses, $H_0$ and $H_1$. A decision rule is reached by finding the discriminative features in the PDFA set $\mathcal{G}$ for maximum separability of the null and the alternate hypothesis. The idea is to use the gap statistics [132] of the clusters to learn whether the attributes of the set $\mathcal{G}$ belong to either the environment-only or the target-present scenario.

B.1 k-means Clustering

The $k$-means clustering [114][115][116] is used to group the elements of $\mathcal{G}$ into $k = 2$ mutually exclusive clusters. After initiating the positions of cluster centroids, each $G_i$ is assigned to the closest cluster according to a pre-defined distance measure (e.g., cosine similarity). The cluster centroids are then updated by minimizing the sum of the distance measures between every $G_i$ and its corresponding cluster centroid. The cluster assignment and distance minimization processes are iteratively repeated until the algorithm converges and there is no further change in assignment of the elements of $\mathcal{G}$ to the clusters. The algorithmic details of $k$-means clustering is provided in Algorithm 5. The positions of cluster centroids are initialized by randomizing from the set $\mathcal{G}$. To avoid local minimum due to a bad initialization, 10 replicates with individual random initial starts have been used in parallel to ensure that the algorithm returns the lowest sum of distance measures. The outputs of $k$-means clustering are locations of centroids of the two clusters and element indices of each cluster. The gap (i.e., cluster centroid distance) statistics can be used to design a threshold for declaration of target.
Algorithm 5  k-means Clustering

Require: \( \{\Pi_i\}_{i \in I} \) features extracted from different sensors \( \{\mathcal{L}_i\}_{i \in I} \), where \( I \) is the set of all available sensors.

Ensure: Centroid locations of the two clusters and element indexes for all sensors.

1: Randomly choose one sensor feature from \( \{\Pi_i\}_{i \in I} \) as the first centroid, denoted as \( c_1 \).
2: Compute distance from each sensor feature \( \Pi_i \) to \( c_1 \) as \( m(\Pi_i, c_1) \).
3: Randomly select the next centroid \( c_2 \) from \( \{\Pi_i\}_{i \in I} \) with probability \( P(\mathcal{L}_i) = \frac{m(\mathcal{L}_i, c_1)^2}{\sum_{j \in I} m(\mathcal{L}_j, c_1)^2} \).
4: Compute point-to-cluster-centroid distance of all observations to each centroid.
5: Assign each sensor \( \mathcal{L}_i \) to the cluster with its feature \( \Pi_i \) is closest to that centroid. i.e., \( \mathcal{L}_i \in C_k \), if \( k = \arg \min_k m(\mathcal{L}_i, c_k) \).
6: Compute the average of the observations in each cluster to obtain new centroid locations.
7: Repeat step 4 to 6 until cluster assignments do not change, or the maximum number of iterations is reached.

B.2  Agglomerative Hierarchical Clustering

Agglomerative hierarchical clustering [115] is a bottom-up method that generates a sparse network (e.g., a binary tree) of the PGSA set \( \mathcal{G} \) by successive addition of edges between the elements of \( \mathcal{G} \). Initially, each of the PDFA \( G_1, G_2, \ldots, G_n \) is in its own cluster \( C_1, C_2, \ldots, C_n \), where \( C_i \in \mathcal{C} \), which is the set of all clusters for the hierarchical cluster tree. In terms of the distance measured by Eqs. (2.23) through (2.25), the the pair of clusters that are nearest to each other are merged and this step is repeated till only one cluster is left. (see Algorithm 6). Figure B.1 presents two examples of binary trees obtained from environment-only and target-present scenarios from Chapter 6 as an example to make the visualization easier (details of the corresponding experiments are presented in the next section). In both examples, the sensor number 3, 4, and 5 are grouped together as a single cluster (i.e., they are subjected to high ambient illumination level compared to others). However, the distance between the cluster, consisting of sensor 3, 4, and 5, and others is significantly larger than that of the environment-only case. This tree structure displays the order of splits in the network, which can be used to find the expected behavior of the network for environment-only and target-present scenarios.

Algorithm 6  Agglomerative Hierarchical Clustering

Require: \( \{\Pi_i\}_{i \in I} \) feature vectors extracted from different sensors \( \{\mathcal{L}_i\}_{i \in I} \), where \( I \) is the set of all available sensors.

