SYMBOLIZATION AND INFORMATION-THEORETIC MODELLING OF DYNAMIC DATA-DRIVEN APPLICATION SYSTEMS

SOUMALYA SARKAR, PRITTHI CHATTOPDHYAY and ASOK RAY

The Pennsylvania State University
University Park
PA 16802
USA
e-mail: sms388@gmail.com
pritthichatterjee@gmail.com
axr2@psu.edu

Abstract

Probabilistic finite state automata (PFSA) have been widely used as an analysis tool for modelling of physical systems. In this context, the concept of symbolic time series analysis (STSA) has led to the development of signal representation tools in the paradigm of dynamic data-driven application systems (DDDAS). In STSA, time series of sensor signals are partitioned to generate symbol strings for construction of PFSA. Although various methods for construction of PFSA from symbol strings have been reported in literature, similar efforts have not been expended on identification of a symbol alphabet for partitioning of time series so that the symbol strings can be optimally generated from time series in

2010 Mathematics Subject Classification: 37B10, 62-07, 94A17

Keywords and phrases: dynamic data-driven application systems, symbolic time series analysis, information theory, optimized partitioning.

This work has been supported in part by the U.S. Air Force Office of Scientific Research under Grant No. FA9550-15-1-0400. Any opinions findings and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the sponsoring agencies.

Communicated by Marcelo Bourguignon.

Received January 17, 2017

a specified sense. The paper addresses this critical issue and proposes an information-theoretic procedure for partitioning of time series to extract low-dimensional features. The key idea is robust and optimal identification of boundary locations of the partitioning segments via maximization of the mutual information between the state probability vector of PFSA and the members of the pattern classes. The proposed algorithms of alphabet-size selection and time-series partitioning have been validated by two examples. The first example addresses parameter identification in a simulated Duffing system with sinusoidal input excitation. The second example is built upon an ensemble of time series of chemiluminescence data to predict lean blowout (LBO) phenomena in a laboratory-scale swirl-stabilized combustor apparatus.

Nomenclature

(1) A : Amplitude of sinusoidal excitation in the Duffing system (Example #1)

(2) $\mathscr{C}$: Set of pattern class labels.

(3) $|\mathscr{C}|$: Number of (finitely many) pattern class labels ($|\mathscr{C}| \geq 2$) in $\mathscr{C}$.

(4) c_j : j-th class label in $\mathscr{C}$.

(5) D : Depth parameter of a D-Markov machine (D is a positive integer).

(6) $H(X)$: Entropy of a random variable (rv) X with probability mass/density function p_X.

(7) $H(Y|X)$: Entropy of a rv Y conditioned on another rv X with conditional probability mass/density function $p_{Y|X}$.

(8) $I(X;Y)$: Mutual information between two rv's X and Y with joint probability mass/density function p_{XY}.

(9) $I_{\max}$: Positive scalar threshold of mutual information (to be specified by the user).

(10) $\mathscr{I}_c$: Set of indices of training samples belonging to class label $c \in \mathscr{C}$.

(11) L : Number of (finitely many) partition segments in the range space of time series (possible choice: $L \gg |\sum|$).

(12) M : Sample size in the computation of the mutual information I for selection of boundary locations.

(13) m : Average length of time series data.

(14) $N(\bullet, \bullet)$: Gaussian distribution.

(15) n : Total number of training time series belonging to all pattern classes in $\mathscr{C}$.

(16) n_c : Number of training time series belonging to the class label $c \in \mathscr{C}$.

(17) $\mathscr{P}$: Feature vector extracted from time series data.

(18) $\mathcal{P}$: Space of all possible partitioning boundaries.

(19) P_E : Probability of incorrect class estimation (to be minimized).

(20) P_{95} : 95% percentile.

(21) $\mathbf{P}$: State probability transition matrix of PFSA of dimension $|Q| \times |Q|$.

(22) $\mathbf{p}$: State probability vector of PFSA of dimension $|Q|$ (used as a feature).

(23) Q : Set of states in FSA and PFSA.

(24) $|Q|$: Number (finitely many) states in FSA and PFSA $(|Q| \geq |\sum|)$.

(25) q : A state in FSA and PFSA.

(26) S : Covariance matrix of multivariate Gaussion distribution.

(27) $\boldsymbol{s}$: Symbol string of finite length $(s \in \sum^*)$.

(28) t : Time in the fast scale.

(29) y : Duffing system output variable (Example #1).

(30) α : Parameter in the Duffing system (Example #1).

(31) β : Parameter in the Duffing system (Example #1).

(32) δ : State transition map in FSA and PFSA.

(33) $\delta^{\star}$: Extended state transition map in FSA and PFSA.

(34) ϵ : Empty string of symbols (zero-length).

(35) η_{stop} : Positive scalar threshold of normalized increment in mutual information (to be specified by the user).

(36) $\Lambda_{|\sum|}$: Set of partition locations for alphabet $\sum$.

(37) $\Lambda^{*}_{|\sum|}$: Suboptimal set of partition locations for alphabet $\sum$.

(38) λ : Arbitrary partition location.

(39) λ^{*} : Suboptimal partition location.

(40) ν : Number of counts in PFSA parameter calculation.

(41) $\sum$: Alphabet of symbols for symbolization of time series.

(42) $\left|\sum\right|$: Number of symbols in the symbol alphabet.

(43) $\sum^{*}$: Set of all finite-length symbol strings in $\sum$.

(44) σ : Symbol in an alphabet.

(45) ϕ : Equivalence (i.e., fuel/air) ratio in combustion (Example #2).

(46) ω : Frequency of sinusoidal excitation in the Duffing system (Example #1).

(47) Υ : Set of $(L-1)$ boundary locations in the range space of time series (excluding the end points).

Acronym

- ASSP : Analytic signal space partitioning.
- DDDAS : Dynamic data-driven application systems.
- DFSA : Deterministic finite state automata.
- FSA : Finite state automata.
- ILBO : Impending lean blowout.
- LBO : Lean blowout of combustion.
- MEP : Maximum entropy partitioning.
- PFSA : Probabilistic finite state automata.
- PLBO : Progressive lean blowout.
- SFNNP : Symbolic false nearest neighbourhood partitioning.
- STSA : Symbolic time series analysis.
- UP : Uniform partitioning.
- WSP : Wavelet space partitioning.

1. Introduction

Symbolic time series analysis (STSA) ([1], [2]) has been used for constructing statistical models of (possibly nonlinear) dynamical systems, which rely on temporal and spatial discretization based on the fundamental concepts of symbolic dynamics [3]. Along this line, Ray and coworkers ([4], [5], [6], [7], [8], [9], [10]) have developed data-driven procedures for generation of probabilistic finite state automata (PFSA) models, where the major role of STSA is to serve as a feature extraction tool for information compression and pattern classification in dynamic data-driven application systems (DDDAS) (see [11] and references therein for details of the DDDAS concept); this procedure has been used for early detection of anomalous behaviour as well as for pattern

recognition in diverse physical systems (e.g., see [12], [13], [14]). While extensive research work (e.g., see [15], [16]) has been reported for investigation of the properties and variations of transformation from a symbol space to a feature space in the conversion of symbol strings into PFSA models, similar efforts have not been expended on how to find an appropriate alphabet size for symbolization of time series (e.g., see [17]) so that the symbol sequences can be optimally (or suboptimally) generated from the respective time series data in a specified sense [18]. In recent literature of STSA-based feature extraction for the discovery of anomaly patterns (e.g., [4], [10]), PFSA models have been constructed by first partitioning the time series at a single reference condition to obtain a reference pattern and then generating subsequent patterns from the time series at other conditions with the same partitioning. Apparently, the issue of alphabet size selection for symbolization of time series has not been duly addressed.

The current paper is an extension of the preliminary version of the work reported earlier as a short paper [18] and specifically addresses the critical issues of:

- Robust alphabet size selection.
- Performance comparison.

Major contributions of the work reported in this paper are concisely stated below.

(1) *Partitioning of time series in a way that maximizes the mutual information between the symbolic dynamic feature* (*e.g., the state probability vector of PFSA*) *and the pattern class to which it belongs. The details of analysis are significantly more rigorous than those in* [18].

(2) *Development of robust algorithms of alphabet size selection for symbolization of time series so that the symbol strings can be optimally generated in a specified sense.*

(3) *Performance comparison of the proposed partitioning technique with another commonly used method, namely, maximum entropy partitioning* (*MEP*).

(4) *Validation of the proposed concepts on*:

● *Simulated data collected from a Duffing system with sinusoidal excitation, and*

● *Experimental data from a laboratory-scale swirl-stabilized combustor for lean blowout* (*LBO*) *prediction. Algorithm validation with experimental results is significantly more detailed than that reported in* [18].

The paper is organized into five sections (including the current section) and an Appendix. Section 2 presents a brief background of symbolic time series analysis (STSA) along with basic concepts of entropy and mutual information [19]. The alphabet size selection scheme is elaborated in Section 3 along with its key features. Section 4 validates the proposed concepts on a simulation model of the second order forced Duffing equation as well as on experimental data from a laboratory-scale combustor for lean blowout (LBO) prediction. Section 5 summarizes and concludes the paper along with recommendations for future research. Appendix consists of four algorithms used in the paper.

2. Mathematical Preliminaries and Concepts

This section briefly presents the mathematical preliminaries and concepts of symbolic time series analysis (STSA) [1], [4], [10] and related information-theoretic definitions [19] in the context of feature extraction and pattern classification.

A. Symbolic time series analysis (STSA)

Although the methodology of symbolic time series analysis (STSA) has been extensively reported in literature (e.g., [1], [4], [10]), a brief outline of the underlying concepts is succinctly presented here for completeness of the paper.

(1) Partitioning of time series space for generation of symbol strings

Steuer et al. ([20]) reported a comparison of maximum entropy partitioning and uniform partitioning, where it is concluded that maximum entropy partitioning is, in general, a better tool for change detection in symbolized time series than uniform partitioning. Buhl and Kennel ([17]) reported symbolic false nearest neighbour partitioning (SFNNP) to optimize generating partitions by avoiding topological degeneracy. However, SFNNP may become extremely computationally intensive if the phase-space dimension of the underlying dynamical system is large. Furthermore, if the time series data become noise-corrupted, the symbols in SFNN tend to grow rapidly in number and may erroneously require a large number of redundant symbols to capture pertinent information on the system dynamics. Rajagopalan and Ray ([5]) introduced wavelet space partitioning (WSP), where the wavelet transform largely alleviates the above shortcoming and is particularly effective with noisy data for large-dimensional dynamical systems; Subbu and Ray ([6]) introduced Hilbert-transform-based analytic signal space partitioning (ASSP) as an alternative to WSP. Nevertheless, these partitioning techniques primarily attempt to provide an accurate symbolic representation of the underlying dynamical system under a given quasi-stationary condition, rather than trying to capture the data-evolution characteristics due to an anomaly in the system. Jin et al. ([21]) reported the theory and validation of a wavelet-based feature extraction tool that used maximum entropy partitioning of the space of wavelet coefficients. Even if this partitioning is optimal (e.g., in terms of maximum entropy or some other criteria) under nominal conditions, it may not remain optimal at other conditions. Along this line Sarkar et al. ([22]) proposed a time-series partitioning procedure to extract low-dimensional features from time series while optimizing the class separability; however, this method is strongly dependent on the choice of the classifier tool. The goal of the work reported in the current paper is to

overcome the difficulties of the above-mentioned partitioning methods with the objective of making STSA a robust data-driven feature-extraction tool based on an information-theoretic concept.

