Advances in Applied Mathematics and Mechanics
Adv. Appl. Math. Mech., Vol. **xx**, No. 1, pp. 1-41

DOI: 10.4208/aamm.OA-2022-0163
xxx 2024

A Surrogate Reduced Order Model of the Unsteady Advection Dominant Problems Based on Combination of Deep Autoencoders-LSTM and POD

Mahdi Kherad[1,*], Mohammad Kazem Moayyedi[2] and Faranak Fotouhi[1]

[1] *Department of Computer Engineering and IT, University of Qom, Iran*
[2] *CFD, Turbulence and Combustion Research Lab., Department of Mechanical Engineering, University of Qom, Iran*

Received 18 June 2022; Accepted (in revised version) 3 June 2023

Abstract. Model Order Reduction is an approximation of the main system so that the simplified system retains important features of the main system. Deep learning technology is a recent breakthrough in artificial neural networks that can find more hidden information from the data. In this paper, a non-intrusive reduced order model (NIROM) based on combining deep neural networks (DNNs) and POD abilities, namely FAE-CAE-LSTM is presented. This method combines the obtained features based on Fully connected autoencoders (FAE), Convolutional autoencoders (CAE), and POD and then, a deep Long short-term memory network is trained by obtained features to predict the pressure and velocity fields at future time instances. We investigate the performance of the proposed methodology by solving two well-known canonical cases: a strong shear flow exhibiting the Kelvin–Helmholtz instability, and flow past a cylinder. The performance of the proposed FAE-CAE-LSTM method in future state prediction of the flow is compared with other NIROM methods such as CAE-LSTM, autoencoder-LSTM, autoencoder-DMD and POD-RNN based models. Results show that the FAE-CAE-LSTM method is considerably capable of predicting fluid flow evolution and obtains the best results in the prediction of the pressure and velocity fields in future time instances.

AMS subject classifications: 35Q35, 68T07, 35K40

Key words: Reduced order model, deep learning, autoencoder, long short-term memory, proper orthogonal decomposition.

1 Introduction

Model order reduction (MOR) is used to reduce the size of the large-scale structure and the dimensions of the dynamic problem. By obtaining the dominant modes, the dimen-

*Corresponding author.
Email: ma.kherad2@gmail.com (M. Kherad)

sions of the corresponding state space are reduced and the approximation of the original model is calculated with acceptable accuracy. Model order reduction is used in various fields, for example, nonlinear large-scale systems [1], ocean modeling [2], sensor position optimization [3], air pollution model [4], shape optimization [5], aerospace [6], optimal control [7] and neutron problems [8]. Model order reduction methods are proposed to achieve the main dynamics of the process through which an efficient approximate computational solution can be obtained [9]. Mathematically, the goal in MOR is to reduce the model, that is, to represent high-dimensional data in a low-dimensional subspace that is necessary to reduce data processing and calculation costs [10].

The first model order reduction works discussed that larger matrices can be simplified to smaller matrices while still having a good approximation [11, 12]. Early MOR methods were published in the 1980s and 1990s. Subsequently, new methods were proposed, such as truncated balanced realization [13], Hankel-norm [14], aggregation method [15], Pade approximation [16], moment matching [17], Routh approximation [18], dynamic mode decomposition (DMD) [19], Koopman analysis [20], and proper orthogonal decomposition (POD), also known as Karhunen-Loeve decomposition and principal component analysis (PCA) [21]. In general, the conventional method of reduced order model is POD because the modes reduced by the POD are mathematically optimal for each dataset. After reducing the model, the next step is to use the reduced modes for modeling. The POD method reduces the number of state variables by mapping the dynamics to a linear subspace provided by the principal components of a data matrix. A very popular technique for this is the Galerkin projection method [22]. The Galerkin method uses spatio-temporal dynamics obtained by model order reduction methods such as POD which can predict them in time rather than governing equations. Galerkin models typically do not take into account spatial variations in flow and become unstable under different conditions, even in conventional cases. Therefore, significant research efforts have been devoted to improving the stability of Galerkin models, and some efforts have focused on formulating without Galerkin. DMD is a method for analyzing the time evolution of a dynamical system and provides an accurate decomposition of complex systems into coherent spatiotemporal structures that may be used for short-time future state prediction and control. It was started in the fluid dynamics community and was first presented by Schmid and Sesterhenn [19]. It is a data-driven equation-free method based on the ability of the singular value decomposition (SVD), and it has been utilized for modal analysis of a diversity of fluid flows.

