# Transfer Learning for Detection of Combustion Instability Via Symbolic Time-Series Analysis

**Chandrachur Bhattacharya**
Departments of Mechanical and Electrical Engineering,
Pennsylvania State University,
University Park, PA 16802
e-mail: chandrachur.bhattacharya@gmail.com

**Asok Ray**
Fellow ASME
Departments of Mechanical Engineering and Mathematics,
Pennsylvania State University,
University Park, PA 16802
e-mail: axr2@psu.edu

*Transfer learning (TL) is a machine learning (ML) tool where the knowledge, acquired from a source domain, is "transferred" to perform a task in a target domain that has (to some extent) a similar setting. The underlying concept does not require the ML method to analyze a new problem from the beginning, and thereby both the learning time and the amount of required target-domain data are reduced for training. An example is the occurrence of thermoacoustic instability (TAI) in combustors, which may cause pressure oscillations, possibly leading to flame extinction as well as undesirable vibrations in the mechanical structures. In this situation, it is difficult to collect useful data from industrial combustion systems, due to the transient nature of TAI phenomena. A feasible solution is the usage of prototypes or emulators, like a Rijke tube, to produce largely similar phenomena. This paper proposes symbolic time-series analysis (STSA)-based TL, where the key idea is to develop a capability of discrimination between stable and unstable operations of a combustor, based on the time-series of pressure oscillations from a data source that contains sufficient information, even if it is not the target regime, and then transfer the learnt models to the target regime. The proposed STSA-based pattern classifier is trained on a previously validated numerical model of a Rijke-tube apparatus. The knowledge of this trained classifier is transferred to classify similar operational regimes in: (i) an experimental Rijke-tube apparatus and (ii) an experimental combustion system apparatus. Results of the proposed TL have been validated by comparison with those of two shallow neural networks (NNs)-based TL and another NN having an additional long short-term memory (LSTM) layer, which serve as benchmarks, in terms of classification accuracy and computational complexity.* [DOI: 10.1115/1.4050847]

*Keywords: transfer learning, symbolic time-series analysis, neural networks, combustion systems*

## 1 Introduction

Transfer learning (TL) is a machine learning (ML) tool that is useful for data-driven assessment of operational conditions in a target domain that does not have sufficient amount of labeled data to train the underlying algorithms, while there exists another (to some extent similar) source domain with adequate labeled data that closely mimic similar operational scenarios in the target domain [1,2]. Thus, by taking advantage of TL, various machine learning algorithms (e.g., classifiers) can be trained on the available labeled data from a source domain and be deployed for analyzing a different target domain.

Transfer learning has gained momentum in the last two decades and has found its applications in diverse fields that include medical imaging [3], drug discovery [4], natural language processing [5], general game playing [6], estimation of remaining useful life in mechanical structures [7], and many more [1,2]. Although numerous applications of TL follow neural network (NN)-based architectures, other concepts such as $k$-nearest neighbors and Bayesian networks [8] have also been used for TL. Transfer learning methods are broadly categorized as:

(1) *Inductive transfer learning*: If the target domain is the same as the source domain but the tasks are different, albeit being related.

(2) *Transductive transfer learning (TTL)*: If the tasks are largely similar but the target domain is different from the source domain.

The work, reported in this paper, deals with detection of thermoacoustic instability (TAI) in combustion systems, which belongs to the second category (i.e., TTL) of transfer learning. In combustion systems, TAI is an undesirable phenomenon, because of the resulting pressure oscillations leading to flame extinction as well as vibrations of mechanical structures [9]. The dynamics of the underlying combustion process are inherently nonlinear [10] due to the constructive interference between the heat release rate (due to fuel burning) and the natural acoustics of the chamber [11], which can occur at either fuel-rich or fuel-lean conditions, depending on the construction and operation of the combustor.

From the perspectives of mechanical design and control, it is important to understand whether the combustion system is in normal operation, or is undergoing TAI, or is transitioning into TAI from normal operation. The data sources for TAI analysis are available mostly from laboratory-scale experimental combustor apparatuses, physical emulators (e.g., Rijke tubes [12])), and numerical simulation models.

Much research efforts have been expended for classifying an identified regime of operation in combustion processes as stable or unstable, with data-driven methods being a good choice for classification. To this end, various ML methods have been used to identify the operational regimes of a combustor. For example, Sarkar et al. [13] used neural networks for regime identification from dynamic flame image data. Bhattacharya et al. [14] used fast Fourier transform (FFT)-based methods on pressure time-series data; and Mondal et al. [15] used traditional ML methods such as

Contributed by the Dynamic Systems Division of ASME for publication in the Journal of Dynamic Systems, Measurement, and Control. Manuscript received November 27, 2020; final manuscript received April 5, 2021; published online May 13, 2021. Assoc. Editor: Carrie Hall.

hidden Markov models (HMM). Other methods (e.g., complex networks [16,17], recurrence analysis [18,19], and adjoint methods [20,21]) have also been used, although they do not form part of a "truly" data-driven approach.

Although neural networks are fast during testing, they usually take a long time and a very large amount of data to train. Similarly, HMM is also relatively slow to train and even to test. A much faster alternative is probabilistic finite state automata (PFSA) [22–24], which belongs to the class of symbolic time-series analysis (STSA) [25]. The PFSA-based ML tools are fast to train and test, and they require comparatively less training data [26] than neural networks, while still providing good accuracy. In another work, Bhattacharya et al. [27] have demonstrated that, for TAI detection, PFSA-based methods perform with almost similar accuracy as HMM, while PFSA is an order of magnitude (OOM) faster in testing and 2 orders of magnitude faster in training as compared to HMM.

Keeping in mind the good accuracy of PFSA-based detection for the problem at hand (i.e., TAI detection), and the ease of the algorithm implementation, the authors propose a PFSA/STSA-based method of TL for TAI detection in different regimes. This method would be very useful because, by transferring the knowledge between systems, the PFSA could be trained on a system with abundant data and subsequently use the acquired knowledge without the need for training on a large amount of additional data.

The paper has also made a significant modification of the standard PFSA [23,24] (called s-PFSA), as the projection-based PFSA (called p-PFSA). This modification is based on the following two facts that: (i) the left eigenvector, with respect to the unique unity eigenvalue of the state probability transition matrix of an ergodic PFSA, is the steady-state solution of the state probability vector [28], and (ii) this left eigenvector is orthogonal to each of the right eigenvectors corresponding to the remaining eigenvalues.

Summarized below are the primary contributions of the paper:

(1) *Extensions of the s-PFSA*: The algorithms of s-PFSA are modified, based on the algebraic and geometric properties of state-transition probability matrices, using a projection method, and hence is called p-PFSA. This method shows superior accuracy and robustness of classification.
(2) *Development of STSA/PFSA-based transfer learning*: The objective here is to learn from a system dataset and to apply the learnt models on different problems.
(3) *Experimental validation*: Efficacy of the proposed TL method is demonstrated on the problem of identifying stable and unstable operations in combustion systems. The models are trained on a numerical Rijke-tube model [29], and the learnt knowledge is transferred to classification of similar regimes of operation in: (i) a Rijke-tube apparatus [15] and (ii) a combustor apparatus [30].
(4) *Comparison of the PFSA-based TL with NN-based TL*: The proposed PFSA-based TL has been validated by comparison with NN-based TL using three separate NN architectures: two shallow fully connected nets and another net that has an long short-term memory (LSTM) layer in addition to two hidden layers, all of which serve as benchmarks, in terms of classification accuracy, computation time, and amount of data needed to train good classifiers.

The paper is organized as follows: Section 2 describes the three systems that generate ensembles of data to demonstrate the efficacy of the proposed TL method. Section 3 provides a brief background on STSA and PFSA-based classification. Section 4 succinctly describes both PFSA-based TL and NN-based TL. Section 5 presents the results and detailed discussions for validation of the proposed PFSA-based TL with NN-based TL as benchmarks. Section 6 summarizes and concludes the paper with recommendations for future research.

## 2 Data Generation for Transfer Learning

This section deals with generation of data for training the transfer learning algorithms from the following three different sources of time-series data from: (i) a previously validated numerical model [29] of a Rijke tube [12], (ii) an experimental apparatus of electrically heated Rijke tube [15], and (iii) an experimental apparatus of fuel-burning combustor [30]. The knowledge, derived from the numerical model, is first transferred for identification of the identical states in the experimental Rijke-tube apparatus [15], which had been originally used to validate the numerical model. Then, the unstable regime in the experimental combustor apparatus [30] is identified by transferring the aforementioned knowledge derived from the numerical model [29]. Subsections that follow describe each of the three systems used to generate the pressure time-series to perform the classification. A description of each system is available in the respective reference.