Symbol string generation from time series data is posed as a two-time-scale problem, as depicted in Figure 1. The *fast scale* is related to the response time of the process dynamics. Over the span of data acquisition, dynamic behaviour of the system is assumed to remain invariant, i.e., the process is statistically quasi-stationary on the fast scale. In contrast, the *slow scale* is related to the time span over which non-stationary evolution of the system dynamics may occur. It is expected that the features extracted from the fast-scale data will depict statistical changes between two different slow-scale epochs if the dynamical system has undergone a change. The method of extracting features from quasi-stationary time series on the fast scale is comprised of the following steps:

- A set of (sensor) time series data is generated at slow-scale epochs from a physical process. The approximate length of each time series data is m, which is sufficiently large for convergence of its statistical parameters within specified thresholds. (Note: individual time series need not be of exactly the same length.)

- Encoding of the data space of time series is accomplished by introducing a partition that consists of finitely many mutually exclusive and exhaustive cells. Let the j-th cell be labelled by a symbol $\sigma_j \in \sum$ and data points of the time series, which visit the j-th cell, are denoted as σ_j. The finite set $\sum = \{\sigma_0, \ldots, \sigma_{|\sum|-1}\}$ is called an alphabet and its cardinality $|\sum| \geq 2$ is called the alphabet size. This process of coarse graining can be executed by uniform, maximum entropy, or any other scheme of partitioning [5] to enable transformation of a time series into a symbol string. (Note: $|\sum| \ll m$.)

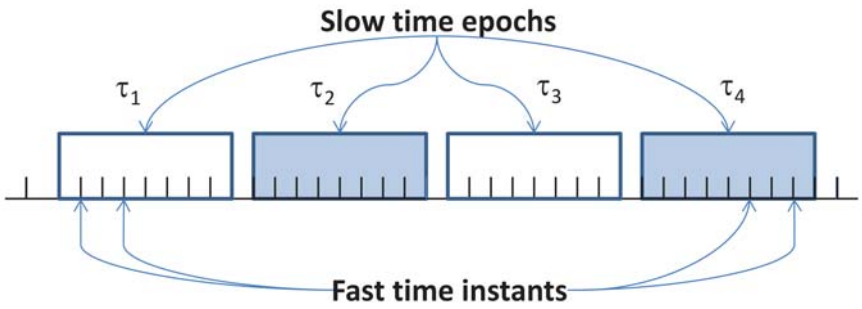


Figure 1. Pictorial view of two time scales: (i) Slow scale of process evolution and (ii) Fast instants of data acquisition.

(2) Construction of probabilistic finite state automata from symbol strings

A probabilistic finite state automaton (PFSA) is then constructed from the symbol string (see Subsection 2-A2 below). To compress the information further, the state probability vector of the PFSA (see Definition 2.5 below) can be used as an extracted feature. The following standard definitions are used for PFSA construction.

Definition 2.1 (Finite state automaton)**.** A finite state automaton (FSA), having a deterministic algebraic structure, is called a DFSA that is a triple $(\sum, Q, \delta)$ [10], where

- $\sum$ is a (nonempty) finite alphabet with cardinality $|\sum|$;
- Q is a (nonempty) finite set of states with cardinality $|Q|$;
- $\delta : Q \times \sum \rightarrow Q$ is the state transition map.

Definition 2.2 (Symbol block)**.** A symbol block, also called a word, is a finite-length string of symbols belonging to the alphabet $\sum$, where the length of a word $w \triangleq s_1 s_2 \cdots s_\ell$ with $s_i \in \sum$ is $|w| = \ell$, and the length of the empty word ϵ is $|\epsilon| = 0$. The parameters of a DFSA are extended as:

● The set of all words constructed from symbols in $\sum$, including the empty word ϵ, is denoted as $\sum^*$.

● The set of all words, whose suffix (respectively, prefix) is the word w, is denoted as $\sum^\star w$ (respectively, $w\sum^\star$).

● The set of all words of (finite) length ℓ, where $\ell > 0$, is denoted as $\sum^\ell$.

Definition 2.3 (Extended map)**.** The extended state transition map $\delta^\star : Q \times \sum^\star \to Q$ transfers one state to another through finitely many transitions such that, for all $q \in Q, \sigma \in \sum$ and $w \in \sum^\star$,

$$\delta^\star(q, \epsilon) = q \text{ and } \delta^\star(q, w\sigma) = \delta(\delta^*(q, w), \sigma),$$

where $w\sigma$ is suffixing of the word w by the symbol σ.

Definition 2.4 (Irreducible DFSA)**.** A DFSA G is said to be irreducible if, for all $q_1, q_2 \in Q$, there exists a word $w_{1,2} \in \sum^*$ such that $q_1 = \delta^*(q_2, w_{1,2})$.

Definition 2.5 (PFSA)**.** A probabilistic finite state automaton (PFSA) K is a pair (G, π), where

● The deterministic FSA G is called the underlying FSA of the PFSA K;

● The probability *map* $\pi : Q \times \sum \to [0, 1]$ is called the morph function (also known as symbol generation probability function) that satisfies the condition: $\sum_{\sigma \in \sum} \pi(q, \sigma) = 1$ for all $q \in Q$.

Equivalently, a PFSA is a quadruple $K = (\sum, Q, \delta, \pi)$, where

• The alphabet $\sum$ of symbols is a (nonempty) finite set, i.e., $0 < |\sum| < \infty$, where $|\sum|$ is the cardinality of $\sum$;

• The set Q of automaton states is (nonempty) finite, i.e., $0 < |Q| < \infty$, where $|Q|$ is the cardinality of Q;

• The state transition function $\delta : Q \times \sum \to Q$;

• The morph function $\pi : Q \times \sum \to [0, 1]$, where $\sum_{\sigma \in \sum} \pi(q, \sigma) = 1$ for all $q \in Q$. The morph function π generates the $(|Q| \times |\sum|)$ morph matrix $\prod$.

• A combination of the morph function π and the state transition map δ is used to construct the $(|Q| \times |Q|)$ state probability transition matrix $\boldsymbol{P}$ that is an irreducible stochastic matrix [23].

• The state probability vector $\boldsymbol{p}$ is the sum-normalized left eigenvector of $\boldsymbol{P}$ with respect to its (unique) unity eigenvalue [23].

Definition 2.6 (Extended morph function)**.** The morph function $\pi : Q \times \sum, \to [0, 1]$ of PFSA is extended as $\pi^\star : Q \times \sum^* \to [0, 1]$ such that, for all $q \in Q$, $\sigma \in \sum$ and $w \in \sum^\star$,

$$\pi^\star(q, \epsilon) = 1 \text{ and } \pi^\star(q, w\sigma) = \pi^\star(q, w) \times \pi(\delta^\star(q, w), \sigma),$$

where $w\sigma$ is suffixing of the word w by the symbol σ.

(3) Construction of *D*-Markov machines

This subsection introduces the pertinent definitions that are necessary to construct a *D*-Markov machine, which serves as a feature extractor for pattern classification; the feature extraction performance of

D-Markov machines has been shown to be comparable (and often superior) to other commonly used tools [13], [14]. The PFSA model of the *D*-Markov machine generates symbol strings $\{s_1 s_2 \cdots s_\ell \in \sum^\ell : \ell \in \mathbb{N}\}$ on the underlying Markov process. The morph function π (see Definition 2.5 implicitly alludes to the fact that the PFSA satisfies the Markov condition, where generation of a symbol *only* depends on the current state. However, if the state is unknown, the next symbol generation may depend on the past history of the symbols generated by the PFSA. In the PFSA model, a transition from one state to another is independent of the previous history of states. Therefore, the states and transitions form a Markov process, which is a special class of hidden Markov models (HMM) [15], [16]. In general, for PFSA construction from a symbol string, the states are implicit and generation of the next symbol may depend on the complete history of symbols. In the construction of a *D*-Markov machine [4], [10], generation of the next symbol depends only on a *finite* history of at most D consecutive symbols, i.e., a symbol block of length not exceeding D. A formal definition of the *D*-Markov machine follows.

Definition 2.7 (*D*-Markov machine [4], [10])**.** A *D*-Markov machine is a PFSA in the sense of Definition 2.5 and it generates symbols that solely depend on the (most recent) history of at most D symbols in the string, where the positive integer D is called the depth of the machine. Equivalently, a *D*-Markov machine is a statistically stationary symbol string $\cdots s_{-1} s_0 s_1 \cdots$, where each s_i is a symbol in $\sum$ and the probability of occurrence of a new symbol depends only on the last D symbols, i.e.,

$$P[s_n \mid \cdots s_{n-D} \cdots s_{n-1}] = P[s_n \mid s_{n-D} \cdots s_{n-1}]. \tag{1}$$

Consequently, for $w \in \sum^D$ (see Definition 2.2), the equivalence class $\sum^\star w$ of all (finite-length) words, whose suffix is w, is qualified to be a *D*-Markov state that is denoted as w.

It is noted that *D*-Markov machines belong to the class of shifts of finite type [3], i.e., shift spaces that can be described by a finite set (possibly the empty set) of forbidden symbol blocks. Given a probability distribution, construction of an exact PFSA model appears to be computationally infeasible. Furthermore, a *D*-Markov machine is a finite-history automaton, where the past information is embedded as words of length D or less. That is, if w is a word of length D or more, then $\delta^*(q, w)$ in a *D*-Markov machine is independent of the state q. Whatever the initial state q is, the finite string of transitions represented by the word $w \in \sum^D$ always leads to the same terminal state that could be represented by the word w itself. Consequently, in a *D*-Markov machine, a word $w \in \sum^D$ can be associated to a state of the machine. Moreover, the state transition map δ can be automatically constructed from the words that correspond to individual states. Then, the morph functions π (that are the entries of the morph matrix $\prod$) can be estimated by counting the frequency of occurrences of the individual symbols emanating from each state ([9], [10]) as described below.