Recently, deep learning has attracted a lot of attention [23,24]. A deep neural network (DNN) is at the core of deep learning and is a computing system inspired by brain architecture. DNN has shown its strength in overcoming some of the complexities and challenges of machine vision tasks such as large image classification, speech recognition, and more. A deep neural network usually consists of an input layer, an output layer, and several hidden middle layers with several neurons in each layer. In the learning process, each layer converts its input data into an abstract representation and transmits the signal to the next layer. Some activation functions, which act as nonlinear transformers on the

input signal, are used to determine whether a neuron is active or not. Deep learning technology is a recent advancement in artificial neural networks that can find more hidden information from data. Deep learning has the advantage of processing data in its raw form, training the nonlinear system with different levels of representation and predicting the data [25]. Recently, these advances have been made in areas such as image processing, video and speech recognition, genetics, and disease diagnosis [26,27]. In the review paper [28], they introduce three typical data-driven methods, including system identification, feature extraction and data fusion. System identification is the theory to construct mathematical models from system measurements, which lays the foundation for system control and analysis. Feature extraction methods mainly refer to the mode decomposition methods that identify coherent structures (flow modes) from a large flow dataset. Data fusion generally refers to the process of combining data and information from multiple sources, in order to refine, estimate, and achieve a greater understanding of the data. In particular, the main approaches to improve the performance of data-driven models in accuracy, stability and generalization capability are reported. The efficacy of data-driven methods in modeling unsteady aerodynamics is described by several benchmark cases in fluid mechanics and aeroelasticity.

Reduced order models can be divided into intrusive ROM (IROM) and non-intrusive ROM (NIROM) groups in terms of the need for governing equations. IROM is dependent on the governing equations used to generate the full order model, such as the Euler or Navier–Stokes equations, and requires code modification of high-fidelity models. It retains much of the physical characteristics from the original system due to its intrusiveness. The most common IROM method is the POD-Galerkin [29,30], in which the governing equations are projected onto the reduced basis extracted by the POD. However, the main problems of POD-Galerkin are stability and non-linear inefficiency. To overcome these challenges, different models have been offered such as discrete empirical interpolation method [31,32], Petrov-Galerkin projection [33,34,35], the Gauss-Newton method with approximated tensors [36], L1-norm minimization [37]. Against, NIMOR is a data-driven approach that is independent of the governing equations and the interpolated coefficients across the parameters are obtained without projection on the governing equation. It does not require access to the full-order model operators. Thus, the construction of the surrogate model will require only the set of high-fidelity model solutions (snapshots) and no information about the governing equations. Although cubic interpolation [38] or radial basis functions [39] were originally used for the interpolation, NIROM methods have since embraced machine learning, using a variety of neural networks for this purpose, including autoencoders in combination with long short-term memory networks [40,41], multilayer perceptrons [42], and Gaussian process regression [43,44]. San et al. [45] consider the use of artificial neural networks to stabilize POD based ROMs for quasi-stationary geophysical turbulent flows. An extreme learning machine approach is offered for computing an eddy-viscosity closure dynamically to combine the effects of the truncated modes. Lu et al. [46] introduce a novel NIMOR when an inexpensive low-fidelity model and a heavy high-fidelity model are available. The method