**2.1 Source Domain: Numerical Rijke-Tube Model.** This subsection describes the source domain as a Galerkin-decomposition-based reduced-order numerical model of a Rijke tube [6,31], where the heat source interacts with the acoustics of the Rijke tube to produce TAI [11]. The one-dimensional wave equation is derived for the pressure perturbations ($p'$) as

$$\frac{\partial^2 p'}{\partial t^2} - a^2 \frac{\partial^2 p'}{\partial x^2} = (\gamma - 1)\frac{\partial \dot{Q}}{\partial t} \tag{1}$$

where $a$ is the speed of sound, and $\dot{Q}$ is the volumetric rate of thermal power addition. The effects of mean flow on the acoustic field are not included in Eq. (1). Culick's expansion [31] has been used for the pressure perturbations ($p'$) and velocity perturbations ($u'$) using the Galerkin eigen-acoustic modes. The decomposition into $n$ modes having individual time-varying modal amplitudes of $\eta_j(t)$ yields

$$p'(x,t) = \sum_{j=1}^{n} p'_j(x,t) = p_0 \sum_{j=1}^{n} \eta_j(t)\psi_j(x) \tag{2}$$

$$u'(x,t) = \sum_{j=1}^{n} u'_j(x,t) = \sum_{j=1}^{n} \frac{\dot{\eta}_j(t)}{\gamma k_j^2}\frac{d\psi_j(x)}{dx} \tag{3}$$

where $p_0$ is the mean undisturbed pressure, the mode shape (at the location $x$) and the wave number of the $j$th mode (which has a natural frequency of $\omega_j$) are denoted by $\psi_j(x)$ and $k_j$, respectively, and $\gamma$ is the ratio of specific heats $C_p$ and $C_v$ of air. Substituting Eq. (2) into Eq. (1), expanding into eigenmodes, and adding a damping term $\xi_j$ [9], yields

$$\frac{d^2\eta_j}{dt^2} + 2\xi_j\omega_j\frac{d\eta_j}{dt} + \omega_j^2\eta_j = \frac{\gamma - 1}{p_0}\int \psi_j \frac{\partial \dot{Q}}{\partial t}dx \tag{4}$$

The left-hand side of Eq. (4) represents a set of $n$ uncoupled linear oscillators that are excited by the forcing terms on the right-hand side. For the Rijke tube, Heckl [32] proposed a modified version of King's law for computing the volumetric rate of heat addition ($\dot{Q}$) which can be modified for a two heater Rijke tube as

$$\frac{d^2\eta_j}{dt^2} + 2\xi_j\omega_j\frac{d\eta_j}{dt} + \omega_j^2\eta_j = \frac{d\dot{Q}'_1}{dt} + \frac{d\dot{Q}'_2}{dt} \tag{5}$$

$$\dot{Q}'_i = \frac{2(\gamma-1)L_w(T_{w,i} - \bar{T})}{p_0 SL\sqrt{3}}\sqrt{\pi\lambda C_v\rho_0\frac{d_w}{2}} \left[\sqrt{\left|\frac{u_0}{3} + u'_{f,i}(t-\tau_i)\right|} - \sqrt{\left|\frac{u_0}{3}\right|}\right]\psi_j(x_{f,i}),\ i = 1,2 \tag{6}$$

where $L$ is the length of the Rijke tube, $L_w$ and $d_w$ are the equivalent length and diameter of the (electrically heated) wire, respectively, $\lambda$ is the thermal conductivity of air, $C_v$ is the constant-volume specific heat capacity of air, $\rho_0$ and $u_0$ are the mean density and (linear) speed of the Rijke-tube air, respectively, $\tau$ is the time lag between the heat transfer and the velocity as a result of thermal inertia (computed by using Lighthill's correlation as $\tau \simeq 0.2(d_w/u_0)$), $(T_w - \bar{T})$ is the mean temperature difference between the heater and the air, $S$ is the cross-sectional area of the Rijke tube, $x_f$ is the heater location, and $u'_f$ is the acoustic velocity perturbation at the heater location. The subscript $i$ takes values 1 and 2 for the primary and secondary heaters, respectively. The frequency-dependent damping coefficient $\xi_j$ is given in Ref. [33] as

$$\xi_j \triangleq \left( c_1 \frac{\omega_j}{\omega_1} + c_2 \sqrt{\frac{\omega_1}{\omega_j}} \right) \tag{7}$$

The first term in Eq. (7) is responsible for the end losses, and the second term represents losses due to boundary layers; and the constants $c_1$ and $c_2$ are the damping coefficients that represent the amount of acoustic damping in the Rijke tube. The modal equations, derived above, can be cast in a linearized state-space form, and the dimensionality of the ordinary differential equation system depends on the number of the selected "significant" acoustic modes. For each mode $j$, there are two states, $\eta_j$ and $\dot{\eta}_j$. This ordinary differential equation system can be solved using a numerical method (e.g., Runge–Kutta).

Bhattacharya et al. [29] modified the above formulation slightly to incorporate the effects of the heater response times and transient behavior, i.e., the heater is switched on at the time $t = 0$ and the system dynamics evolve over time, which may lead to unstable operation provided that the conditions are favorable, which is typically ignored in other similar reduced-order models. The rate of heat loss ($\dot{Q}_{\text{heater}}$) from the heater in a time-step is computed [34] as

$$\dot{Q}_{\text{heater}}(t) = L_w (T_{w,i} - \bar{T}) \left[ \lambda + 2 \sqrt{\pi \lambda C_v \rho_0 \frac{d_w}{2}} \left( \left( 1 - \frac{1}{3\sqrt{3}} \right) \sqrt{\bar{u}} + \frac{1}{\sqrt{3}} \sqrt{\left| \frac{\bar{u}}{3} + u'(t-\tau) \right|} \right) \right] \tag{8}$$

Thus, the temperature of the heater wire ($T_w$) changes in the time interval $[t,\ t + dt)$ as

$$T_w \leftarrow T_w + \frac{(P(t) - \dot{Q}_{\text{heater}}(t))dt}{MCp_{\text{wire}}(T_w)} \tag{9}$$

where $M$ is the mass of the wire mesh that was obtained by measurement, $P(t)$ is the time-dependant (electrical) power supplied to the heater, and $Cp_{\text{wire}}$ is the (temperature-dependent) specific heat capacity of the wire material, which is available from manufacturer's specifications. For computing the temperature in the Rijke tube, the flow domain in the tube is split into three segments; one being the volume between the inlet and first heater, the next being the volume between the two heaters, and the third being the volume between the secondary heater and the outlet. The average temperature, which controls the physics of the system, is measured as the length-weighted average of the segment temperatures.

Bhattacharya et al. [29] also made other modifications to the Galerkin model, which ensure that the reduced-order model closely mimics the operation of the experimental Rijke tube that is used to validate the model.

The ensemble of generated data consists of 250 simulated time-series, each of duration 30 s at a sampling frequency of 10,000 Hz (i.e., a numerical integration time-step of $1 \times 10^{-4}$ s). Each time-series corresponds to a particular fixed value of inlet air-flow rate and heater power, with the system beginning at a stable condition and remaining stable or going unstable depending on the flow-rate and heater power combination. In the Rijke-tube experiments, used to validate this numerical model, the noise was generated by both the process (e.g., heat transfer and fluid-flow) and the sensor (i.e., pressure transducers). In the numerical model of the Rijke tube, the noise has been jointly emulated by inserting an additive zero-mean Gaussian noise of standard deviation of 0.3 Pa, chosen to match the experimental results [29].

**2.2 Target Domain 1: Experimental Rijke Tube.** The first target domain is the experimental Rijke-tube apparatus, constructed at Penn State [15], which has been used to validate the previously described numerical Rijke-tube model. The apparatus consists of a hollow aluminum duct of square cross section. The Rijke tube is 1.5 m long, and each side of the cross section measures 93 mm along each inner wall. This apparatus has two heating elements: a fixed primary heater at 0.375 m from the flow inlet and a movable secondary heater downstream. The heaters are made of compact wire-mesh nichrome for generating thermal power, which emulate the flame in a combustible fuel–air mixture in a real-life combustor. The movable secondary (control) heater has a maximum displacement of 500 mm from the exit end of the tube to the center. Figure 1 depicts the Rijke-tube apparatus.

The pressure signals are measured by an array of eight wall-mounted sensors that are placed at equidistant axial locations; the pressure data are acquired at a sampling rate of 8192 Hz. An Alicat mass flow controller (0–1000 SLPM) controls the air mass-flow rate into the system. The mass-flow rate controls not just the velocity over the heater but also affects the convective heat transfer from the wire mesh to the air and heat loss to the walls. Decouplers are fitted at the inlet and outlet of the tube, which are large hollow enclosures serving the purpose of producing pressure waves under open–open end boundary conditions of the Rijke tube. Additionally, the upstream decoupler attenuates flow fluctuations at the inlet, while the downstream decoupler serves as a heat sink, which allow the hot air exiting by the outlet to be cooled, before it is released to the atmosphere.

The nichrome heaters are designed to be capable of handling high heating loads for sufficiently long times without being oxidized at the high operating temperatures. The square-weave 40-mesh structure of each heater forms acoustically compact sources of thermal energy and allows uniform heating of air over the flow cross section. Power is supplied from a programable DC power supply which can provide powers up to 2000 W per heater. The length of the tube downstream of the primary heater is insulated to prevent heat loss from the walls allowing for maintaining the same initial and running conditions of different experimental runs, and also acts as a safety measure to prevent the operator from coming in contact with the hot metal walls.

The experimental results used in this paper were obtained by operating the Rijke tube at different conditions corresponding to various values of air flow-rate (ranging from 60 to 300 LPM in increments of 10 LPM) and primary heater powers (ranging from 200 to 2000 W in increments of 200 W). The data were collected over windows of 30 s duration for each condition, which captures the transient phenomena. In all the results presented in this paper, the control heater was kept fixed at 1.125 m, and it remained switched off in all experimental trials.

**2.3 Target Domain 2: Experimental Combustor.** The second target domain is an experimental combustor which is a laboratory-scale apparatus, also constructed at Penn State [30]. This experimental combustor, a schematic of which is shown in Fig. 2, consists of an inlet section, an injector, a combustion chamber, and an exhaust section. An optically accessible quartz section is placed downstream of the fuel injector and is followed by a variable length steel section which is set to have various

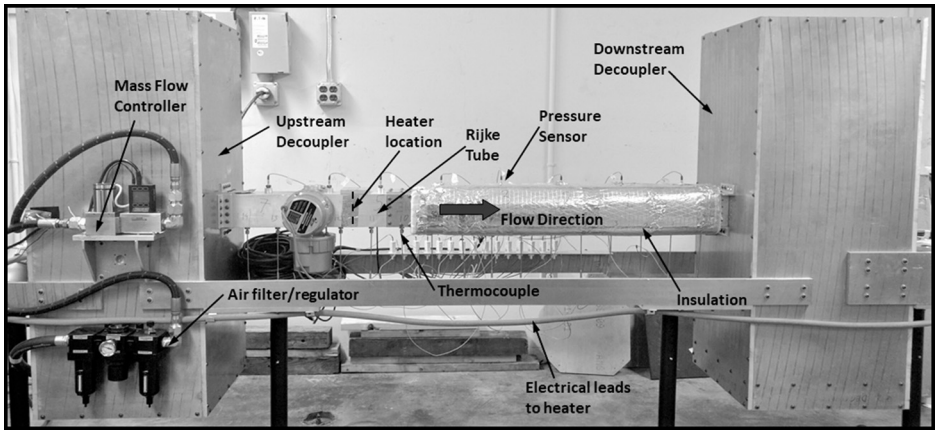


**Fig. 1 Rijke-tube experimental apparatus**

lengths between ∼25 and 59 in. (∼64 and 150 mm) with ∼1 in. (∼25 mm) increments. High pressure air is supplied by a compressor, which is preheated to 250 °C by an electric heater. The airflow rate ranges from 25 to 30 m/s with increments of 5 m/s. The fuel is natural gas (approximately 95% methane), and its flow-rate is adjusted to obtain fuel–air equivalence ratios ($\phi$) of 0.525, 0.55, 0.60, and 0.65. Statistically stationary pressure time-series data of 8 s duration were obtained from pressure probes at a sampling rate of 8192 Hz for different combinations of combustor length and equivalence ratio.