Let ν_{ij} be the number of times that a symbol σ_j is generated from the state q_i upon observing a symbol string. The maximum a posteriori probability (MAP) estimate of the probability map for the PFSA is computed by frequency counting [10] as:

$$\hat{\pi}_{MAP}(q_i, \sigma_j) \triangleq \frac{1 + \nu_{ij}}{|\sum| + \sum_{\ell=1}^{|\Sigma|} \nu_{i\ell}}. \tag{2}$$

The rationale for initializing each element of the count matrix 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 $\hat{\pi}_{MAP}(q, \sigma) = \frac{1}{|\sum|}$, $\forall \sigma \in \sum$, i.e., the uniform distribution of event generation at the state q. The above procedure guarantees that the PFSA, constructed from a (finite-length) symbol string, must have an (elementwise) strictly positive morph matrix $\prod$.

(4) Entropy and mutual information

Given that entropy is a measure of uncertainty of random variables, let X and Y be two discrete random variables that are defined on a probability space and that take values from respective finite sets $\mathbb{X}$ and $\mathbb{Y}$. Let the probability mass functions be denoted as: $p_X(x) \triangleq \Pr\{X = x\}$, $x \in \mathbb{X}$, and $p_Y(y) \triangleq \Pr\{Y = y\}$, $y \in \mathbb{Y}$; let their joint probability mass function be denoted as: $p_{XY}(x, y) \triangleq \Pr\{X = x, Y = y\}$, and conditional probability mass function be denoted as: $p_{Y|X}(y|x) \triangleq \Pr\{Y = y|X = x\}$. Then, Shannon entropy of X is defined in units of bits [19] as:

$$H(X) = -\sum_{x \in \mathbb{X}} p_X(x) \log_2 p_X(x). \tag{3}$$

Similarly, Shannon entropy of Y conditioned on X is defined as:

$$H(Y|X) = -\sum_{x \in \mathbb{X}} \sum_{y \in \mathbb{Y}} p_{XY}(x, y) \log_2 p_{Y|X}(y|x). \tag{4}$$

Mutual information of X and Y, which is the information commonly shared between them, is defined as:

$$I(X; Y) = \sum_{x \in \mathbb{X}} \sum_{y \in \mathbb{Y}} p_{XY}(x, y) \log_2 \frac{p_{XY}(x, y)}{p_X(x) p_Y(y)}. \tag{5}$$

The mutual information is symmetric relative to X and Y and it has the following relationship with entropy [19]:

$$I(X; Y) = I(Y; X) \text{ and } I(X; Y) = H(Y) - H(Y|X). \tag{6}$$

It is noted that the larger the mutual information $I(X; Y)$ is, more closely correlated X and Y are.

For continuous random variables, Shannon entropy and mutual information are defined in terms of the respective probability density functions as:

$$H(X) = -\int_{x \in X} p_X(x) \log_2 p_X(x) dx; \tag{7}$$

$$I(X;\, Y) = \int_{(x,\, y)\in X\times Y} p_{XY}(x,\, y) \log_2 \frac{p_{XY}(x,\, y)}{p_X(x) p_Y(y)}\, dxdy. \tag{8}$$

3. Problem Formulation

This section describes the procedure of selection of alphabet size and partitioning of time series for feature extraction in a PFSA framework, followed by pattern classification.

A. Alphabet size selection for symbolization of time series

This subsection uses the state probability vector $\boldsymbol{p}$ (see Definition 2.5) as a feature for pattern classification (i.e., prediction of the class from a given feature); an alternative choice for a feature could be the morph matrix $\prod$ (e.g., see [4], [10]). The finite set of class labels are specified as $\mathscr{C} = \{c_0, c_1, \cdots, c_{(|\mathscr{C}|-1)}\}$, where $|\mathscr{C}| \geq 2$ is selected a priori, for pattern classification.

The success of a time-series partitioning methodology depends on how much information embedded in the time series is captured by the PFSA (e.g., the state probability vector $\boldsymbol{p}$). A lower bound of the probability P_E of incorrect estimation $\hat{\mathscr{C}}$ of the true class $\mathscr{C}$ (i.e., $P_E = \Pr[\hat{\mathscr{C}} \neq \mathscr{C}]$) is obtained from the weak form of Fano's inequality [19] as:

$$P_E \geq \frac{H(\mathscr{C}|\mathscr{P}) - 1}{\log_2|\mathscr{C}|} = \frac{H(\mathscr{C}) - I(\mathscr{P};\, \mathscr{C}) - 1}{\log_2|\mathscr{C}|}, \tag{9}$$

where the random vector $\mathscr{P}$ represents the input feature extracted from a given time series whose pattern class (belonging to $\mathscr{C}$) may be unknown; $H(\mathscr{C})$ and $H(\mathscr{C}|\mathscr{P})$ are the entropy and conditional entropy of pattern class $\mathscr{C}$, respectively; and $I(\mathscr{P};\, \mathscr{C})$ is the mutual information between the input feature and the pattern class. Since $H(\mathscr{C})$ and $|\mathscr{C}|$ are fixed, the lower bound of P_E is minimized when $I(\mathscr{P};\, \mathscr{C})$ reaches its maximum.

As the continuity and higher order differentiability of the partitioning function in the range space of mutual information is not guaranteed nor adequately analyzed, a sequential search-based technique has been adopted for optimized alphabet size selection instead of a gradient-based optimization procedure. The process of constructing a search space is started with an initial fine grid size, where each of the grid boundaries denotes a possible cell boundary of trial partitions. These boundaries are obtained via either uniform partitioning (UP) or maximum entropy partitioning (MEP) [5] of the range space of the time series, where the respective algorithms are listed in the Appendix. (Note: MEP provides a larger number of partition boundaries at data-rich zones and smaller number of partition boundaries at sparse zones, resulting in an approximately equal number of data points in each partition segment, whereas UP divides the range space uniformly.)

At the beginning, the time series is divided into L regions on its signal space that is marked by $(L-1)$ boundaries Υ (excluding the end points) for search via MEP or UP. The positive integer L with $m > L > |\sum|$ is to be selected by the user (see Algorithm 1 and Algorithm 2 in the Appendix). Thus, for an alphabet $\sum$, there are $(|\sum|-1)$ partitioning boundaries to choose from $(L-1)$ possibilities, which is equivalent to making one choice in the space of all possible partitioning vectors (i.e., selecting one $(|\sum|-1)$-dimensional partitioning vector from ${}^{(L-1)}C_{(|\Sigma|-1)}$ different choices). It follows that the space $\mathcal{P}$ of all possible partitioning boundaries may become significantly large as L and $|\sum|$ are increased. For example, if $L \gg |\sum|$, then computational complexity increases approximately by a factor of $L/|\sum|$ as $|\sum|$ is increased by one.

The cost function to be maximized is the scalar-valued mutual information $I(\mathscr{P}; \mathscr{C})$, while decisions are made in the space $\mathcal{P}$ of partitioning boundaries. The cost is dependent on a specific partitioning Λ, because the extracted feature $\mathscr{P}$ is a function of the chosen $(|\sum| - 1)$ -dimensional partitioning vector Λ; the cost is denoted by $I(\mathscr{P}(\Lambda); \mathscr{C})$. This suboptimal partitioning scheme involves sequential estimation of the elements of the partitioning vector Λ.

The partitioning process is initiated by searching the optimal cell boundary that divides the data range into two cells, i.e., $\Lambda_2 = \{\lambda_1\}$, where λ_1 is optimized as:

$$\lambda_1^* = \arg\max_{\lambda_1 \in \Upsilon} I(\mathscr{P}(\Lambda_2); \mathscr{C}). \tag{10}$$

Now, the two-cell optimal partitioning is given by $\Lambda_2^* = \{\lambda_1^*\}$. The next step is to partition the data range into three cells as Λ_3) by dividing either of the two cells of Λ_2^* by placing a new partition boundary at λ_2, where λ_2 is evaluated as:

$$\lambda_2^* = \arg\max_{\lambda_2 \in \Upsilon \setminus \Lambda_2^*} I(\mathscr{P}(\Lambda_3); \mathscr{C}), \tag{11}$$

where $\Lambda_3 = \{\lambda_1^*, \lambda_2\}$. The optimal 3-cell partitioning is obtained as $\Lambda_3^* = \{\lambda_1^*, \lambda_2^*\}$. In this (local) optimization procedure, the cell that provides the largest increment in $I(\mathscr{P}; \mathscr{C})$ upon further segmentation ends up being partitioned. Iteratively, this procedure is extended to obtain the parameter $|\sum|$ of cell partitioning as follows:

$$\lambda_{|\sum|-1}^* = \arg\max_{\lambda_{|\sum|-1} \in \Upsilon \setminus \Lambda_{|\sum|-1}^*} I(\mathscr{P}(\Lambda_{|\sum|}); \mathscr{C}), \tag{12}$$

where $\Lambda_{|\sum|} = \Lambda_{|\sum|-1}^* \cup \{\lambda_{|\sum|-1}\}$ and the optimal $|\sum|$ is given by $\Lambda_{|\sum|}^* = \Lambda_{|\sum|-1}^* \cup \{\lambda_{|\sum|-1}^*\}$.

In this optimization procedure, the mutual information increases monotonically with additional sequential operation provided that the mutual information is computed correctly. This monotonicity property allows formulation of a rule for termination of the sequential optimization algorithm. The process of creating additional partitioning cells is stopped if the normalized mutual information, relative to $H(\mathscr{C})$ with a uniform class prior, crosses a specified positive threshold $I_{\max} \in [0, 1]$. The stopping criterion is: $\Lambda^*_{|\sum|}$ is the optimal partitioning and $|\sum|$ is the optimal alphabet size if

$$\frac{I(\mathscr{P}(\Lambda^*_{|\sum|}); \mathscr{C})}{H(\mathscr{C})} > I_{\max}. \tag{13}$$

An alternative form of the stopping criterion in Equation (13) is based on the normalized mutual information gain being less than a specified positive scalar threshold η_{stop} as stated below:

$$I(\mathscr{P}(\Lambda^*_{|\sum|}); \mathscr{C}) - I(\mathscr{P}(\Lambda^*_{|\sum|-1}); \mathscr{C}) \leq \eta_{stop}. \tag{14}$$

In contrast to the direct search of the entire partitioning space, the computational complexity in the current setting increases linearly with $|\sum|$. Thus, the proposed approach would allow a finer grid size for the partitioning search with relatively low computational complexity.

The pseudocode of the above procedure is listed as Algorithm 3 in the Appendix. Figure 2 elucidates an outline of the alphabet size selection method explained above for $|\mathscr{C}| = 2$. Labelled time series data from the training set are fed into the "alphabet size selection" block in Figure 2. For each possible partitioning under consideration, low-dimensional feature vectors are generated by symbolization of time series and subsequent PFSA construction (see Subsection 2-A2 and the "STSA feature extraction" sub-block on the left side within the "alphabet size selection" block), which together with the provided class labels (in the simplex plane sub-block on the right side within the "alphabet size

selection" block) are then used to compute mutual information between the class and the feature via Parzen window method [24]. The computed mutual information is then fed back to the "STSA feature extraction" sub-block on the left side within the "alphabet size selection" block in Figure 2. Based on the feedback of the mutual information for each possible partitioning at a given alphabet size, the partitioning is updated to further increase the mutual information between the class and the features by including an additional partition boundary at this step. The process is repeated until the stopping criterion is satisfied. Then, the optimal partitioning is used to extract features in both training and testing phases. In this way, a classifier is trained for the testing data to be classified. Figure 2 shows a sample step in the process, where each time series is converted into a feature vector and the adjacent simplex plane exhibits the distribution of classes in the feature space.