relies on POD to create the high-fidelity reduced basis and a shallow MLP to learn the high-fidelity reduced coefficients. In the work of Xiao et al. [44] a Gaussian Process Regression (GPR) machine learning method is composed with POD to construct a NIROM, which is exploited for urban flows. Pawar et al. [47] present a modular deep neural network model for data-driven ROM of dynamical systems relevant to fluid flows.They offer different deep neural network models which numerically predict the evolution of dynamical systems by learning from either using discrete state or slope information of the system. San et al. [48] suggest a supervised machine learning model for the NIROM of unsteady fluid flows to prepare accurate predictions of non-stationary state variables when the control parameter values are unstable. Their method uses a training process from full-order scale direct numerical simulation data projected on POD modes to get an ANN model with lower memory requirements.

Today, deep learning in the field of model reduction is also used as one of the important solutions. In the context of ROM, both dimensionality reduction and future state prediction could be advanced from deep learning strategies. Recurrent neural networks are of particular interest to fluid mechanics due to their ability in learning and prediction of the sequential data. The renewed interest in RNNs has largely been attributed to the development of the LSTM algorithms. Several studies have been conducted to construct an NIROM based on POD modes. Autoencoder is an unsupervised learning algorithm used to representation learning, data compression, data encoding, and to find hidden factors. This algorithm consists of two parts: encoder and decoder. When a data sample is considered as the input of the algorithm, this neural network encodes the data with hidden factors in the hidden layer. It then generates the output with the decoder using hidden factors. The purpose of this algorithm is to reduce the difference between input and output. Neural network autoencoders can be built in various structures to achieve sufficient dimensionality reduction. A standard, fully connected autoencoder (FC-AE) is a special type of feed-forward neural network that is trained to learn the identity map by fully connected layers. Moreover, as embedding Convolutional Neural Network (CNN) can facilitate neural network models to perform powerful feature detection. More recently, the population of a convolutional neural network (CNN)-based AE has also been increasing thanks to the concept of filter sharing in CNN, which is suitable to handle high-dimensional fluid data set.

Pawar et al. [49] introduce a modular hybrid analysis and modeling (HAM) approach to account for hidden physics in reduced order modeling (ROM) of parameterized systems relevant to fluid dynamics. They employ POD as a compression tool to construct orthonormal bases and Galerkin projection (GP) as a model to build the dynamical core of the system. Their proposed methodology hence compensates structural or epistemic uncertainties in models and utilizes the observed data snapshots to compute true modal coefficients spanned by these bases. The GP model is then corrected at every time step with a data-driven rectification using a LSTM neural network architecture to incorporate hidden physics. A Grassmannian manifold approach is also adapted for interpolating basis functions to unseen parametric conditions. The control parameter governing the

system's behavior is thus implicitly considered through true modal coefficients as input features to the LSTM network. The effectiveness of the HAM approach is discussed through illustrative examples that are generated synthetically to take hidden physics into account.

In [50], an unsteady aerodynamic ROM with varying Mach numbers based on the LSTM network is proposed. Unlike most of the works in the previous study, the proposed model is capable of treating a large number of training samples, with the capability of predicting aerodynamic loads in a wide range of Mach numbers. The performance of the proposed model is evaluated by predicting the aerodynamic and aeroelastic responses of a NACA 64A010 airfoil in the transonic flow. It is proved that the results from the proposed model agree with those of the CFD solver very well.