## 3 Symbolic Time-Series Analysis and Its Extensions

Symbolic time-series analysis [22] and its extensions, namely, PFSA [23] and *D*-Markov machines [24], have been used for developing the associated TL algorithms. Although the above topics and their applications to combustion systems (e.g., Ref. [27]) have been extensively reported in literature, this section presents their essential concepts for completeness of the paper and ease of its readability. Following this quick recall, the p-PFSA modification is proposed in Sec. 3.3, and its algorithm follows in Sec. 3.4.

### 3.1 Introduction to Probabilistic Finite State Automata.

This subsection describes the construction of PFSA by first symbolizing the continuous-valued time-series data with a partitioning method of choice [35], which converts the time-series into a string of discrete symbols from a (finite) alphabet. The cardinality of the alphabet is equal to the number of cells in the partitioning used for symbolization.

DEFINITION 1. *A finite state automaton (FSA) G, having a deterministic algebraic structure, is a triple* $(\mathcal{A}, Q, \delta)$ *where:*

- $\mathcal{A}$ *is a (nonempty finite) alphabet, i.e., its cardinality* $|\mathcal{A}|$ *is a positive integer.*
- $Q$ *is a (nonempty finite) set of states, i.e., its cardinality* $|Q|$ *is a positive integer.*
- $\delta : Q \times \mathcal{A} \to Q$ *is a (deterministic) state-transition map.*

DEFINITION 2. *A PFSA, J, is a pair* $J = (G, \pi)$, *where:*

- *The deterministic FSA, G, is called the underlying FSA of the PFSA.*
- *The probability map,* $\pi : Q \times \mathcal{A} \to [0, 1]$, *is called the morph function (also known as symbol generation probability function) that satisfies the condition:* $\sum_{s \in \mathcal{A}} \pi(q, s) = 1$ *for each* $q \in Q$. *The map,* $\pi$, *can be represented by a* $|Q| \times |\mathcal{A}|$ *stochastic matrix,* $\Pi$ *(i.e., each element of* $\Pi$ *is non-negative and each row sum of* $\Pi$ *is unity). In that case, the PFSA is a quadruple, i.e.,* $J = (\mathcal{A}, Q, \delta, \Pi)$.
- *The state-transition probability mass function,* $\tau : Q \times Q \to [0, 1]$, *is constructed by combining* $\delta$ *and* $\Pi$, *which can be structured as a* $|Q| \times |Q|$ *state-transition probability matrix, T. In that case, the PFSA can also be described as a triple, i.e.,* $J = (\mathcal{A}, Q, T)$.

*Remark 3*. The time-series is typically normalized prior to partitioning, with the preferred methods being: (i) scaling the data to a fixed range (e.g., unit range) or (ii) performing $z$-normalization, which converts the data to having zero-mean and unit-variance; this is done to mitigate the effects of spurious noise and bias in the time-series, while ensuring that a fixed set of finite partition boundaries can be used across the full range of data. The fixed partitioning allows comparison of different PFSA, as they evolve, and eliminates the need for recomputation of the partitioning boundaries during testing.

The PFSA structure of a *D*-Markov machine generates symbol strings $\{s_1 s_2 \cdots s_\ell : s_j \in \mathcal{A} \text{ and } \ell \text{ is a positive integer}\}$ on the underlying Markov process [24]. When constructing the *D*-Markov machine, it is assumed that the generation of the next symbol has a dependence only on a *finite* history of the last $D$ or less consecutive symbols, i.e., the (most recent) symbol block of length not exceeding $D$. A *D*-Markov machine is formally defined as follows.

DEFINITION 4. *A D-Markov machine [24] is a PFSA in the sense of Definition 2, and it generates symbols that solely depend on the (most recent) history of at most D consecutive symbols, where the positive integer D is called the depth of the machine. Equivalently, a D-Markov machine is a statistically stationary stochastic process* $\cdots s_{-1} s_0 s_1 \cdots$, *where the probability of occurrence of a new symbol depends only on the last consecutive (at most) D symbols, i.e.,*

$$P[s_n | \cdots s_{n-D} \cdots s_{n-1}] = P[s_n | s_{n-D} \cdots s_{n-1}]$$

*Remark 5*. If the depth of the *D*-Markov machine is unity (i.e., $D = 1$), then the state set, $Q$, and symbol alphabet, $\mathcal{A}$, become equivalent, i.e., the morph matrix, $\Pi$, and the state-transition probability matrix, $T$, are identical if $D = 1$.

Once the alphabet size, $|\mathcal{A}|$, and depth, $D$, are set, the maximum possible number of states is fixed to $|\mathcal{A}|^D$. To generate the morph

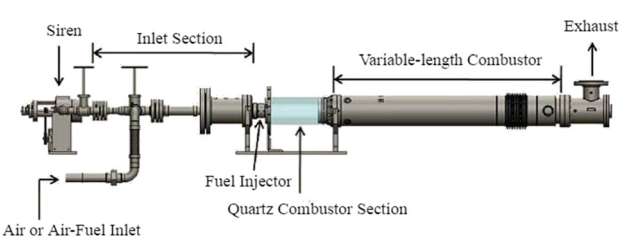


**Fig. 2 Schematic diagram of the combustor apparatus**

matrix, $\Pi$, from a (finite-length) symbol string, the occurrence of each state is sequentially counted, as explained below.

Let $N_{ij}$ denote the number of times the symbol $s_j \in \mathcal{A}$ is emitted from the state $q_i \in Q$. That is

$$\Pi_{ij} = \pi(q_i, s_j) \triangleq \frac{1 + N_{ij}}{|\mathcal{A}| + \sum_\ell N_{i\ell}} \tag{10}$$

Initializing the count of each element to 1 in Eq. (10) ensures that, if no event is generated at a state $q \in Q$, there should be no preference to any particular symbol, making it logical to have $\pi(q, s) = 1/|\mathcal{A}| \, \forall s \in \mathcal{A}$, i.e., the uniform distribution of event generation at the state, $q$. This procedure guarantees that the PFSA, constructed from a (finite-length) symbol string, must have an (elementwise) strictly positive morph map, $\Pi$.

*Remark 6*. The mathematical logic used to generate the morph matrix, $\Pi$, and state-transition probability matrix, $T$, guarantees that these matrices are both stochastic (i.e., each matrix element is positive and each row sum is unity [28]) and ergodic (i.e., every state of the PFSA can be reached within a finite number of iterations irrespective of the starting state [28,36,37]). These matrix properties of stochasticity and ergodicity are necessary for developing the mathematical theory of p-PFSA classification, presented in this paper.

**3.2 Standard Probabilistic Finite State Automata-Based Classification.** This section develops the classification algorithms, based on the s-PFSA, by first training the s-PFSA and then testing the trained s-PFSA on the remainder of the data. Let $\mathcal{C}$ be the number of labeled classes in a setting of supervised classification; then, a total of $\mathcal{C}$ PFSA morph matrices are trained, each corresponding to a certain class, and are denoted as $\Pi^c$, $c = 1, \ldots, \mathcal{C}$. During the testing phase, the s-PFSA algorithm is tested on a dataset belonging to an unknown class. The test time-series is symbolized by using the same partitioning method with the same parameters as it was done during training; and the resulting morph matrix is $\Pi^{\text{tst}}$. The divergence of $\Pi^{\text{tst}}$ from each of the trained morph matrices, $\Pi^c$, $c = 1, \ldots, \mathcal{C}$ is computed; and the class that yields the smallest distance is classified to be the class that the testing time-series is claimed to belong to

$$\text{identified class} = \underset{c \in \{1,2,\ldots,\mathcal{C}\}}{\text{argmin}} \; ||\Pi^{\text{tst}} - \Pi^c|| \tag{11}$$

Thus, decision-making in Eq. (11) is thresholdless. Details of constructing the s-PFSA algorithm are available in Ref. [27].

**3.3 Projection-Based Probabilistic Finite State Automata for Classification.** This subsection develops the proposed method of p-PFSA. In p-PFSA, the state probability vector of the PFSA is projected onto the spaces of feature hyperplanes, corresponding to different operating regimes of the dynamical system under consideration.

Let the state-transition probability matrix $T \in \mathbb{R}^{n \times n}$ (see Definition 2), where $n$ is a positive integer, be an ergodic stochastic matrix [28]. Ergodicity of the stochastic matrix, $T$, implies the existence of exactly one eigenvalue, $\lambda_0 = 1$, and the remaining $m \leq (n-1)$ distinct eigenvalues are located on or inside the unit circle (with center at 0) in the complex plane, i.e., $|\lambda_i| \leq 1$ for $i = 1, 2, \ldots, m$. If some of the eigenvalues are repeated, it is possible that there will be only $m$ linearly independent left eigenvectors, $v_1', \ldots, v_m'$ and $m$ linearly independent right eigenvectors, $u_1, \ldots, u_m$, where $1 \leq m \leq (n-1)$. Each of these eigenvectors could be normalized and ordered for convenience as follows:

$$1 = \lambda_0 > |\lambda_1| > \cdots > |\lambda_m| \geq 0$$

The above eigenvalues, $\lambda_i$, $i = 1, 2, \ldots, m$, are either real or pairs of complex conjugates and their respective left eigenvectors, $\{v_i'\}$, and right eigenvectors, $\{u_i\}$, are also either real or pairs of complex conjugates, because

$$v_i' T = \lambda_i v_i' \Rightarrow \bar{v}_i' T = \bar{\lambda}_i \bar{v}_i' \text{ and } T u_i = \lambda_i u_i \Rightarrow T \bar{u}_i = \bar{\lambda}_i \bar{u}_i$$

where $\bar{v}_i$, $\bar{u}_i$, and $\bar{\lambda}_i$ are complex conjugates of $v_i$, $u_i$, and $\lambda_i$, respectively. Let $v_0'$ and $u_0$ be, respectively, the (normalized) left and right eigenvectors of $T$ with respect to the unique eigenvalue, $\lambda_0 = 1$. The remaining left and right eigenvectors are $v_i'$ and $u_i$, respectively, corresponding to distinct eigenvalues, $\lambda_i$.