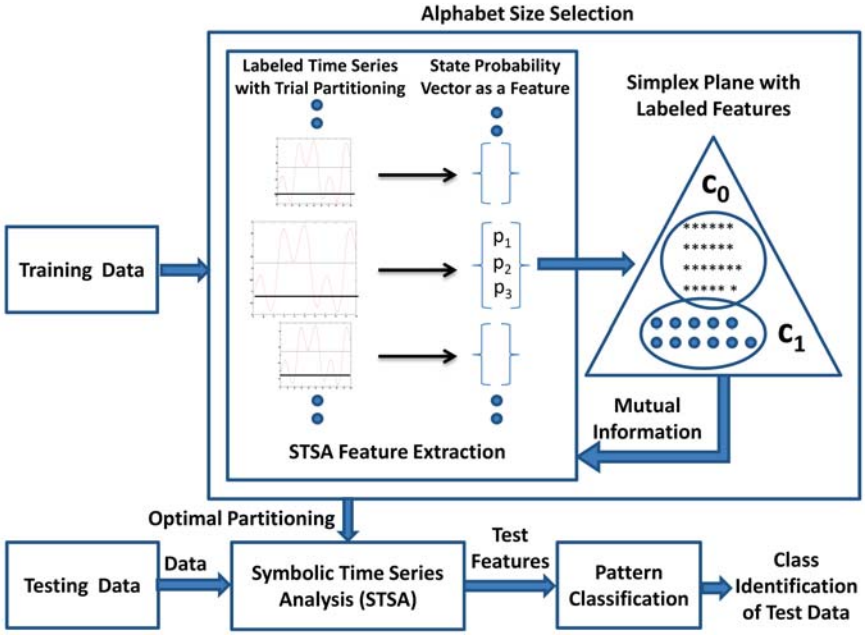


Figure 2. An information-theoretic framework for time-series partitioning and alphabet size selection.

B. Robustness

This subsection addresses robustness of classification performance when perturbed in partition locations. In this procedure, a zero-mean Gaussian noise is added to generate samples of random boundary locations. If a large number of samples are drawn, the effect of the perturbations is realized from the statistical characteristics of the set of mutual information values corresponding to each sample. To this end, i-th partition boundary location is drawn as samples from a Gaussian distribution $N(\lambda_i, \sigma_i)$ (i.e., with mean λ_i and standard deviation σ_i). In particular, σ_i's are chosen to be fractions of $\min(\lambda_i - \lambda_{i-1}, \lambda_{i+1} - \lambda_i)$. At each step, M samples are drawn from the distribution centered at a partition location λ_i that is not yet included in the partition set, i.e., the j-th independent and identically distributed (iid) sample $\lambda_i^j \sim N(\lambda_i, \sigma_i)$, $j = 1, \cdots, M$. The mutual information corresponding to the features extracted after incorporating λ_i^j into the existing partition set is obtained as $(I(\mathscr{P}(\Lambda_i^j); \mathscr{C})$, where $\Lambda_i^j = \Lambda_{i-1}^* \cup \{\lambda_i^j\}$.

The mutual information of the feature vectors resulting from adding λ_i into the existing partition set is taken to be a specified (95-th in this paper) percentile of the set of mutual information values corresponding to M samples drawn from the distribution centered at λ_i, i.e.,

$$I(\mathscr{P}(\Lambda_i); \mathscr{C}) = P_{95}\{I(\mathscr{P}(\Lambda_i^j); \mathscr{C}),\ j = 1, 2, \cdots, M\}. \tag{15}$$

Hence, at the i-th step, the (suboptimal) partition location λ_i^* is obtained as:

$$\lambda_i^* = \arg\max_{\lambda_i}((I(\mathscr{P}(\Lambda_i); \mathscr{C}))). \tag{16}$$

C. Estimation of mutual information with Parzen window

This subsection explains usage of the Parzen window method ([24], [25]) for estimation of mutual information at each step of sequential optimization. In classification problems, a class has discrete values while the input features $\mathscr{P}$ (i.e., state probability vectors $\mathbf{p}$ of PFSA) are usually continuously varying. Similar to its usage in Equation (9), the mutual information between input feature $\mathscr{P}$ and class $\mathscr{C}$ is represented as:

$$I(\mathscr{P}; \mathscr{C}) = H(\mathscr{C}) - H(\mathscr{C}|\mathscr{P}), \tag{17}$$

where $H(\mathscr{C})$ is obtained by Equation (3) with a uniform class prior.

The conditional entropy $H(\mathscr{C}|\mathscr{P})$ based on the input feature $\mathscr{P}$ (that is a $|Q|$-dimensional random vector) is obtained as:

$$H(\mathscr{C}|\mathscr{P}) = -\int_{\mathscr{P}} p_{\mathscr{P}}(\mathbf{p}) \sum_{i=0}^{|\mathscr{C}|-1} p_{\mathscr{C}|\mathscr{P}}(c_i|\mathbf{p}) \log_2 p_{\mathscr{C}|\mathscr{P}}(c_i|\mathbf{p}) d\mathbf{p}, \tag{18}$$

where $|\mathscr{C}|$ is the number of classes. By following the Bayesian rule, the probability $p_{\mathscr{C}|\mathscr{P}}(c|\mathbf{p})$ is expressed as:

$$p_{\mathscr{C}|\mathscr{P}}(c|\mathbf{p}) = \frac{p_{\mathscr{P}|\mathscr{C}}(\mathbf{p}|c) p_{\mathscr{C}}(c)}{p_{\mathscr{P}}(\mathbf{p})}. \tag{19}$$

Since $\sum_{i=0}^{|\mathscr{C}|-1} p_{\mathscr{C}|\mathscr{P}}(c_i|\mathbf{p}) = 1$ for any given $\boldsymbol{p}$, the conditional probability in Equation (19) reduces to

$$p_{\mathscr{C}|\mathscr{P}}(c|\mathbf{p}) = \frac{p_{\mathscr{C}|\mathscr{P}}(c|\mathbf{p})}{\sum_{i=0}^{|\mathscr{C}|-1} p_{\mathscr{C}|\mathscr{P}}(c_i|\mathbf{p})} = \frac{p_{\mathscr{P}|\mathscr{C}}(\mathbf{p}|c) p_{\mathscr{C}}(c)}{\sum_{i=0}^{|\mathscr{C}|-1} p_{\mathscr{P}|\mathscr{C}}(\mathbf{p}|c_i) p_{\mathscr{C}}(c_i)}. \tag{20}$$

The Parzen window estimator at each class $c \in \mathscr{C}$ is obtained as:

$$\hat{p}_{\mathscr{P}|\mathscr{C}}(\mathbf{p}|c) = \frac{1}{n_c} \sum_{i \in \mathscr{I}_c} \varphi(\mathbf{p} - \mathbf{p}^i, h_c), \tag{21}$$

where n_c is the number of training samples belonging to the class $c \in \mathscr{C}$ and $\mathscr{I}_c$ is the set of the respective indices of training samples (i.e., $|\mathscr{I}_c| = n_c$); φ is the Parzen window function; and h_c is the Parzen window width parameter for the pattern class $c \in \mathscr{C}$. Parzen [25] showed that the estimated density converges to the true density if φ and h are selected properly. By combining Equations (20) and (21), the Parzen window estimator is constructed [24] as:

$$\hat{p}_{\mathscr{C}|\mathscr{P}}(c|\mathbf{p}) = \frac{\frac{1}{n_c}\sum_{i\in\mathscr{I}_c}\varphi(\mathbf{p}-\mathbf{p}^i,\, h_c)}{\sum_{k=0}^{|\mathscr{C}|-1}\frac{1}{n_k}\sum_{i\in\mathscr{I}_k}\varphi(\mathbf{p}-\mathbf{p}^j,\, h_k)}, \tag{22}$$

where h_c and h_k are class-specific width parameters of Parzen window.

Using a d-variate Gaussian window (with covariance matrix S) the Parzen density estimator is chosen as:

$$\varphi(\mathbf{p},\, h) = \frac{1}{(2\pi)^{d/2}h^d|S|^{1/2}}\exp\left(-\frac{\mathbf{p}^T S^{-1}\mathbf{p}}{2h^2}\right), \tag{23}$$

where the parameter h controls the tradeoff between variance and bias (i.e., expected value of the error of the Parzen window estimator). An increment in h would reduce the variance at the expense of increased bias and vice-versa for a decrement in h. Following [24], the current paper uses $h_c = \frac{1}{2\log_e(n_c)}$ for each $c \in \mathscr{C}$.

Remark 3.1. The problem of choosing h is crucial for the Parzen density estimator in Equation (23). A large h will over-smooth the estimator and mask the data structure; on the other hand, a small h will yield an estimator that is spiky and will be very hard to interpret.

If the same parameter h of Parzen window and the same covariance matrix S are used for each class and if the number of training samples in each class is the same, i.e., n_c is uniform for all classes, then the estimate of the conditional probability becomes:

$$\hat{p}_{\mathscr{C}|\mathscr{P}}(c|\mathbf{p}) = \frac{\sum_{i\in\mathscr{I}_c} \exp\left(-\frac{(\mathbf{p}-\mathbf{p}^i)^T S^{-1}(\mathbf{p}-\mathbf{p}^i)}{2h^2}\right)}{\sum_{k=0}^{|\mathscr{C}|-1}\sum_{j\in\mathscr{I}_k} \exp\left(-\frac{(\mathbf{p}-\mathbf{p}^j)^T S^{-1}(\mathbf{p}-\mathbf{p}^j)}{2h^2}\right)}. \tag{24}$$

If the integration in Equation (18) is replaced by summation of the sample points with equal sample probability, then the conditional entropy (based on an input feature $\mathscr{P}$) derived from the training data belonging to all classes in $\mathscr{C}$ becomes:

$$\hat{H}(\mathscr{C}|\mathscr{P}) = -\frac{1}{n}\sum_{j=1}^{n}\sum_{i=0}^{|\mathscr{C}|-1} \hat{p}_{\mathscr{C}|\mathscr{P}}(c_i|\mathbf{p}^j)\log_2 \hat{p}_{\mathscr{C}|\mathscr{P}}(c_i|\mathbf{p}^j), \tag{25}$$

where $n \triangleq \sum_{i=0}^{|\mathscr{C}|-1} n_{c_i}$ is the total number of training samples under consideration and $\mathbf{p}^j$ is the feature vector computed from the j-th training data in the ensemble of all classes. Finally, the estimated mutual information is obtained from Equations (24) and (25).