Wang et al. introduce a new approach for NIROMs which is according to POD and deep LSTM technique and used to fluid issues [51]. The LSTM is used to create a set of hypersurfaces that show the reduced fluid dynamic system. In [52], Wang et al. deep autoencoder is applied for reduce dimensions of distributed parameter systems. Next, the reduced representations are applied to make a ROM. Deep learning is exploited to perform reduce dimensions, however does not take the superiority of its predictive ability. Gonzalez et al. [40] suggest a deep learning based method for nonlinear ROM that is derived from projection-based ROM where the purpose is to recognize some optimal low-dimensional representation and evolve it in time. Their method creates a modular model including a deep convolutional autoencoder and an improved LSTM network. Ahmed et al. [53] present an uplifted ROM method by combine of standard projection-based IROM with LSTM embedding. Their method has three components or layers. In the first layer, they use a projection-based IROM method to model dynamics represented by the maximum modes. The second layer includes a LSTM network to consider residuals outside this truncation. In the third layer, they enhance their approximation and develop the solution subspace to recover some of the flow's details by learning a map between the large and smaller scales. Vlachas et al. [54] suggest a LSTM-based data-driven method for predict the state obtained by chaotic systems exploiting the short-term record of the reduced order states. The predicted derivatives are then used for one-step forward prediction of the high-dimensional dynamics. In addition, they created a composite model incorporating LSTM and mean stochastic model for data-driven to develop the predictability of the suggested model. Mohan and Gaitonde [55] offered a NIROM using LSTM and POD for flow control applications through a detailed analysis on different ROM LSTM training and testing hyperparameter tuning parameters. Rahman et al. [56] developed a NIROM method for large-scale quasistationary systems. Their method uses the time series prediction ability of deep LSTM network model. Kherad et al. [57] present a NIROM Framework Based on LSTM and POD/DMD Modes. They consider LSTM architecture to develop their data-driven ROM. They investigate the performance of the proposed methodology by solving two well-known canonical cases: a strong shear flow exhibiting the Kelvin–Helmholtz instability, and two-dimensional and unsteady mass diffusion equation. They use a deep LSTM network to learn the time dynamics of simulated two-

dimensional and unsteady diffusion equation data and use POD/DMD to generate the order reduction model.

Eivazi et al. introduce a novel data-driven technique based on the power of deep neural networks for reduced order modeling of the unsteady fluid flows. An autoencoder network is used for nonlinear dimension reduction and feature extraction as an alternative for singular value decomposition (SVD). Then, the extracted features are used as an input for long short-term memory network (LSTM) to predict the velocity field at future time instances. The proposed autoencoder-LSTM method is compared with non-intrusive reduced order models based on dynamic mode decomposition (DMD) and proper orthogonal decomposition (POD). Results show that the autoencoder-LSTM method is considerably capable of predicting fluid flow evolution, where higher values for coefficient of determination R2 are obtained using autoencoder-LSTM compared to other models [58]. Fukami et al. investigate the capability of neural network-based model order reduction, i.e., autoencoder (AE), for fluid flows. As an example model, an AE which comprises of a convolutional neural network and multi-layer perceptrons is considered in this study. The AE model is assessed with four canonical fluid flows, namely: (1) two-dimensional cylinder wake, (2) its transient process, (3) sea surface temperature, and (4) $y-z$ sectional field of turbulent channel flow, in terms of a number of latent modes, a choice of nonlinear activation functions, and a number of weights contained in the AE model. They find that the AE models are sensitive against the choice of the aforementioned parameters depending on the target flows [59].

Xiang et al. by comparing four types of encoding-decoding models (i.e., the proper orthogonal decomposition models, autoencoders with linear/ nonlinear fully connected neural networks and autoencoders with Convolutional Neural Network (AE-CNN)) find that the AE-CNN model with advantages of portable model size and convenient updating procedure is capable for high resolution simulations of urban airflow with dynamic boundary conditions at a low computational cost [60]. Bukka and Magee present two deep learning-based hybrid data-driven reduced order models for the prediction of unsteady fluid flows. These hybrid models rely on recurrent neural networks (RNNs) to evolve low dimensional states of unsteady flow. The first model projects the high-fidelity time series data from a finite element Navier-Stokes solver to a low-dimensional subspace via proper orthogonal decomposition (POD). The time-dependent coefficients in the POD subspace are propagated by the recurrent net (closed-loop encoder decoder updates) and mapped to a high-dimensional state via the mean flow field and POD basis vectors. This model is referred as POD-RNN. The second model, referred to as convolution recurrent autoencoder network (CRAN), employs convolutional neural networks (CNN) (instead of POD), as layers of linear kernels with nonlinear activations, to extract low-dimensional features from flow field snapshots. Maulik et al. demonstrate that an encoding using convolutional autoencoders (CAEs) followed by a reduced-space time evolution by recurrent neural networks overcomes advection-dominated PDEs limitation effectively. They demonstrate that a truncated system of only two latent-space dimensions can reproduce a sharp advecting shock profile for the viscous Burgers equation