*Claim*: For distinct eigenvalues $\lambda_i \neq \lambda_j$, $(i \neq j)$, $i, j \in \{0, 1, \ldots, m\}$, the inner product $\langle v_i, u_j \rangle = 0$, i.e., $v_i \perp u_j$.

*Justification of Claim*: Since $\lambda_i \neq \lambda_j$, at least one of them is nonzero. Without loss of generality, we set $\lambda_i \neq 0$. Then

$$\langle v_i, u_j \rangle = v_i' u_j = \frac{1}{\lambda_i} v_i' T u_j = \frac{\lambda_j}{\lambda_i} v_i' u_j = \frac{\lambda_j}{\lambda_i} \langle v_i, u_j \rangle$$
$$\Rightarrow \left(1 - \frac{\lambda_j}{\lambda_i}\right) \langle v_i, u_j \rangle = 0$$

Since $\lambda_i \neq \lambda_j$, it follows that $(1 - (\lambda_j/\lambda_i)) \neq 0 \Rightarrow \langle v_i, u_j \rangle = 0$.

*End of Justification*

From the above claim, it is concluded that $v_0 \perp u_j$ corresponding to the eigenvalues, $\lambda_j$, $j \in \{1, 2, \ldots, m\}$. In the training phase, $v_0$ and $u_j \, \forall j \in \{1, \ldots, m\}$ are identified; in the testing phase, it is necessary to identify only $v_0^{\text{tst}}$ (corresponding to an observed test time-series belonging to an unknown class) for detection and classification of anomalous events, if any. If there is no anomaly, the vector, $v_0^{\text{tst}}$ should be nearly coincident with $v_0$ and, thus, nearly orthogonal to the $m$-dimensional subspace spanned by $u_j$, $j \in \{1, 2, \ldots, m\}$, which has already been identified in the training phase. However, in the presence of an anomaly, $v_0^{\text{tst}}$ should very likely deviate from $v_0$ that corresponds to the nominal phase.

The key idea behind the development of p-PFSA is to quantify the anomaly as a deviation of $v_0^{\text{tst}}$ from $v_0$ by using the projection of $v_0^{\text{tst}}$ onto the space spanned by $u_j \, \forall j = 1, \ldots, m$. From the perspectives of energy distribution, the absolute values, $|\lambda_i|$, of the eigenvalues signify the energy associated with the (normalized) eigenvector $u_i$; therefore, each $u_i$, $i = 1, \ldots, m$ is weighted as $\sqrt{|\lambda_i|}\, u_i$. The normalized version of $\sqrt{|\lambda_i|}\, u_i$ is $u_i$, because the $u_i$'s were originally generated as normalized vectors.

For a general ergodic stochastic matrix $T$, the right eigenvectors may not be orthogonal to each other. Therefore, the (linearly independent) vectors $\sqrt{|\lambda_i|}\, u_i$, $i = 1, \ldots, m$, are further transformed by Gram–Schmidt orthogonalization as a set of mutually orthogonal vectors $\tilde{u}_i, i = 1, \ldots, m$, i.e., $\langle \tilde{u}_i, \tilde{u}_j \rangle = 0 \, \forall i \neq j$. Having identified the $m$ mutually orthogonal $n$-dimensional vectors $\tilde{u}_i$, the $n$-dimensional weighted error vector, $\mathcal{E}$, is obtained by projection of the vector $v_0^{\text{tst}}$ onto the $m$-dimensional subspace, spanned by $\tilde{u}_i$, $i = 1, \ldots, m$, where $1 \leq m \leq (n-1)$, as

$$\mathcal{E} \triangleq \sum_{i=1}^{m} \langle v_0^{\text{tst}}, \tilde{u}_i \rangle u_i \text{ with } ||\mathcal{E}|| = \sqrt{\sum_{i=1}^{m} |\langle v_0^{\text{tst}}, \tilde{u}_i \rangle|^2} \tag{12}$$

**3.4 Projection-Based Probabilistic Finite State Automata Algorithm.** This subsection develops the p-PFSA algorithm, where a classification problem is addressed with a total number of $\mathcal{C}$ classes (i.e., regimes). In the training phase, a PFSA is generated for each class as described in Secs. 3.1 and 3.2. The state-transition probability matrix, $T^c$ for each class, $c \in \{1, 2, \ldots, \mathcal{C}\}$ is treated as the feature for the respective class (i.e., individual operational regime in the setting of supervised classification). In this paper, $T^c$ is equivalent to $\Pi^c$, because the PFSA is constructed with $D = 1$ (see Remark 5) [27], and each $T^c$ is ergodic and stochastic (see Remark 6); therefore, for each $T^c$, there exists a set of orthogonalized right eigenvectors, $\{\tilde{u}_i^c\}$, where for each class $c$, $i = 1, \ldots, m^c$, and $1 \leq m^c \leq (n-1)$ is the number of linearly

independent eigenvectors for class $c$. Each set of orthogonalized right eigenvectors, $\{\tilde{u}_i^c\}$, generates a hyperplane of dimension, $m^c$, respectively. It is expected that, for different classes representing distinct operational regimes, the hyperplanes will be distinct. That is, a hyperplane belonging to class, $c$, will have its distinct normal direction, $v_0^c$. Classes that are close to each other will have directions close to each other, but they are still distinct.

In the testing (i.e., classification) phase, a time-series is observed, which belongs to an unknown class. The (stochastic and ergodic) state-transition probability matrix, $T^{\text{tst}}$, is generated for this testing time-series, and the left eigenvector $(v_0^{\text{tst}})'$ of the $T^{\text{tst}}$-matrix is obtained corresponding to the unique eigenvalue $\lambda_0^{\text{tst}} = 1$. Then, $v_0^{\text{tst}}$ is projected onto each of the $\mathcal{C}$ hyperplanes that have been generated in the training phase; and the norm of the error vector in each hyperplane is computed by following a procedure similar to that in Eq. (12) as $||\mathcal{E}^{c,\text{tst}}||$, where $c = 1, \ldots, \mathcal{C}$. The class corresponding to the lowest error magnitude is identified to be the class to which the testing time-series belongs, i.e.,

$$\text{identified class} = \underset{c\in\{1,\ldots,\mathcal{C}\}}{\text{argmin}} \; ||\mathcal{E}^{c,\text{tst}}|| \tag{13}$$

In another research article, the authors have demonstrated the efficacy of the p-PFSA algorithm in the task of detecting precursors to a potential fatigue crack developing in structural components under fluctuating stresses [38], where it shows extremely good accuracy and robustness.

*Note*: The depth $D$ is chosen in this paper based on a previous work [27], where $D = 1$ provides good results in identification of combustion instability. In fact, in many other cases (e.g., Refs. [26,36], and [38]), it has been shown that $D = 1$ is sufficient for achieving good accuracy in PFSA, provided that the alphabet size is appropriately selected. Nevertheless, the s-PFSA algorithm is not restricted to $D = 1$ as seen in the earlier work of Sarkar et al. [13], and neither is the proposed p-PFSA algorithm. In general, higher values of $D$ increase the number of states, creating a larger state-transition matrix, which can be handled by the algorithm seamlessly.

## 4 Transfer Learning

This section very succinctly describes the core concept of TL. It also presents the basic methodology of NN-based TL to serve as benchmarks for validation of PFSA-based TL. In the setting of TL, there are a source domain $\mathcal{D}^S$, which is used to learn, and one (or more) target domain $\mathcal{D}^T$, which is to be investigated; and the associated tasks, $\mathcal{T}^S$ and $\mathcal{T}^T$, belong to the domains, $\mathcal{D}^S$ and $\mathcal{D}^T$, respectively.

Let $\mathcal{X}$ be the feature space consisting of feature vectors $X^i \triangleq \{x_1^i, \ldots, x_n^i\}$, where the superscript $i$ of $X^i$ represents either a source $S$ or a target $T$. The associated marginal probability measure $P(X^i)$ is defined for each domain $\mathcal{D}^i = \{X^i, P(X^i)\}$. Given a label space $\mathcal{Y}$, a predictive function $f : \mathcal{X} \to \mathcal{Y}$ is used to classify a feature $(x_j^i)$ to a corresponding label $f(x_j^i)$. Then, a task $\mathcal{T} = \{\mathcal{Y}, f(x_j^i)\}$ is learned from the ordered pairs $\{x_j^i, y_j^i\}$ that are available in the source domain $\mathcal{D}^S$. Given the source and target domains, where either $\mathcal{D}^S \neq \mathcal{D}^T$ or $\mathcal{T}^S \neq \mathcal{T}^T$, the objective is to learn a function $f^S(\cdot)$ and then attempt to "transfer" the learned knowledge to the task of performing a classification in the target domain (i.e., to generate $f^T(\cdot)$).

In both TL problems considered in this paper, the source and target domains are different, with a numerical model acting as the source and Rijke-tube (see Fig. 1) and combustor (see Fig. 2) experimental apparatuses, acting as two different targets. However, the task remains the same, namely, classification between stable and unstable operational modes of combustion from time-series ensembles of pressure oscillations. Therefore, in this case, $\mathcal{D}^S \neq \mathcal{D}^T$ but $\mathcal{T}^S = \mathcal{T}^T$; hence, the TL problem at hand belongs to the category of TTL (see Sec. 1). A succinct introduction to how a neural network-based TL functions is presented in Sec. 4.1, while the proposed PFSA-based transfer learning algorithm is explained in Sec. 4.2.