For $|\mathscr{C}| \ll n$, the computational complexity of Parzen window estimation [24] in Equation (25) is of the order $n^2 \times d$; this implies that, unlike the histogram-based methods, Parzen window estimation does not require excessive memory. With this estimation, the problem of alphabet size selection can be solved by the sequential optimization method described in the previous subsection. It is noted that the dimension of the input feature vector (e.g., state probability vector **p**) starts from two at the beginning and increases by one as a new partition segment is added.

D. Pattern classification based on feature vectors generated by STSA

The suboptimal partitioning obtained by the procedure in Subsection 3-A is used to construct a PFSA from each training and testing time series (see Subsection 2-A3 and Algorithm 4 in the Appendix); the state probability vector of the PFSA is the extracted feature for each time series. The objective here is to classify the test data in the constructed low-dimensional feature space. In this respect, there are many options for selecting classifiers that could either be parametric or non-parametric. In this paper, two commonly used pattern classifiers, namely, k-NN and support vector machine (SVM) have been adopted [26].

4. Algorithm Validation and Results

For validation of the proposed time series partitioning and alphabet size selection procedure, this section presents two examples, namely, (i) Parameter identification in nonlinear Duffing systems [27], and (ii) Lean blowout (LBO) prediction based on experimental data generated from a laboratory-scale combustor [28], [29], along with explanations of the results of alphabet size selection.

A. Example #1: Duffing system simulation

The exogenously excited Duffing system is nonlinear and exhibits complex behaviour with chaotic and bifurcation properties; its governing equation is:

$$\frac{d^2y}{dt^2} + \beta \frac{dy}{dt} + \alpha y(t) + y^3(t) = A\cos(\omega t), \tag{26}$$

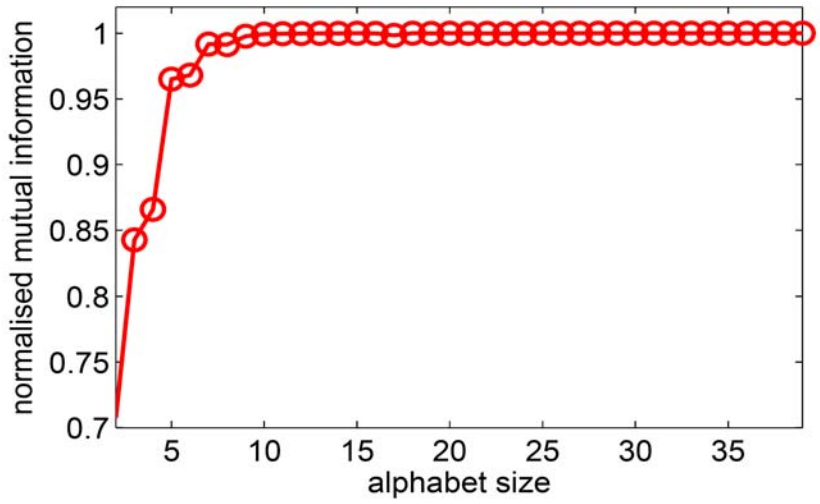
where the amplitude $A = 22.0$, excitation frequency $\omega = 5.0$, and the initial conditions are: $y(0) = 1.0$ and $\frac{dy}{dt}(0) = 0.0$; however, these initial conditions have no significance because only the steady-state oscillatory responses have been analyzed. Reference values of the remaining

parameters, to be identified, are: $\alpha = 1.0$ and $\beta = 0.1$. It is well-known that the Duffing system [27] goes through a bifurcation at different combinations of α and β, which may not be easily identified from time series data. The objective of this example problem is to accurately identify the ranges of the parameters α and β when the Duffing system has not crossed any bifurcation boundaries.

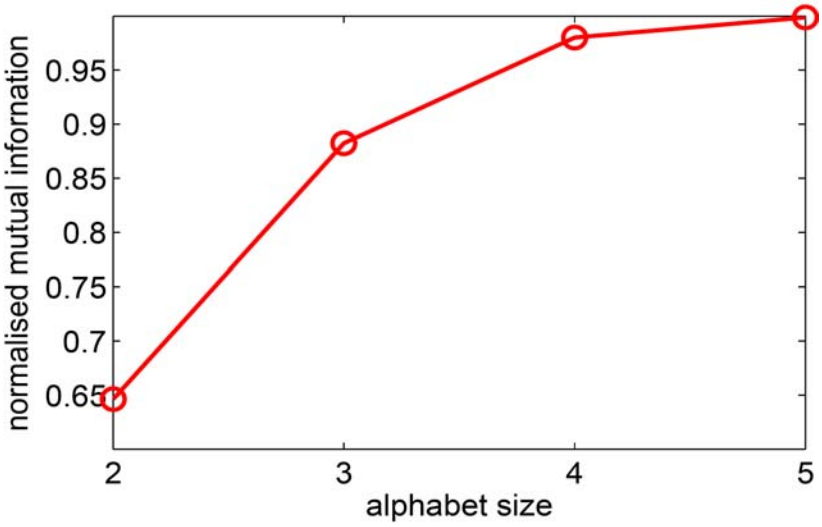
(1) Parameter identification for multi-class Duffing systems

At first, only two classes of Duffing system are defined based on the range of β, that are: (i) Class 1 $(0.100 \leq \beta \leq 0.147)$ and (ii) Class 2 $(0.147 \leq \beta \leq 0.194)$. Two hundred simulation runs of the Duffing system have been conducted for each class to generate data set for analysis among which 30 samples are chosen for determining the optimal partitioning, and three-fold cross validation has been performed on the remaining data set to determine the classification results. Parameters α and β are chosen randomly from independent uniform distributions within the prescribed ranges. Time series data of the output variable y have been collected for each sample point in the parameter space. The length of the simulation time window is 80 seconds sampled at 100Hz, which generates 8,000 data points. The range of time-series is divided into 40 grid cells via uniform partitioning.

The sequential partitioning optimization procedure described in the previous section is then employed to identify the optimal partitioning and alphabet size. Figure 3(a) and Figure 3(b) depict the nature of mutual information between the state probability vector and the class labels for the depth of the D-Markov machine of the input feature being $D = 1$ and $D = 2$, respectively. For $D = 2$, normalized mutual information converges to 1 much earlier for alphabet size $|\sum| = 5$ than that for $D = 1$ and $|\sum| = 9$. In each case, stopping criterion follows Equation (14)) with the parameter $\eta_{stop} = 0.01$.



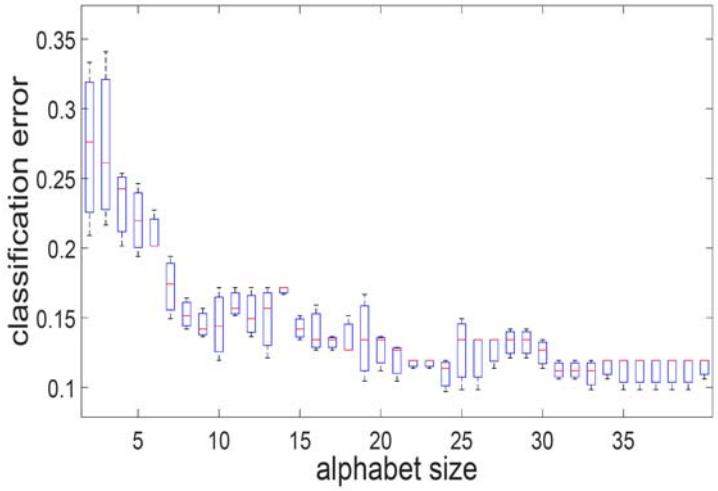
(a)



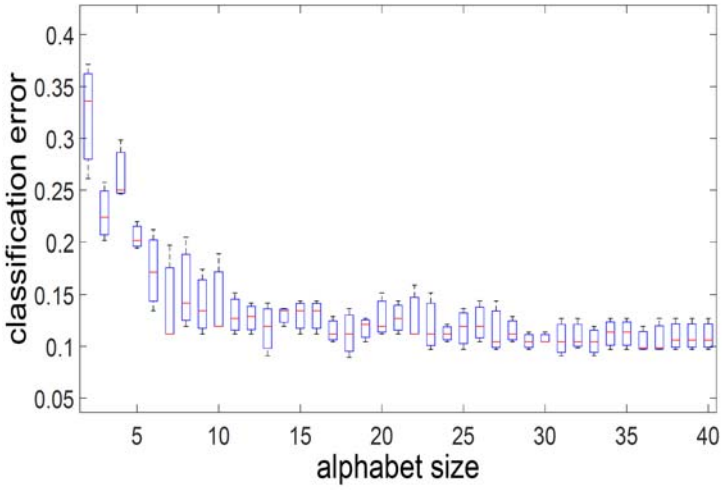
(b)

Figure 3. Mutual information as a function of alphabet size $|\sum|$ for two-class Duffing system with (a) $D = 1$, (b) $D = 2$.

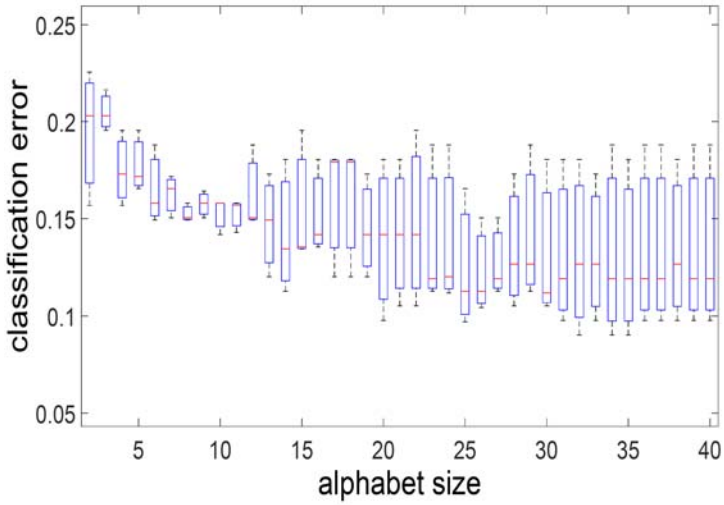
Figures 4(a), 4(b) and 4(c) show the classification performance of the k-NN classifier [26] with $k = 5$ for three different levels of robustness, i.e., different variances that are fractions, 0.67, 0.5, 0.25, of the inter partition width, respectively (see Subsection 3-B). It is observed that the classification errors are smaller with smaller variance which is a consequence of smaller robustness variance fractions.



(a)



(b)



(c)

Figure 4. Misclassification of partitioning for two-class Duffing system with $D = 1$ and variance fraction: (a) 0.67, (b) 0.5, (c) 0.25.