Ensure: A binary cluster tree

1: Initialize each sensor \( \mathcal{L}_1, \mathcal{L}_2, \ldots, \mathcal{L}_n \) is in its own cluster \( 1, C_2, \ldots, C_n \).
2: Find the most similar pair of clusters \( C_i \) and \( C_j \) from the proximity matrix and merge them into a single cluster.
3: Update the proximity matrix in Eq. (2.24): reduce its order by one, by replacing the individual clusters with the merged cluster.
4: Repeat step (2) and (3) until a single cluster is obtained.
Figure B.1: Examples of binary tree structure formed by multiple sensors

B.3 Spectral Clustering

Spectral clustering, in general, performs a dimensionality reduction by using spectral properties (i.e., eigenvalues and eigenvectors) of the similarity matrix generated from the feature set (extracted from the sensed data). The PDFA set $G$ is modeled as a fully connected weighted graph, i.e., every $G_i$ is a node of the graph and the weight for a edge between $i$ and $j$ is calculated from the attributes (or features) of the PDFAs $G_i$ and $G_j$. In particular, the graph properties are analyzed using two commonly used similarity functions: cosine similarity function $s^c(\cdot, \cdot)$ and the Gaussian kernel similarity function $s^g(\cdot, \cdot)$ both expressed below in Eq. (B.1).

$$s^c(a_k, a_\ell) \triangleq \frac{a_k^t a_\ell}{\|a_k\|\|a_\ell\|} \quad \text{and} \quad s^g(a_k, a_\ell) \triangleq e^{-\frac{\|a_k - a_\ell\|^2}{2\sigma^2}} \quad (B.1)$$

where the $(1 \times |Q||\Sigma|)$ vectors $a_k$ and $a_\ell$ are vectorized forms of the matrices $\hat{\Pi}_k$ and $\hat{\Pi}_\ell$, respectively; the column vector $a_\ell^t$ is the transpose of $a_\ell$; the Euclidean norm of the vector $a_k$ is denoted as $\|a_k\|$; and $\sigma$ is a non-zero real parameter. For the PDFAs $G_i$ and $G_j$, these vectors correspond to the vectorized state transition
matrices $\hat{\Pi}_i$ and $\hat{\Pi}_j$ for $G_i$ and $G_j$, respectively. As stated earlier, these $|Q| \times |\Sigma|$ matrices are vectorized as $(1 \times |Q||\Sigma|)$ row vectors by concatenating their respective $|Q|$ rows.

Once an $n \times n$ similarity matrix $A$ (i.e., $A^c \triangleq [s^c(a_k,a_\ell)]$ or $A^g \triangleq [s^g(a_k,a_\ell)]$) is constructed, which is symmetric-positive-definite, it is normalized to obtain the graph Laplacian as

$$L = I - M^{-1/2}AM^{-1/2} \quad (B.2)$$

where $M$ is a positive-definite diagonal matrix and is computed as

$$M_{ii} = \sum_j A_{ij} \quad (B.3)$$

By the Rayleigh-Ritz theorem [133], the clustering results are obtained in terms of the $|Q|$-dimensional eigenvector $f$ corresponding to the second smallest eigenvalue of the (positive semi-definite) Laplacian matrix $L$. The graph partition in clusters is constructed by using the sign of the elements of the eigenvector $f$ as an indicator function, i.e.,

$$\begin{cases} L_i \in C_1, & \text{if } f_i \geq 0 \\ L_i \in C_0, & \text{if } f_i < 0 \end{cases} \quad (B.4)$$

where $C_1$ and $C_0$ represent the two clusters in the PDFA set $\mathbb{G}$. Details of the spectral clustering are described in Algorithm 7.

### Algorithm 7  Spectral Clustering

**Require:** \{features extracted from different sensors $\{L_i\}_{i \in I}$, where $I$ is the set of all available sensors.