**4.1 Neural Network-Based Transfer Learning.** This subsection addresses NN-based transfer learning which has been in use over the past couple of decades, and the details are available in literature (e.g., Refs. [1,2], and [7]). In NN-based TL, the net is trained with data available from the source domain. Such a net can be shallow (i.e., having a single or only a few hidden layers) or deep (i.e., having as many hidden layers as desired) and is not restricted to have fully connected layers. It can also have convolution layers, recurrent layers, or LSTM layers. The last layer is typically the classification/regression layer that takes in inputs from the previous (i.e., second from the last) layer to generate the output. With two shallow NNs and another NN with an LSTM layer-based TL architecture, serving as benchmarks, this paper validates the concept of PFSA-based TL between a source domain of the numerical Rijke-tube model [29] and two target domains of experimental apparatuses: (i) a Rijke-tube apparatus [15] and (ii) a laboratory-scale combustor apparatus [30].

There are two general ways to perform NN-based TL: (i) without retraining any prior information or (ii) with retrained partial information. In the first method, the trained NN is directly applied to the target problem by assuming the absence of any training data from the target domain. In the second method, which could be used if a (possibly) small amount of data is available from the target domain, the initial layers of the NN are frozen and only the last (discriminating) layer is trained on the available training data from the target domain. This procedure does not need to train the weights of the hidden layers as they are retained from the (previously) trained net; and only the final layer needs to be trained. It would also reduce the total training time, and less data will be needed to learn the optimal net, because most of the net is pretrained.

**4.2 Probabilistic Finite State Automata-Based Transfer Learning.** This subsection introduces the notion of PFSA-based TL, which apparently has not been reported in open literature. As mentioned at the beginning of Sec. 4, the idea of TL is to learn a task and the associated function for the source domain and then translate this knowledge to the target domain. This is relatively straight-forward to do in a STSA/PFSA setting. For each of the three systems (i.e., the numerical model acting as the source and two experimental apparatuses acting as two different targets) under consideration in this paper, there exists labeled data corresponding to both stable and unstable states. Since these data are transient for both the numerical model and experimental Rijke tube, they have been investigated in a windowed fashion to emulate real-time detection and classification. The rationale is that an online detection and control algorithm tests the evolving dynamics of the observed time-series to identify any (possible) imminent transition from stable to unstable operation and to take appropriate steps toward retention of stability. Windowing the signal leads to the formation of "pseudo-statistically stationary" pressure waves that are less difficult to classify.

In the first method, a thresholdless approach is used, which is similar to the "no-retraining" case in the NN-based TL (see Sec. 4.1). Initially, a PFSA model is learnt to classify the stable and unstable states in the data ensemble of pressure time-series, obtained from the numerical Rijke-tube model by using the formulation described in Sec. 3.2. Subsequently, this PFSA model is used to discriminate between the stable and unstable states in both target domains, namely, Rijke-tube apparatus and combustor apparatus, without any retraining. The rationale is that, although the above three domains are different, the respective pressure-wave signature in each domain is, to some extent, similar to that during the stable and unstable operations.

In the second method, a thresholded approach, which is (in principle) similar to "retraining" of the NN-based TL as described in Sec. 4.1, has been used. In this case, a scalar-valued parameter is computed to discriminate between the two regimes by rearranging Eqs. (11) and (13) as

$$\frac{||\Pi^{\text{tst}} - \Pi^{\text{Stable}}||}{||\Pi^{\text{tst}} - \Pi^{\text{Unstable}}||} \underset{\text{StableRegime}}{\overset{\text{UnstableRegime}}{\gtrless}} \eta_{\text{s-PFSA}} \tag{14}$$

$$\frac{||\mathcal{E}^{\text{Stable,tst}}||}{||\mathcal{E}^{\text{Unstable,tst}}||} \underset{\text{StableRegime}}{\overset{\text{UnstableRegime}}{\gtrless}} \eta_{\text{p-PFSA}} \tag{15}$$

where the thresholds $\eta_{\text{s-PFSA}}$ and $\eta_{\text{p-PFSA}}$ are obtained by using receiver operating characteristics curves [8,27], which are created by plotting the true positive rate (i.e., the rate at which the stable condition is correctly classified to be stable) against the false positive rate (rate at which the stable state is falsely classified to be unstable) over a range of the threshold. The optimal threshold values are identified as those yielding the least errors (i.e., the least number of misclassified data windows) in the given test set. In the thresholded option, for the "transferred to" systems, new thresholds need to be learnt from a small amount of available labeled data, in the absence of which the thresholdless method is recommended.

## 5 Results and Discussions

Before presenting the results of the proposed PFSA-based TL and comparing these results with those of neural network-based TL with different architectures, this section briefly discusses the nature of the time-series data investigated in this paper. Shown in Fig. 3 are the pressure time-series data obtained from each of the three systems: (i) *source domain*: the reduced-order numerical Rijke-tube model (left-hand column), (ii) *target domain 1*: Rijke-tube apparatus (middle column), and (iii) *target domain 2*: combustor apparatus (right-hand column). The first and third rows (from top) in Fig. 3 display the pressure time-series for the stable and unstable regimes, respectively, while the second and fourth rows show the corresponding frequency contents as plots of FFT.

It is seen in Fig. 3 that the stable operation is characterized by low-amplitude chaotic signals with no strong peak in the FFT plots, while the TAI produces almost sinusoidal signals and the corresponding FFT plots show strong peaks at the resonant frequencies and some of the harmonics (e.g., the harmonics in the data from Rijke-tube apparatus). It is noted that the peak frequencies of the numerical Rijke-tube model and the Rijke-tube apparatus have similar resonant frequencies, because the numerical model is validated against the Rijke-tube apparatus [26]. In contrast, the combustor apparatus has significantly different response characteristics, because the combustion chamber has a different geometry and handles much higher energy transfer.

**5.1 Results of Probabilistic Finite State Automata-Based Transfer Learning.** This subsection reports how the proposed PFSA-based TL transfers the knowledge obtained from the reduced-order numerical model to the Rijke-tube apparatus and also to the combustion apparatus; the results are presented by using both s-PFSA and p-PFSA algorithms. Prior to learning a PFSA, certain parameters need to be set; this paper has reported results for three different alphabet sizes, namely, $|\mathcal{A}| = 4, 10$, and 20, while keeping the PFSA depth fixed at $D = 1$. The maximum entropy partitioning [35] is chosen for symbolization; and $z$-normalization (i.e., zero-mean and unit-variance) is carried out on the data window prior to analysis by the PFSA. The data are downsampled by $\text{DS} = 2$, and the moving window is of length $\text{WL} = 50$ ms, while the windowing is done at a rate of 20 Hz, i.e., 20 windows per second. These values are chosen based on the results reported in a recent publication [27].

In the thresholdless option, explained in Sec. 4.2, the PFSA is trained on the data obtained from the source domain of a numerical (Rijke tube) model. A random 70% of the total available transient data (see Sec. 2.1) are used to train the source domain. The remaining 30% of the data are used to test the learnt classifier. Subsequently, the same PFSA model is used to classify the transient pressure time-series in two target systems (i.e., Rijke-tube apparatus and combustor apparatus) individually. As there is no need to relearn anything, the entire data available from these two systems are used for testing the accuracy of TL in two targets.

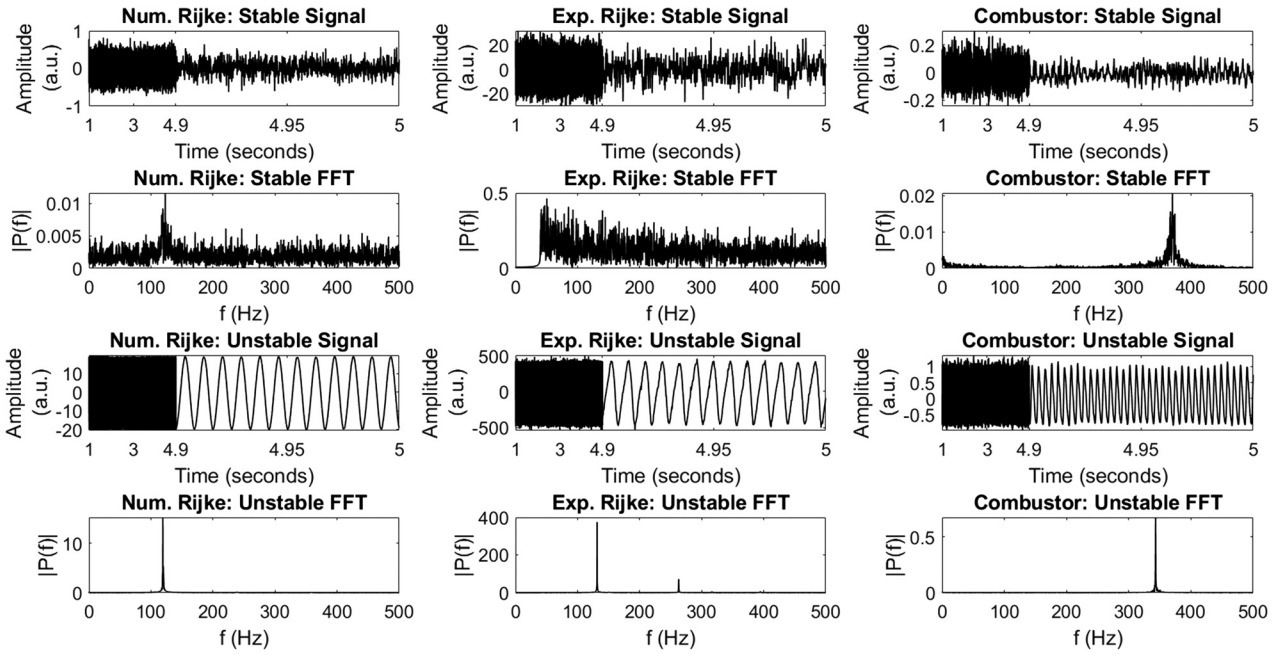


**Fig. 3 Profiles of pressure oscillations and corresponding FFTs under both stable and unstable conditions for (i) reduced-order numerical Rijke-tube model (left-hand column), (ii) experimental Rijke-tube apparatus (middle column), and (iii) combustor apparatus (right-hand column)**

In the thresholded option, after training a PFSA model using 70% of the source domain (i.e., the numerical model) data, the same data are used to learn the thresholds, $\eta_{s-PFSA}$ and $\eta_{p-PFSA}$, as explained in Sec. 4.2. After testing the efficacy of learning the source domain by using the remaining 30% of the data, this knowledge is transferred to classify the two target systems. However, being thresholded, there is a need for further training to find an optimal threshold, given labeled data from the two target systems. In the paper, 5% of the available training data have been used in some trials to relearn the appropriate thresholds, while 10% of the data are used in the others. The procedure would be similar to a situation if the primary data source (e.g., numerical model) is of low or medium fidelity but has abundant data, while the experimental (i.e., target) data are scarce. In contrast, if the target data are of very low quality or are not available at all, the thresholdless option should be chosen.