Figure 5 shows the nature of mutual information between the state probability vector and the class labels for the Duffing system with 4 classes, corresponding to four different combinations of the ranges: $((0.100 \le \beta \le 0.147), (0.147 \le \beta \le 0.194), (0.934 \le \alpha \le 1.067), (0.8 \le \alpha \le 0.934))$, within which the parameters α and β in Equation (26) are located. The convergence rate of the normalized mutual information is smaller in this case than that for the binary classification scheme because a larger alphabet is required to capture the information of four classes. The stopping criterion follows Equation (14)) with the parameter $\eta_{stop} = 0.01$.

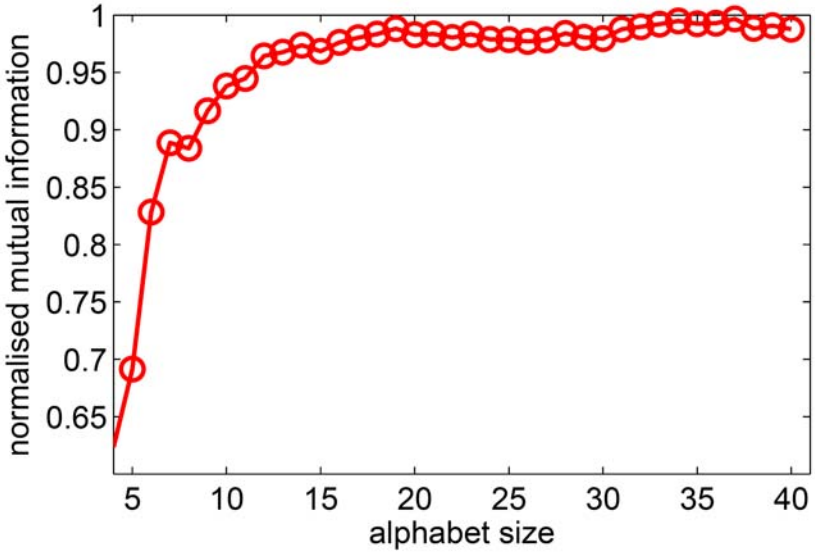


Figure 5. Mutual information as a function of alphabet size $|\sum|$ for four-class Duffing system with $D = 1$ and $\eta_{stop} = 0.01$.

B. Example #2: Lean blowout prediction in a laboratory-scale combustor

Ultra-lean combustion is commonly used for NOx reduction and is susceptible to thermo-acoustic instabilities and lean blowout (LBO) [29]. It is well known that occurrence of LBO could be detrimental for operations of both land-based and aircraft gas turbine engines. For example, LBO in land-based gas turbines may lead to engine shutdown and subsequent relighting involves loss of productivity; similarly, LBO in aircraft gas turbines may cause loss of engine thrust especially if the fuel flow is suddenly reduced during a throttling operation, or if air flow reduction takes place at a much slower rate due to the moment of inertia of the compressor. In essence, a sudden decrease in the equivalence ratio may lead to LBO in gas turbine engines, which could have serious consequences, and calls for early detection & accurate prediction of LBO and appropriate actions for its mitigation.

The proposed procedure of time series partitioning and alphabet size selection has been evaluated under multiple operating conditions (e.g., airflow rates and premixing levels of fuel and air). Figure 6 shows a schematic representation of the combustor and detailed description is available in [28]. A series of experiments have been conducted on this laboratory apparatus with liquefied petroleum gas (LPG) fuel at airflow rates of 150, 175, and 200 liters per minute (lpm) for two different fuel-air 399 premixing lengths (i.e., distance of fuel injection port from the dump plane) of L_{fuel} = 25cm and 15cm for port 3 and port 5, respectively, as seen in Figure 6. For each experiment protocol, chemiluminescence time series data were collected while reducing the fuel-air ratio ϕ in steps till the combustor system reached LBO. The main challenge here is to predict quantitatively how far a combustion process is from the onset of LBO in real time. It is easier to predict LBO under high premixing (i.e., port 3) as the precursor events are more dominant [28] than that under lower premixing (i.e., port 5).

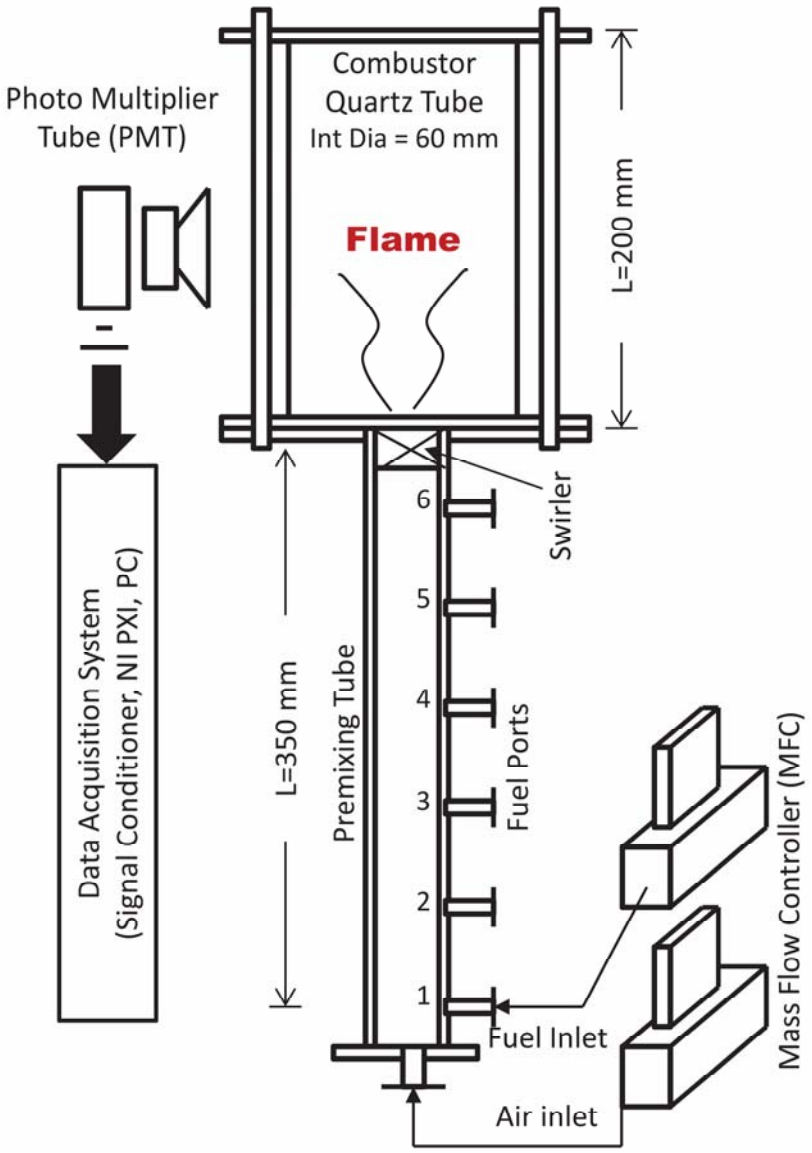


Figure 6. Schematic view of the laboratory-scale swirl-stabilized dump combustor [29].

A nested classification architecture [29] is proposed in accordance with the range of the non-dimensional equivalence ratio of ϕ / ϕ_{LBO} for early detection of LBO. In the training phase, the chemiluminescence time series of duration 3 sec (at the sampling rate of 2kHz) for each premixing length are grouped into two classes as: *Alarm* $(1 \leq \phi / \phi_{LBO} \leq 1.20)$ and *Nominal* $(\phi / \phi_{LBO} > 1.20)$. The class *Alarm* is subdivided into two finer classes as: *Impending LBO* (ILBO) for $1 \leq \phi / \phi_{LBO} \leq 1.1$, and *Progressive LBO* (PLBO) for $1.1 < \phi / \phi_{LBO} \leq 1.2$. Identification of the PLBO phase is critical for avoidance of LBO as control actions need to be initiated typically near the PLBO-ILBO boundary. Figure 7 shows typical time series profiles of chemiluminescence data for Nominal, PLBO and ILBO conditions, respectively, at port 3 level of premixing.

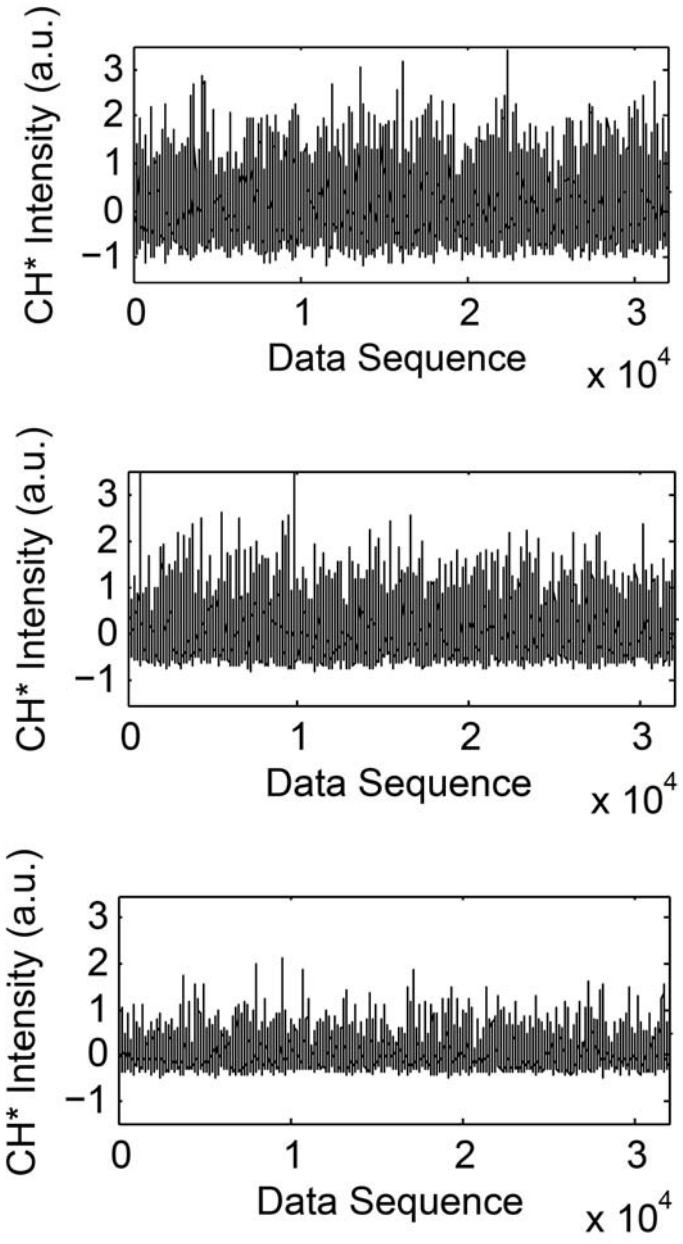
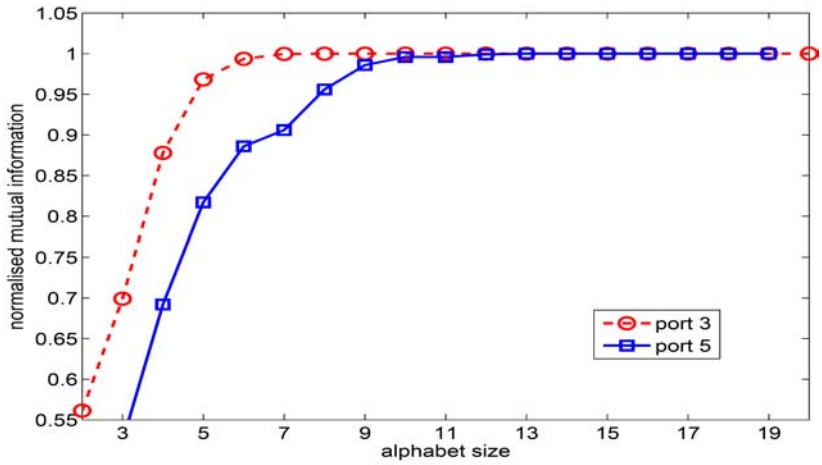
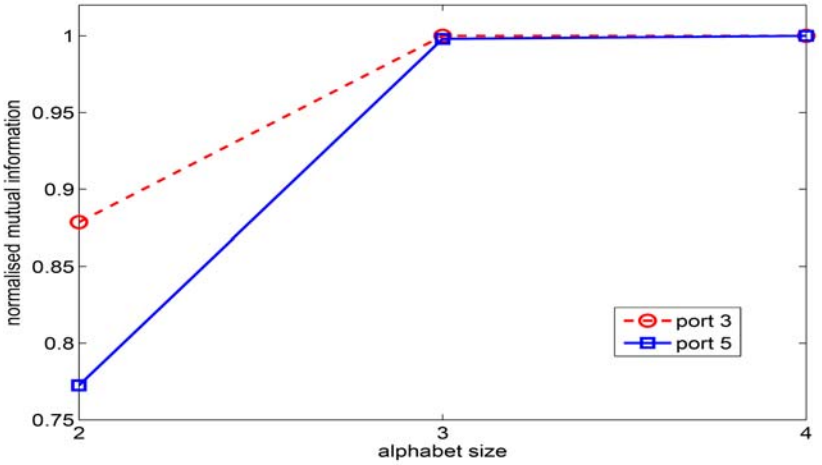