with very low viscosities, and a twelve-dimensional latent space can recreate the evolution of the inviscid shallow water equations. Additionally, the proposed framework is extended to a parametric reduced-order model by directly embedding parametric information into the latent space to detect trends in system evolution [61]. Phillips et al. developed Reduced-order models based on an autoencoder and a novel hybrid SVD-autoencoder. These methods are compared with the standard POD-Galerkin approach and are applied to two test cases taken from the field of nuclear reactor physics [62]. Gonzalez and Balajewicz propose a deep learning-based strategy for nonlinear model reduction that is inspired by projection-based model reduction where the idea is to identify some optimal low-dimensional representation and evolve it in time. Their approach constructs a modular model consisting of a deep convolutional autoencoder and a modified LSTM network. The deep convolutional autoencoder returns a low-dimensional representation in terms of coordinates on some expressive nonlinear data-supporting manifold. The dynamics on this manifold are then modeled by the modified LSTM network in a computationally efficient manner. They demonstrate their model on three illustrative examples each highlighting the model's performance in prediction tasks for fluid systems with large parameter-variations and its stability in long-term prediction [40].

In [63,64], the CNN-AE is utilized to map high-dimensional flow fields obtained by direct numerical simulation (DNS) into a low-dimensional latent space while keeping their spatially coherent information. The LSTM is then utilized to predict a temporal evolution of the latent vectors obtained by the CNN-AE.

Hasegawa et al. [65] propose a method to construct a reduced order model with machine learning for unsteady flows. In their machine-learned reduced order model (ML-ROM), first, the CNN-AE is trained using direct numerical simulation (DNS) data so as to map the high-dimensional flow data into low-dimensional latent space. Then, the LSTM is utilized to establish a temporal prediction system for the low-dimensionalized vectors obtained by CNN-AE. As a test case, they consider flows around a bluff body whose shape is defined using a combination of trigonometric functions with random amplitudes. They also focus on the influence of two main parameters: (1) the latent vector size in the CNN-AE, and (2) the time step size between the mapped vectors used for the LSTM. The results show that the ML-ROM works well even for unseen shapes of bluff bodies when these parameters are properly chosen, which implies great potential for the present type of ML-ROM to be applied to more complex flows.

The authors in [66] presented an approach to improve computational fluid dynamics simulation forecasts of air pollution using deep learning. Once the reduced order model (ROM) is obtained via POD, a long short-term memory network (LSTM) is adversarially trained on the ROM to make forecasts. Once trained, the adversarially trained LSTM outperforms a LSTM trained in a classical way.

For dimensionality reduction, Carlberg et al. [67] propose a novel hierarchical approach. First, the method reduces the number of degrees of freedom within each element of the high-order discretization by applying autoencoders. Second, the methodology applies POD to compress the global vector of encodings. For dynamics learning, they

propose to apply regression techniques to learn the discrete-time velocity characterizing the time evolution of this low-dimensional state.

In [68], the authors introduce a technique to construct parametric non-intrusive ROMs with quantified uncertainty during latent-space time evolution of dynamical systems. They detail the use of Gaussian processes (GPs) that are customized for the time-evolution of parametric nonlinear dynamical systems and demonstrate the ability of the proposed time-evolution algorithm for systems compressed by linear reduced-basis methods such as POD, as well as nonlinear compression frameworks such as variational and convolutional autoencoders.