Table 1 presents the results for both the thresholdless and thresholded options for three alphabet sizes. These results are obtained as an average of 20 independent trials, where the training sets are randomly chosen in each trial, and the PFSAs are constructed from scratch for each trial. The first three columns in Table 1 under "classification error" show the errors made by the trained models in classifying stable and unstable data in the source domain itself, from where the models are initially learnt (e.g., 70% data used to train and the remaining 30% to test). The next three columns show the errors in classifying the experimental data of the Rijke-tube apparatus by using the "transfer learning," while the last three columns show the same for the combustor apparatus.

It is seen in Table 1 that the transfer learning from source (i.e., numerical Rijke tube) data to target 1 (i.e., experimental Rijke tube) data performs very well across all values of alphabet size. The rationale is that the numerical model parameters were by tuned to match experimental results. Addition of more retraining information in the thresholded approach does not change the results significantly. It is noted that there is no winner between s-PFSA and p-PFSA in target 1.

When transferring the learnt knowledge to the task of classifying stable and unstable operation in target 2 (i.e., the combustor apparatus), the thresholdless approach does not fare too well. The rationale is that the combustor apparatus is significantly different from the Rijke-tube apparatus, having different characteristics of pressure waves (see Fig. 3). In the thresholdless approach, p-PFSA performs significantly better than s-PFSA for $|\mathcal{A}| = 10$ and $|\mathcal{A}| = 20$. However, in the thresholded approach, allowing just a minor retraining of the threshold (i.e., using 10% of the available data in this case) increases the accuracy very significantly for target 2. The classification error is ~10%, which closely matches a benchmark case using the same data [16], where the authors had used an FFT-based approach as well as a visibility graph-based method, which produced 10% and 9.45% error, respectively, in classification on 1*s* data windows, as opposed to the shorter 50 ms windows used in this paper. Although these data are inherently difficult to discriminate, the transfer-learnt PFSA achieves similar error rates with an optimal threshold.

**5.2 Comparison of Probabilistic Finite State Automata-Based Transfer Learning and Neural Network-Based Transfer Learning.** This subsection makes a performance comparison of two shallow NNs and one NN having an LSTM layer (NN–LSTM) with p-PFSA for the purpose of transfer learning. The rationale for choosing shallow NNs, instead of deep NNs, is that deep NNs require much larger volumes of training data that are often unavailable for target domains (e.g., Rijke-tube apparatus and combustor apparatus). Therefore, two shallow fully connected NNs are considered, having one and two hidden layers, respectively, which are called NN-1 and NN-2 hereafter. In addition to these two NNs, a third NN, called NN–LSTM, is constructed with a LSTM layer that precedes the fully connected layers. In general, LSTM is an evolution of recurrent neural networks (RNNs) that are designed for sequentially handling the input; and LSTM has found many applications in times-series classification [39] and forecasting [40]. However, LSTM cells may require large datasets for learning good models, and also their training time is significantly large.

For both PFSA-based TL and NN-based TL, the data are first *z*-normalized to remove the effects of amplitude and bias [38] for both thresholdless and thresholded options (see Table 1). In the comparison of p-PFSA-based TL with NN-based TL, the alphabet size is chosen to be $|\mathcal{A}| = 10$, and the remaining PFSA parameters and windowing are the same as those reported in Sec. 5.1.

It is to be noted that in the absence of an LSTM or recurrent layer, the shallow NNs need a fixed length of input, which PFSA and NN–LSTM are not restricted to. Thus, for shallow NNs only, the window size is kept fixed at 250 data-points, which correspond to: (i) 50 ms windows for the source (i.e., numerical Rijke-tube data, sampled at 10,000 Hz and (ii) ~61 ms windows for both target 1 and target 2 (i.e., Rijke tube and combustor experimental) data, sampled at 8192 Hz. The downsampling parameter is DS = 2 for all datasets.

The NN-1 architecture has an input layer with 250 inputs, a hidden layer having 100 neurons with the ReLU (rectified linear unit) activation function, and a binary classifier output layer with two neurons (corresponding to "stable" and "unstable" operations) and the softmax activation function. The relatively deeper net, NN-2, is almost identical with an extra fully connected layer just before the output layer that has 20 neurons with the ReLU activation function. In the retrained configuration, only the final output layer is trained for both NN-1 and NN-2. Deeper fully connected NNs with (up to) four hidden layers and various combinations of layer sizes have also been tested in this work; but since these results are largely similar, they are not reported in this paper due to space limitations. The third net, NN–LSTM, contains an LSTM layer having 100 cells, followed by two fully connected layers having 100 and 20 neurons, respectively, both with the ReLU activation, and ending in an output layer of two neurons. More complex deep NNs (e.g., including convolutional neural networks (CNNs) or more RNN/LSTM layers) have not been considered in this paper as they typically would need much larger volumes of data to train;

**Table 1 Transfer-learnt classification results for both thresholdless and thresholded approaches, using both s-PFSA and p-PFSA, for three values of alphabet size ($|\mathcal{A}|$ = 4, 10, and 20)**

| | | Classification error | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Source domain: Numerical Rijke tube | | | Target domain 1: Experimental Rijke tube | | | Target domain 2: Experimental combustor | | |
| Analysis approach | PFSA technique | $\|\mathcal{A}\| = 4$ | $\|\mathcal{A}\| = 10$ | $\|\mathcal{A}\| = 20$ | $\|\mathcal{A}\| = 4$ | $\|\mathcal{A}\| = 10$ | $\|\mathcal{A}\| = 20$ | $\|\mathcal{A}\| = 4$ | $\|\mathcal{A}\| = 10$ | $\|\mathcal{A}\| = 20$ |
| Thresholdless | s-PFSA | 0.64% | 1.09% | 1.40% | 2.67% | 1.52% | 1.79% | 24.38% | 46.35% | 54.53% |
| | p-PFSA | 1.13% | 1.50% | 1.53% | 1.94% | 1.67% | 1.79% | 27.88% | 26.85% | 33.22% |
| Thresholded | s-PFSA | 0.36% | 0.35% | 0.40% | 1.98% | 1.79% | 1.66% | 12.96% | 10.77% | 9.62% |
| | p-PFSA | 1.06% | 0.98% | 1.02% | 1.89% | 1.71% | 1.77% | 11.10% | 9.56% | 9.32% |

however, this issue is recommended as a topic of future research in Sec. 6.

Table 2 compares the classification performance of p-PFSA with those of NN-1, NN-2, and NN–LSTM, where the "thresholdless" rows correspond to nonoptimal threshold and nonretrained cases for PFSA-based TL and NN-based TL, respectively; and similarly, the "thresholded" rows correspond to optimal threshold and retrained cases for PFSA-based TL and NN-based TL, respectively. The entries in the column "% split" denote the percentage of the total available source-domain data used for training and the percentage of total available target-domain data to retrain. For example, 10-5-5 means that 10% of numerical Rijke-tube data are used to train the initial algorithm, and 5% of data from each of the target domains are used to obtain the best threshold or to retrain the last layer of the NN. For the thresholdless option, no retraining is performed, and hence the corresponding retraining entries are 0% of data.

Table 3 lists the training and testing times for these TL methods for comparison of their time complexities. Of the two shallow nets, only the time taken by NN-1 is reported, because NN-2 takes modestly longer time as expected.

*Note*: A single core processor of a DELL Precision Tower 7910 Workstation running on an Intel® Xeon® E5-2670 CPU has been used to evaluate the computation times for all of the methods. The MATLAB codes of PFSA are available.[1] The NN algorithms are coded in PYTHON using the KERAS library.

Conclusions derived from Tables 1–3 are listed below:

- It is seen from Table 2 that, in the thresholdless option, across the board, the knowledge transfers well to target 1 (i.e., Rijke-tube apparatus), which had been used to validate the numerical model [29] that serves as the source domain for both PFSAs and NNs. However, the NNs do not transfer very well to target 2 (i.e., the combustor apparatus), producing accuracies even poorer than thresholdless s-PFSA option (see Table 1). This is probably due to the difference in respective signal textures (see Fig. 3). Even after adding an LSTM layer, the NN–LSTM-based TL performs poorly without retraining; however, if 70% of the available data are used for training the base network, the accuracy improves and surpasses that of shallow NNs, producing an accuracy similar to thresholdless p-PFSA.
- Even in the thresholded option, where p-PFSA achieves error rates of ∼9.3% (which is similar to the benchmark of ∼9.5% [16]) for transfer learning to the combustor, the NNs yield ∼25% error rate. The performance improves for NN–LSTM if more data (e.g., 40% or 70%) are used to train the base model, which still is not as good as the PFSA performance.
- In all options where fully connected NNs are trained with only 5% or 10% of total training data, it does not learn a sufficiently good model even for classifying the source-domain testing set itself. This is possibly due to the fact that NNs notoriously need a large amount of training data, while PFSA learns a good model even by using a much smaller amount of training data. By increasing the training data, the NN accuracy is expected to eventually surpass the PFSA accuracy, perhaps not significantly. For NN–LSTM, however, the self-classification accuracy is very low when both 5% and 10% of the data are used for training and the performance becomes competitive only with 40–70% of the training data, where it modestly surpasses the other methods.
- In the thresholdless option for the NNs, when the volume of initial training data is low, the transferred models perform quite poorly; but the model accuracy improves with retraining. This trend is not seen for PFSA, because it learns good models from a small amount of data.
- It is seen from Table 3 that the training and testing times (per window) of PFSA are ∼0.3 ms and ∼0.2 ms, respectively, across the board. The training time is higher only for the thresholded option, because some time is required to compute the receiver operating characteristics curve and determine an optimal threshold. However, the training time for the fully connected NNs ranges from ∼0.3 ms to ∼2.0 ms. The training time per window decreases as the amount of training data is increased, indicating a fixed time requirement (in addition to the training time per window) across all trials. The rationale is that, in the initial training iterations, the weights and biases of the recently initialized nets are untrained, and the gradient descent algorithm [8] takes some time to arrive at a pseudo-optimal set of values for the NN. Later iterations using the subsequent data are faster, because it is primarily just fine-tuning the NN. However, the testing time is relatively small, which is about one-tenth of the time taken by PFSA (though both are less than 0.2 ms). In contrast, for NN–LSTM, training and testing times are significantly larger, with a training time ∼30 ms per window, which is ∼2 OOMs larger than those in other methods. Even the testing time is high, at ∼2.5 ms per window, which is ∼1 OOM higher than those of other methods.