Figure 7. Chemiluminescence time-series for three classes (from left to right): nominal, progressive LBO and impending LBO at port 3 level of premixing.

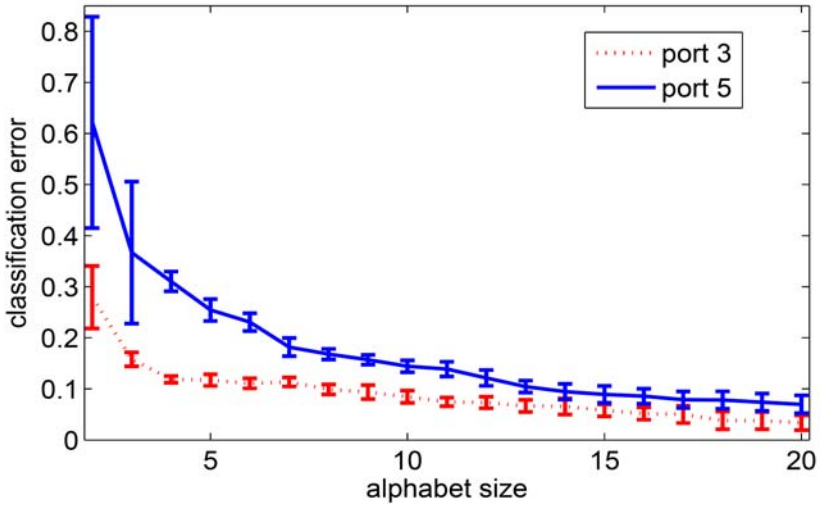
Sequential partitioning optimization is applied on the chemiluminescence data to identify the partitioning and alphabet size, which start with 20 grid cells; robustness with σ fraction of 0.25 (see Subsection 3-B) is chosen in this analysis. Figure 8(a) shows the variations of mutual information between the D-Markov feature vectors with $D = 1$ and the class labels for both premixing levels; Figure 8(b) presents a similar analysis for $D = 2$. It is observed from these results that the normalized mutual information converges to 1 with a much smaller alphabet size $|\sum|$ for $D = 2$ than that for $D = 1$, reflecting the fact that D-Markov features with larger memory should be able to capture the same class information with a smaller $|\sum|$; however, the number of PFSA states for $D = 2$ could be larger than that for $D = 1$. Normalized mutual information for high premixing (port 3) converges to 1 for a smaller $|\sum|$ than that for low premixing (port 5). This phenomenon is more apparent for $D = 1$, where the alphabet size for port 3 and port 5 are chosen as $|\sum| = 7$ and $|\sum| = 12$, respectively, according to a stopping rule of $I_{\max} = 1$ (see Equation (13)). The rationale for this observation is attributed to large class separability for high premixing due to the presence of dominant precursor events leading to LBO. Support vector machines (SVM) with radial basis functions [26] have been used based on 70% training at each layer of the nested classification. Variance of the radial basis function is optimized for each layer of the nested classification via a grid search method and it is found to be 1 in most of the cases. Figures 8(c) and 8(d) present the variations of classification error while the proposed partitioning scheme sequentially increases $|\sum|$ for both $D = 1$ and $D = 2$. The error bars correspond to standard deviations of the classification error over 10-fold cross validation. The classification error is smaller for high premixing (port 3) than that for low premixing (port 5). It is also observed that relatively smaller classification error occurs at $D = 2$ than that at $D = 1$, especially for small $|\sum|$.


normalised mutual information
alphabet size
port 3
port 5
1.05
1
0.95
0.9
0.85
0.8
0.75
0.7
0.65
0.6
0.55
3
5
7
9
11
13
15
17
19

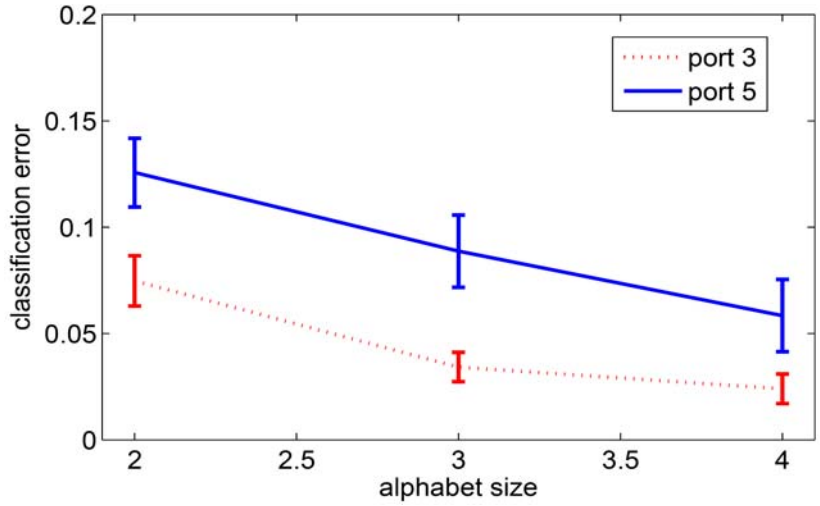
(a)


normalised mutual information
alphabet size
port 3
port 5
1
0.95
0.9
0.85
0.8
0.75
2
3
4

(b)



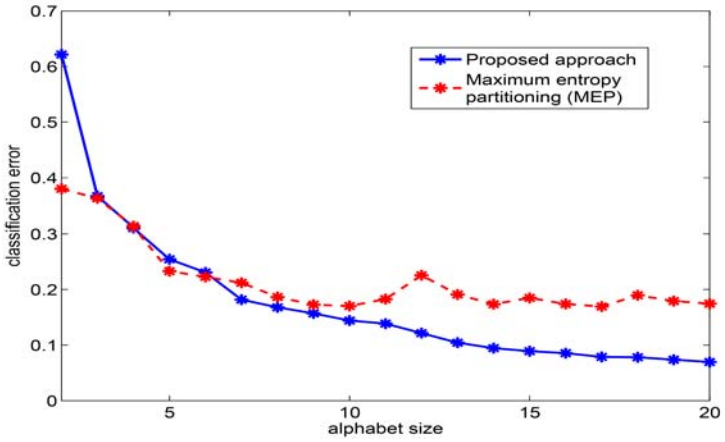
(c)



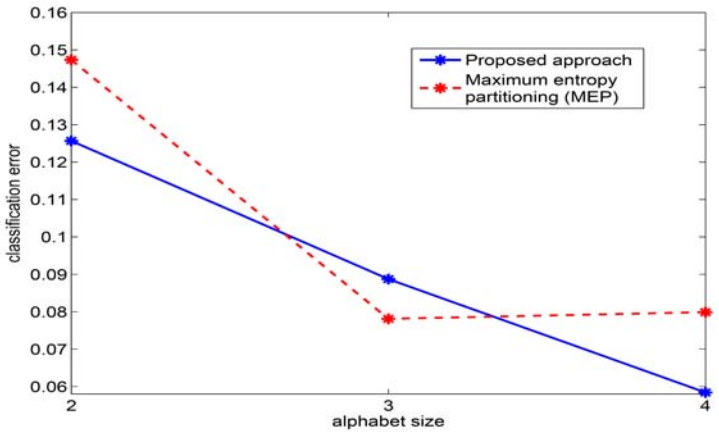
(d)

Figure 8. Top: Mutual information as a function of alphabet size $|\sum|$ for port 3 and port 5 levels of premixing with (a) $D = 1$ and (b) $D = 2$. **Bottom:** Variation of classification error as a function of alphabet size $|\sum|$ for port 3 and port 5 levels of premixing level with (c) $D = 1$ and (d) $D = 2$.

Performance comparison. Performance of the proposed partitioning is compared with that of a benchmark partitioning, namely, maximum entropy partitioning (MEP). Figure 9 shows the profiles of classification error in the proposed approach and in MEP as a function of $|\sum|$ for the port 5 scenario. By applying normalized mutual information stopping criterion as mentioned earlier, the classification error is seen to be smaller for the proposed scheme at $D = 1$ and $|\sum| \geq 6$ in Figure 9(a) and at $D = 2$ and $|\sum| = 4$ in Figure 9(b).



(a)



(b)

Figure 9. Comparison of classification error with varying alphabet size for (port 5) premixing: (a) $D = 1$ and (b) $D = 2$.