**Ensure:** Members indices of the two clusters $C_0$ and $C_1$.

1. Form the similarity matrix $A = \{a_{ij}\} \in \mathbb{R}^{n \times n}$, where $a_{ij} = s(L_i,L_j)$.
2. Define $M \in \mathbb{R}^{n \times n}$ to be the diagonal matrix whose $M_{ii}$ element is computed as $M_{ii} = \sum_j A_{ij}$.
3. Construct the Laplacian Matrix $L$ according to Eq. (B.2).
4. Obtain the eigenvector $f$ corresponding the second smallest eigenvalue of $L$.
5. Partition the the graph by using the sign of $f$ as indicator function as Eq. (B.4)
Principal Component Analysis

Principal component analysis (PCA) [32] is a standard technique that takes the original data and maps it to a new equivalent space defined in a rotated basis such that the greatest variance of the data is contained along the first basis dimension, the second greatest variance is contained along the second basis dimension, and so on. If the mean-square variability in the data is mostly contained in the space of first few principal components, then only the corresponding directions in the transformed space are necessary for discriminating the data, and so the dimensionality of the data is thus effectively reduced without any significant loss of information.

Principal component analysis targeted on the information such that can maximize the distance of the data by mapping data into low-dimensional feature space [32]. Let the training data sets (i.e., the ensemble of time series) $X$ be organized as an $(M \times N)$-dimensional data matrix, where $M$ is the number of training data sets and $N$ is the length of each data set (i.e., 1-D time series), where each row of $X$ is an individual data sample $x$ with zero mean. In the application of target detection and classification, $M$ is often smaller than $N$, i.e., the number of training samples are fewer than the length of the time-series. Therefore, it is numerically more efficient to analyze the $(M \times M)$ matrix $(1/M)XX^T$ that has the same nonzero eigenvalues than the $(N \times N)$ computed covariance matrix $(1/M)X^TX$ [32].

Let $v_i$ be the normalized eigenvectors of the real symmetric matrix $(1/M)XX^T$ corresponding to the (real positive) eigenvalues $\lambda_i$ that are arranged in decreasing order of magnitude, i.e., $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_M$. Let $m$ be the smallest integer such that $\sum_{i=1}^{m} \lambda_i \geq \eta \sum_{i=1}^{M} \lambda_i$ where $1 \leq m \leq M$, where the threshold parameter $\eta$ is a real positive fraction close to one. The corresponding (normalized) eigenvectors $u_i$ in the original data space are obtained in terms of $v_i$ and $\lambda_i$ as follows [32][130]:

$$u_i = \frac{1}{\sqrt{M\lambda_i}}X^Tv_i, \ i = 1, 2, \ldots, m.$$  

(C.1)
The PCA projection matrix $W \in \mathbb{R}^{N \times m}$ is then obtained by grouping the computed eigenvectors as follows:

$$W = [u_1 u_2 \ldots u_m] \quad (C.2)$$

The PCA-based feature vector $p \in \mathbb{R}^{1 \times m}$ for a given (train or test) time-series $x \in \mathbb{R}^{1 \times N}$, is then computed as follows:

$$p = xW. \quad (C.3)$$
Bibliography

*J. Math. pures appl.*, vol. 4, pp. 27–73, 1898.


BILE Mobile Computing and Communications Review*, vol. 5, no. 1, pp. 3–55, 


time series, with implications for streaming algorithms,” in *Proceedings of 
the 8th ACM SIGMOD workshop on Research issues in data mining and 

series by symbolic dynamics,” *Physical Review E*, vol. 64, no. 5, p. 051104, 

symbolic time series analysis,” *Structural Control and Health Monitoring*, 


analysis,” *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 14, 


126


129


Vita

Yue Li

Yue Li was born in Fu Jian province, China on November 22, 1987. He received the B.S. degree in mechanical engineering from Shanghai Jiao Tong University in China in July 2010. In his senior year of undergraduate study, he attended the Global Engineering Alliance for Research and Education (GEARE) program at Karlsruhe Institute of Technology (KIT) in Germany as an exchange student. Yue Li joined the Pennsylvania State University as a graduate student in the Department of Mechanical Engineering in August 2010 and received the M.A. degree in Mathematics in May 2016. His doctoral research in Mechanical Engineering is under the guidance of Professor Asok Ray and he successfully defended his dissertation on June 15, 2016. Yue Li will join as a research engineer at FARO Technologies Inc. at Lake Mary, Florida in July 2016.