*Note*: In this problem, STSA/PFSA is apparently a good choice, because the primary information lies in the signal waveform and symbol transitions; hence, the state-transition probability matrix is capable of capturing the texture of signal waves. On the other hand, a fully connected NN is apparently not well-suited for time-series analyses; more advanced configurations of NN, such as LSTM (as shown in the paper), RNN, or CNN could potentially improve the results. Similarly, hidden Markov models (HMMs) are more suited for time-series analysis, although no HMM-based TL apparently exists in open literature, and hence has not been added as a benchmark. There is no "one fits all" solution in ML, and a fully black-box approach is detrimental to applications of ML in engineering problems.

## 6 Summary, Conclusions, and Future Work

This paper has proposed STSA-based TL and validated the underlying concept by experimentally demonstrating its capability of discrimination between stable and unstable operations in combustion processes. In this context, PFSA models have been trained on the information generated from a numerical model of a Rijke tube [29] as the source domain, which had been validated with the data from a Rijke-tube apparatus [15]. The knowledge of the trained PFSA has been used for classifying stable and unstable operations in: (i) target 1: a Rijke-tube apparatus and (ii) target 2: a combustion apparatus [30]. While the source domain is largely similar to that of target 1, the characteristics of the source are different from those of target 2.

Both thresholded and thresholdless options have been used for classification of stable and unstable operations, where the thresholded option yields better results at the expense of (additional) labeled data from the target domain to compute the appropriate thresholds. The s-PFSA has also been modified by introducing the concept of geometric projection, hence called p-PFSA, to improve upon the classification accuracy almost across the board (more prominently in thresholdless TL). The architecture of PFSA-based TL has been compared with those of two shallow NN-based TL as well as another architecture where an LSTM layer is added to the NN. The results show that the PFSA-based thresholded TL performs very well even when transferring the knowledge from the source domain to classify stable and unstable operations in target 2. In general, NN-based TL algorithms does not perform well unless there are ample data to train. The NN–LSTM architecture needs significantly more training data, which eventually surpasses those of fully connected NNs. Although TL with more complex NNs (that would require even more training data) may do better, it might be an overkill, given the excellent performance of the

[1] https://github.com/Chandrachur92/PFSA

**Table 2 Performance comparison of p-PFSA ($|\mathcal{A}|=10$) and shallow and LSTM neural networks for transfer learning**

| | | p-PFSA | | | NN-1 | | | NN-2 | | | NN–LSTM | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | % split | Numerical Rijke error | Experimental Rijke error | Experimental combustor error | Numerical Rijke error | Experimental Rijke error | Experimental comb error | Numerical Rijke error | Experimental Rijke error | Experimental combustor error | Numerical Rijke error | Experimental Rijke error | Experimental combustor error |
| Thresholdless | 5-0-0 | 1.10% | 2.04% | 27.13% | 6.69% | 26.28% | 73.91% | 6.44% | 22.79% | 74.16% | 21.20% | 40.31% | 74.09% |
| | 10-0-0 | 1.13% | 1.95% | 28.67% | 3.63% | 20.78% | 74.15% | 4.23% | 18.10% | 74.45% | 17.76% | 35.21% | 74.23% |
| | 40-0-0 | 1.09% | 1.96% | 27.30% | 1.45% | 1.18% | 79.97% | 1.42% | 3.22% | 73.82% | 0.45% | 2.84% | 28.65% |
| | 70-0-0 | 1.13% | 1.94% | 27.62% | 1.03% | 1.18% | 73.81% | 1.27% | 1.31% | 73.39% | 0.60% | 3.45% | 25.57% |
| Thresholded | 5-5-5 | 1.03% | 1.78% | 9.43% | 6.39% | 1.90% | 29.28% | 4.75% | 9.26% | 26.64% | 24.06% | 40.31% | 26.24% |
| | 10-5-5 | 0.99% | 1.80% | 9.38% | 3.57% | 2.23% | 28.82% | 2.94% | 8.79% | 26.66% | 20.83% | 40.29% | 25.65% |
| | 10-10-10 | 1.01% | 1.72% | 9.12% | 3.13% | 1.56% | 26.23% | 3.95% | 8.49% | 25.67% | 15.12% | 26.53% | 25.93% |
| | 40-10-10 | 0.94% | 1.69% | 9.33% | 0.91% | 1.32% | 25.98% | 1.45% | 2.54% | 25.68% | 0.80% | 2.21% | 21.16% |
| | 70-10-10 | 0.98% | 1.72% | 9.32% | 0.73% | 1.26% | 26.88% | 0.71% | 1.41% | 25.96% | 0.50% | 1.98% | 17.69% |

**Table 3 Execution times (per data window) of p-PFSA ($|\mathcal{A}| = 10$), NN-1, and NN–LSTM for transfer learning**

| | | p-PFSA | | | | | | NN-1 | | | | | | NN–LSTM | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Source | | Target-1 | | Target-2 | | Source | | Target-1 | | Target-2 | | Source | | Target-1 | | Target-2 | |
| | % split | Training time (in ms) | Test time (in ms) | Retrain time (in ms) | Test time (in ms) | Retrain time (in ms) | Test time (in ms) | Training time (in ms) | Test time (in ms) | Retrain time (in ms) | Test time (in ms) | Retrain time (in ms) | Test time (in ms) | Training time (in ms) | Test time (in ms) | Retrain time (in ms) | Test time (in ms) | Retrain time (in ms) | Test time (in ms) |
| Thresholdless | 5-0-0 | 0.3339 | 0.1952 | — | 0.1956 | — | 0.1833 | 2.1707 | 0.0435 | — | 0.0219 | — | 0.0217 | 33.3496 | 2.4126 | — | 2.3033 | — | 2.2841 |
| | 10-0-0 | 0.3369 | 0.2051 | — | 0.2026 | — | 0.1954 | 1.2273 | 0.0452 | — | 0.0252 | — | 0.0248 | 32.6847 | 2.3736 | — | 2.3245 | — | 2.3027 |
| | 40-0-0 | 0.3400 | 0.2059 | — | 0.2040 | — | 0.1934 | 0.5048 | 0.0572 | — | 0.0255 | — | 0.0254 | 30.3627 | 2.4098 | — | 2.3463 | — | 2.2772 |
| | 70-0-0 | 0.3425 | 0.2086 | — | 0.2054 | — | 0.1950 | 0.3852 | 0.0870 | — | 0.0252 | — | 0.0253 | 27.0580 | 2.2541 | — | 2.1714 | — | 2.1089 |
| Thresholded | 5-5-5 | 0.5416 | 0.1904 | 0.1923 | 0.1877 | 0.1814 | 0.1754 | 1.1898 | 0.0783 | 0.6852 | 0.0461 | 0.3199 | 0.0467 | 35.1011 | 2.4765 | 7.5752 | 2.3532 | 7.2584 | 2.3393 |
| | 10-5-5 | 0.5588 | 0.2012 | 0.2060 | 0.1988 | 0.1966 | 0.1887 | 0.7519 | 0.0853 | 0.7089 | 0.0502 | 0.3437 | 0.0500 | 29.6397 | 2.6472 | 8.2457 | 2.5506 | 7.7519 | 2.5158 |
| | 10-10-10 | 0.5721 | 0.2058 | 0.2108 | 0.2023 | 0.1976 | 0.1909 | 0.7564 | 0.0848 | 0.5278 | 0.0506 | 0.3453 | 0.0506 | 31.9728 | 2.5830 | 8.1912 | 2.5970 | 7.4736 | 2.2785 |
| | 40-10-10 | 0.5688 | 0.2048 | 0.2103 | 0.2029 | 0.1995 | 0.1916 | 0.4834 | 0.1005 | 0.5213 | 0.0505 | 0.3459 | 0.0511 | 31.1396 | 2.6334 | 8.2422 | 2.6003 | 7.5530 | 2.3669 |
| | 70-10-10 | 0.5600 | 0.1997 | 0.2060 | 0.1987 | 0.1949 | 0.1873 | 0.2675 | 0.0577 | 0.2789 | 0.0263 | 0.2106 | 0.0262 | 30.1221 | 2.7126 | 8.1226 | 2.6746 | 7.7313 | 2.4992 |

PFSA-based TL. The time complexities of PFSA-based TL are largely comparable to those of shallow NN-based TL.

In conclusion, the proposed PFSA-based TL is apparently a good alternative to NN-based TL for certain classes of applications. Specifically, the PFSA-based TL should be potentially very useful for industry applications. An example is a real-life combustor, where test data are expensive to collect, but they can be simulated with relative ease; therefore, PFSA-based TL is ideally suited in such situations.