5. Summary and Conclusions

This paper addresses the issues of: (i) alphabet size selection and partitioning of time series data for symbolization of time series, and (ii) information-theoretic analysis in dynamic data-driven application systems (DDDAS) [11] with a focus on feature extraction and pattern

classification from sensor data. The feature extraction algorithm maximizes the mutual information between the input features and pattern classes in the framework of symbolic time series analysis (STSA) [1]. The proposed technique is validated on two examples:

(1) Simulation data for an exogenously excited multi-class Duffing system [27]; and

(2) Experimental data of chemiluminescence time series generated from a laboratory-scale swirl-stabilized combustor [28] for lean blowout (LBO) prediction.

The optimization procedure has also been used to evaluate the capability of other partitioning schemes towards achieving a specified objective. Apparently, the proposed partitioning technique yields satisfactory performance of pattern classification in several test phases. Nevertheless, its efficacy for optimization may depend on the very nature of the time series under consideration.

Incorporation of an explicit term for class separability in the proposed objective function is a topic of future investigation. Apart from this issue, the following research topics are recommended for future research:

- Implementation of simultaneous optimization techniques instead of sequential ones.

- Tradeoff between the performance gain and the loss of computational speed.

- Validation of the proposed algorithm for other applications of pattern classification.

Acknowledgements

The authors are grateful to Professor Achintya Mukhopadhyay of Jadavpur University, Kolkata, India for providing the experimental data for testing the algorithms reported in this paper. The authors also acknowledge the benefits of discussion with Dr. Shashi Phoha of Penn State Applied Research Laboratory.

References

[1] C. Daw, C. Fenney and E. Tracy, A review of symbolic analysis of experimental data, Review of Scientific Instruments 74 (2003), 915-930.

[2] J. Lin, E. Keogh, L. Wei and S. lonardi, Experiencing sax: A novel symbolic representation of time series, Data Mining and Knowledge Discovery 15(2) (2007).

doi: 10.1007/s10618–007–0064–z

[3] D. Lind and B. Marcus, An Introduction to Symbolic Dynamics and Coding, Cambridge University Press, Cambridge, U. K., 1995.

[4] A. Ray, Symbolic dynamic analysis of complex systems for anomaly detection, Signal Processing 84(7) (2004), 1115-1130.

[5] V. Rajagopalan and A. Ray, Symbolic time series analysis via wavelet-based partitioning, Signal Processing 86(11) (2006), 3309-3320.

[6] A. Subbu and A. Ray, Space partitioning via Hilbert transform for symbolic time series analysis, Applied Physics Letters 92(8) (2008), 084107–1-084107–3.

[7] P. Adenis, Y. Wen and A. Ray, An inner product space on irreducible and synchronizable probabilistic finite state automata, Mathematics of Control, Signals, and Systems 23(1) (2012), 281-310.

[8] Y. Wen, A. Ray and S. Phoha, Hilbert space formulation of symbolic systems for signal representation and analysis, Signal Processing 93 (2013), 2594-2611.

[9] Y. Wen, K. Mukherjee and A. Ray, Adaptive pattern classification for symbolic dynamic systems, Signal Processing 93 (2013), 252-260.

[10] K. Mukherjee and A. Ray, State splitting and merging in probabilistic finite state automata for signal representation and analysis, Signal Processing 104 (2014), 105-119.

[11] F. Darema, Dynamic data driven applications systems: New capabilities for application simulations and measurements, in 5-th International Conference on Computational Science - ICCS 2005, (Atlanta, GA; United States), 2005.

[12] S. Gupta, A. Ray and E. Keller, Symbolic time series analysis of ultrasonic data for early detection of fatigue damage, Mechanical Systems and Signal Processing, 21(2) (2007), 866-884.

[13] C. Rao, A. Ray, S. Sarkar and M. Yasar, Review and comparative evaluation of symbolic dynamic filtering for detection of anomaly patterns, Signal, Image and Video Processing 3(2) (2009), 101-114.

[14] S. Bahrampour, A. Ray, S. Sarkar, T. Damarla and N. Nasrabadi, Performance comparison of feature extraction algorithms for target detection and classification, Pattern Recognition Letters 34 (2013), 2126-2134.

[15] P. Dupont, F. Denis and Y. Esposito, Links between probabilistic automata and hidden Markov models: probability distributions, learning models and induction algorithms, Pattern Recognition 38(9) (2005), 1349-1371.

[16] E. Vidal, F. Thollard, C. de la Higuera, F. Casacuberta and R. Carrasco, Probabilistic finite-state machines - Part I and Part II, IEEE Transactions on Pattern Analysis and Machine Intelligence 27 (2005), 1013-1039.

[17] M. Buhl and M. Kennel, Statistically relaxing to generating partitions for observed time-series data, Physical Review E 71(4) (2005), 046213.

[18] S. Sarkar, P. Chattopadhyay and A. Ray, Symbolization of dynamic data-driven systems for signal representation, Signal, Image, and Video Processing (SIVP) (2016), 1535-1542.

[19] T. Cover and J. Thomas, Elements of Information Theory, 2nd Edition, Wiley, 2006.

[20] R. Steuer, L. Molgedey, W. Ebeling and M. Jimenez-Montano, Entropy and optimal partition for data analysis, The European Physical Journal B 19 (2001), 265-269.

[21] X. Jin, S. Gupta, K. Mukherjee and A. Ray, Wavelet-based feature extraction using probabilistic finite state automata for pattern classification, Pattern Recognition 44(7) (2011), 1343-1356.

[22] S. Sarkar, K. Mukherjee, X. Jin, D. Singh and A. Ray, Optimization of symbolic feature extraction for pattern classification, Signal Processing 92(3) (2012), 625-635.

[23] A. Berman and R. Plemmons, Nonnegative Matrices in the Mathematical Sciences, SIAM Publications, Philadelphia, PA, USA, 1994.

[24] N. Kwak and C. Choi, Input feature selection by mutual information based on Parzen window, IEEE Transactions on Pattern Analysis and Machine Learning 24(12) (2002), 1667-1671.

[25] E. Parzen, On estimation of a probability density function and mode, Annals of Math. Statistics 33 (1962), 1065-1076.

[26] C. M. Bishop, Pattern Recognition and Machine Learning, Springer-Verlag, New York, Inc., 2006.

[27] J. Thompson and H. Stewart, Nonlinear Dynamics and Chaos, Wiley, Chichester, UK, 1986.

[28] S. Sarkar, A. Ray, A. Mukhopadhyay, R. Chaudhuri and S. Sen, Early detection of lean blow out (lbo) via generalized *d*-markov machine construction, in 2014 American Control Conference (ACC), (Portland, OR, USA), June 4-6, 2014.

[29] S. Sarkar, A. Ray, A. Mukhopadhyay and S. Sen, Dynamic data-driven prediction of lean blowout in a swirl-stabilized combustor, International Journal of Spray and Combustion Dynamics (2015), 209-242.

■

Appendix

This appendix lists four algorithms used in the main body of the paper, based on which the codes were written in Matlab.

Algorithm 1: Maximum entropy partitioning of time series

Require: Time-series data $\boldsymbol{x}$, number of symbols $\left|\sum\right|$

Ensure: Partitioning $\wp$

1: Sort $\boldsymbol{x}$ in ascending order

2: Let ℓ = length($\boldsymbol{x}$)

3: $\wp(1) = \boldsymbol{x}(1)$, i.e., the minimum element of $\boldsymbol{x}$

4: **for** $i = 2$ to $\left|\sum\right|$ **do**

5: $\wp(i) = \boldsymbol{x}\left(ceil\left(\frac{(i-1)*\ell}{\left|\sum\right|}\right)\right)$

6: **end for**

7: $\wp(\left|\sum\right|+1) = \boldsymbol{x}(\ell)$; maximum of $\boldsymbol{x}$

Algorithm 2: Uniform partitioning of time series

Require: Time-series data $\boldsymbol{x}$, number of symbols $\left|\sum\right|$

Ensure: Partitioning $\wp$

1: Let $x_{\min} = \min(x)$ and $x_{\max} = \max(x)$.

2: Range of $\boldsymbol{x}$ $R = x_{\max} - x_{\min}$.

3: $\wp(1) = x_{\min}$.

4: **for** $i = 2$ to $\left|\sum\right|$ **do**

5: $\wp(i) = \wp(1) + (i-1)*\frac{R}{\left|\sum\right|}$

6: **end for**

7: $\wp(\left|\sum\right|+1) = x_{\max}$

Algorithm 3: Selection of alphabet size and partitioning of time series

Require: Training set: Time series $\{v_i \in \mathbb{R}^m,\ i = 1, 2, \cdots, n\}$, where m is the approximate length of each time series $(m \gg |\sum|)$; Class labels $\{c_i,\ i = 1, 2, \cdots, k\}$ for each time series; n is the total number of time series data available for training; and there are k classes; and a positive integer $L < m$ leading to the cardinality $(L-1)$ of the set of possible boundary locations, from which the final selection of a (possibly) reduced set of partitioning locations is made; and the parameter η_{stop}.

Ensure: The suboptimal alphabet size $|\sum|$ and the set of partition locations for alphabet size $\sum$: $\Lambda^*_{|\Sigma|}$

Divide the data range into L regions marked by L–1 boundaries Υ by MEP or UP

$\Lambda^*_1 = \emptyset$

for $i = 2$ to L-1 **do**

Set of partition locations not yet chosen $P = \Upsilon \backslash \Lambda^*_{i-1}$

for $j = 1$ to $|\Upsilon \backslash \Lambda^*_{i-1}|$ **do**

Partition set $\Lambda^j_i = \Lambda^*_{i-1} \cup P(j)$

Compute the feature vectors $\{\ \mathbf{p}_i,\ i = 1, 2, \cdots, n$ using the method described in section 2$\}$

Compute $I(\mathscr{P}(\Lambda^j_i); \mathscr{C})$ using Equation (25)

end for

$\lambda^*_{i-1} = \arg\max_{P(j)} I(\mathscr{P}(\Lambda^j_i); \mathscr{C})$

$\Lambda^*_i = \Lambda^*_{i-1} \cup \lambda^*_{i-1}$

Compute the feature vectors $\{\ \mathbf{p}_i,\ i = 1, 2, \cdots, n$ using the method described in Section 2$\}$

Compute $I(\mathscr{P}(\Lambda^*_i); \mathscr{C})$ using Equation (25)

if $I(\mathscr{P}(\Lambda^*_i); \mathscr{C}) - I(\mathscr{P}(\Lambda^*_{i-1}); \mathscr{C}) \le \eta_{stop}$ **then**

Optimal alphabet size = i; Partition $set = \Lambda^*_i$

Stop

end if

end for

Algorithm 4: Construction of PFSA) for feature extraction

Require: Time series data $\boldsymbol{x}$; partitioning P obtained from Algorithm 3

Ensure: State probability vector $\boldsymbol{p}$

1: Partition $\boldsymbol{x}$ using P to obtain the symbol string $\boldsymbol{s}$

2: Construct PFSA K using $\boldsymbol{s}$ as described in Section 2