While there are many areas of theoretical and experimental research to improve the proposed PFSA-based TL so that it can be gainfully applied to real-life problems, the following topics are suggested for near-term future research:

- Usage of more advanced tools of neural networks (e.g., CNNs and deep RNNs): Such advancements are expected to yield wider ranges of applications at the expense of extensive training data and computation time.
- Further testing and validation: The proposed PFSA-based TL needs to be tested on different types of platforms and datasets to fully explore its potential capabilities.

## Acknowledgment

The authors are grateful to Professor Domenic Santavicca and Dr. Sudeepta Mondal at Penn State, who kindly provided the experimental data on combustor and Rijke-tube apparatuses, respectively. The first author is also thankful to Indo-US Science and Technology Forum (IUSSTF) for granting him the Research Internship for Science and Engineering (RISE) scholarship.

## Funding Data

- U.S. Air Force Office of Scientific Research (AFOSR) (Grant Nos. FA9550-15-1-0400 and FA9550-18-1-0135; Funder ID: 10.13039/100000181).
- U.S. Army Research Office (Grant No. W911NF-20-1-0226; Funder ID: 10.13039/100000183).

## References

[1] Pan, S. J., and Yang, Q., 2010, "A Survey on Transfer Learning," IEEE Trans. Knowl. Data Eng., **22**(10), pp. 1345–1359.
[2] Zhuang, F., Qi, Z., Duan, K., Xi, D., Zhu, Y., Zhu, H., Xiong, H., and He, Q., 2020, "A Comprehensive Survey on Transfer Learning," Proc. IEEE, **109**(1), pp. 43–76.
[3] Raghu, M., Zhang, C., Kleinberg, J., and Bengio, S., 2019, "Transfusion: Understanding Transfer Learning for Medical Imaging," Advances in Neural Information Processing Systems, Vancouver, Canada, Dec. 8–14, pp. 3347–3357.
[4] Cai, C., Wang, S., Xu, Y., Zhang, W., Tang, K., Ouyang, Q., Lai, L., and Pei, J., 2020, "Transfer Learning for Drug Discovery," J. Med. Chem., **63**(16), pp. 8683–8694.
[5] Ruder, S., Peters, M. E., Swayamdipta, S., and Wolf, T., 2019, "Transfer Learning in Natural Language Processing," Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Tutorials, Minneapolis, MN, June 2–7, pp. 15–18.
[6] Banerjee, B., and Stone, P., 2007, "General Game Learning Using Knowledge Transfer," IJCAI, Hyderabad, India, Jan. 6–12, pp. 672–677.
[7] Fan, Y., Nowaczyk, S., and Rögnvaldsson, T., 2020, "Transfer Learning for Remaining Useful Life Prediction Based on Consensus Self-Organizing Models," Reliab. Eng. Syst. Saf., **203**, p. 107098.
[8] Murphy, K., 2012, *Machine Learning: A Probabilistic Perspective*, The MIT Press, Cambridge, MA.
[9] Matveev, K. I., and Culick, F., 2003, "A Model for Combustion Instability Involving Vortex Shedding," Combust. Sci. Technol., **175**(6), pp. 1059–1083.
[10] Culick, F., 1988, "Combustion Instabilities in Liquid-Fuelled Propulsion Systems," Caltech, Neuilly Sur Seine, France, Report No. AGARD-CP-450.
[11] Rayleigh, J. W. S., 1845, *The Theory of Sound*, Dover Publications, New York.
[12] Rijke, P. L., 1859, "Notiz Über Eine Neue Art, Die in Einer an Beiden Enden Offenen Röhre Enthaltene Luft in Schwingungen zu Versetzen," Ann. Phys., **183**(6), pp. 339–343.
[13] Sarkar, S., Lore, K. G., Sarkar, S., Ramanan, V., Chakravarthy, S. R., Phoha, S., and Ray, A., 2015, "Early Detection of Combustion Instability From Hi-Speed Flame Images Via Deep Learning and Symbolic Time Series Analysis," Annual Conference of the Prognostics and Health Management Society, Coronado, CA, Oct. 18–22, pp. 353–362.
[14] Bhattacharya, C., De, S., Mukhopadhyay, A., Sen, S., and Ray, A., 2020, "Detection and Classification of Lean Blow-Out and Thermoacoustic Instability in Turbulent Combustors," Appl. Therm. Eng., **180**, p. 115808.
[15] Mondal, S., Ghalyan, N. F., Ray, A., and Mukhopadhyay, A., 2019, "Early Detection of Thermoacoustic Instabilities Using Hidden Markov Models," Combust. Sci. Technol., **191**(8), pp. 1309–1336.
[16] Mondal, S., Bhattacharya, C., Chattopadhyay, P., Mukhopadhyay, A., and Ray, A., 2017, "Prediction of Thermoacoustic Instabilities in a Premixed Combustor Based on FFT-Based Dynamic Characterization," AIAA Paper No. 2017-4892.
[17] Kobayashi, T., Murayama, S., Hachijo, T., and Gotoda, H., 2019, "Early Detection of Thermoacoustic Combustion Instability Using a Methodology Combining Complex Networks and Machine Learning," Phys. Rev. Appl., **11**(6), p. 064034.
[18] Sen, U., Gangopadhyay, T., Bhattacharya, C., Mukhopadhyay, A., and Sen, S., 2018, "Dynamic Characterization of a Ducted Inverse Diffusion Flame Using Recurrence Analysis," Combust. Sci. Technol., **190**(1), pp. 32–56.
[19] Kasthuri, P., Pavithran, I., Pawar, S. A., Sujith, R., Gejji, R., and Anderson, W., 2019, "Dynamical Systems Approach to Study Thermoacoustic Transitions in a Liquid Rocket Combustor," Chaos: Interdiscip. J. Nonlinear Sci., **29**(10), p. 103115.
[20] Silva, C. F., Magri, L., Runte, T., and Polifke, W., 2017, "Uncertainty Quantification of Growth Rates of Thermoacoustic Instability by an Adjoint Helmholtz Solver," ASME J. Eng. Gas Turbines Power, **139**(1), p. 011901.
[21] Magri, L., 2019, "Adjoint Methods as Design Tools in Thermoacoustics," ASME Appl. Mech. Rev., **71**(2), p. 020801.
[22] Dupont, P., Denis, F., and Esposito, Y., 2005, "Links Between Probabilistic Automata and Hidden Markov Models: Probability Distributions, Learning Models and Induction Algorithms," Pattern Recognit., **38**(9), pp. 1349–1371.
[23] Ray, A., 2004, "Symbolic Dynamic Analysis of Complex Systems for Anomaly Detection," Signal Process., **84**(7), pp. 1115–1130.
[24] Mukherjee, K., and Ray, A., 2014, "State Splitting and Merging in Probabilistic Finite State Automata for Signal Representation and Analysis," Signal Process., **104**, pp. 105–119.
[25] Daw, C. S., Finney, C. E. A., and Tracy, E. R., 2003, "A Review of Symbolic Analysis of Experimental Data," Rev. Sci. Instrum., **74**(2), pp. 915–930.
[26] Bhattacharya, C., and Ray, A., 2020, "Online Discovery and Classification of Operational Regimes From an Ensemble of Time Series Data," ASME J. Dyn. Syst., Meas., Control, **142**(11), p. 114501.
[27] Bhattacharya, C., O'Connor, J., and Ray, A., 2020, "Data-Driven Detection and Early Prediction of Thermoacoustic Instability in a Multi-Nozzle Combustor," Combust. Sci. Technol., In press, pp. 1–32.
[28] Berman, A., and Plemmons, R., 1994, *Nonnegative Matrices in the Mathematical Sciences*, SIAM, Philadelphia, PA.
[29] Bhattacharya, C., Mondal, S., Ray, A., and Mukhopadhyay, A., 2020, "Reduced-Order Modelling of Thermoacoustic Instabilities in a Two-Heater Rijke Tube," Combust. Theory Modell., **24**(3), pp. 530–548.
[30] Kim, K. T., and Santavicca, D. A., 2013, "Interference Mechanisms of Acoustic/Convective Disturbances in a Swirl-Stabilized Lean-Premixed Combustor," Combust. Flame, **160**(8), pp. 1441–1457.
[31] Culick, F., 1976, "Nonlinear Behavior of Acoustic Waves in Combustion Chambers—I," Acta Astronaut., **3**(9–10), pp. 715–734.
[32] Heckl, M. A., 1988, "Active Control of the Noise From a Rijke Tube," J. Sound Vib., **124**(1), pp. 117–133.
[33] Matveev, K. I., 2003, "Thermoacoustic Instabilities in the Rijke Tube: Experiments and Modeling," Ph.D. thesis, California Institute of Technology, Pasadena, CA.
[34] Heckl, M. A., 1990, "Non-Linear Acoustic Effects in the Rijke Tube," Acustica, **72**, pp. 63–71.
[35] Rajagopalan, V., and Ray, A., 2006, "Symbolic Time Series Analysis Via Wavelet-Based Partitioning," Signal Process., **86**(11), pp. 3309–3320.
[36] Ghalyan, N. F., and Ray, A., 2020, "Symbolic Time Series Analysis for Anomaly Detection in Measure-Invariant Ergodic Systems," ASME J. Dyn. Syst., Meas., Control, **142**(6), p. 061003.
[37] Ghalyan, N. F., and Ray, A., 2021, "Measure Invariance of Ergodic Symbolic Systems for Low-Delay Detection of Anomalous Events," Mech. Syst. Signal Process., **159**, p. 107746.
[38] Bhattacharya, C., Dharmadhikari, S., Basak, A., and Ray, A., 2021, "Early Detection of Fatigue Crack Damage in Ductile Materials: A Projection-Based Probabilistic Finite State Automata Approach," ASME Lett. Dyn. Syst. Control, **1**(4), p. 041003.
[39] Karim, F., Majumdar, S., and Darabi, H., 2019, "Insights Into LSTM Fully Convolutional Networks for Time Series Classification," IEEE Access, **7**, pp. 67718–67725.
[40] Cao, J., Li, Z., and Li, J., 2019, "Financial Time Series Forecasting Model Based on CEEMDAN and LSTM," Phys. A: Stat. Mech. Appl., **519**, pp. 127–139.