In [69], the authors develop a reduced-order model for chaotic electroconvection at high electric Rayleigh number. Coherent structures (modes) are extracted from temporally and spatially resolved charge density fields via POD. A nonlinear model is then developed for the chaotic time evolution of these coherent structures using the sparse identification of nonlinear dynamics (SINDy) algorithm, constrained to preserve the symmetries observed in the original system. Fukami et al. [70] employ the CAE to map a high-dimensional dynamics into a low-dimensional latent space. The SINDy then seeks a governing equation of the mapped low-dimensional latent vector. Temporal evolution of high-dimensional dynamics can be provided by combining the predicted latent vector by SINDy with the CNN decoder which can remap the low-dimensional latent vector to the original dimension.

Furthermore, finite-dimensional data-driven approximations based on the Koopman operator have received considerable attention in recent years, especially for problems involving complex spatio-temporal behaviors such as unstable fluid flows. Standard DMD, implicitly uses linear observables of the system state. Extended DMD [71] and Hankel-DMD [72] were introduced to include a richer set of observables that spans a Koopman invariant subspace. Several works have recently been done to introduce fully data-driven methods to learning Kopman embedding using DNNs [73,74,75]. Morton et al. [76] presented a method for learning the forced and unforced dynamics of airflow over a cylinder directly from CFD data. Their approach, grounded in Koopman theory, is shown to produce stable dynamical models that can predict the time evolution of the cylinder system over extended time horizons. However, it has been shown by Khodkar et al. [77] that the linear composition of a finite number of modes may not precisely reconstruct the nonlinear specifications of chaotic dynamics for an appropriately long term of time. They indicated that adding nonlinearities to the linear model as a forcing term leads to an excellent short-term prediction of several well-known prototypes of chaos. Furthermore, Eivazi et al. [78] indicated that the Koopman framework with nonlinear forcing gives precise predictions of the dynamical evolution of the coefficients of a low-order model for near-wall turbulence.

Adding nonlinear to the dimension reduction step using nonlinear activation functions in the autoencoder network can effectively increase the efficiency of the dimension reduction step. Since fully-connected autoencoders require that the input data be flattened into an 1D array, the local spatial relations between values are eliminated and can

only be recovered by initially considering dense models. In addition to using a fully-connected autoencoder directly to complex, high-dimensional simulation or experimental data, we apply it to a vectorized feature map of a much lower-dimension obtained from a deep convolutional network acting directly on the high-dimensional data. We attempt to find to utilize local correlations present in many physics-based data through convolutional neural networks. Moreover, the good ability of the LSTM network in the learning of the complex systems dynamic is the reason for implementing the LSTM network for future time prediction rather than a linear mapping of the previous observations to the future time instances. On the other hand, the power of POD to identify the features of flow most important for reconstructing a dataset has been shown in previous research. Therefore, in this paper presents a novel NIROM based on combine deep neural networks (DNNs) and POD abilities. A framework for nonlinear dimensionality reduction and dominant features extraction of different flows by combining the obtained features based on Fully connected autoencoders (FAE), Convolutional autoencoders (CAE), and POD is developed, and then, a unified training strategy is introduced that a deep LSTM model is trained by obtained features from FAE, CAE, and POD for prediction of the flow evolution. A sequence of features is the input of a LSTM neural network model, and the output is the flow field in the future time step. The benefit of the proposed method, in comparison to POD-GP, is the improved compression ratios obtained by the nonlinear embeddings of the AEs and the equation-free evolution of the state using the LSTM. Train and test sets are acquired from numerical simulation of a strong shear flow exhibiting the Kelvin–Helmholtz instability, and flow past a cylinder. The performance of the proposed FAE-CAE-LSTM method in future state prediction of the flow is compared with other NIROM methods such as CAE-LSTM [61], autoencoder-LSTM [58], autoencoder-DMD [58] and POD-RNN [79] based models. The main novelties of this research study are listed below:

1. Propose a novel data-driven approach based on combining deep fully connected and convolutional autoencoders, LSTM and POD methods for NIROM of complex unsteady fluid flows.

2. Implementation of deep fully connected and convolutional autoencoders and POD method and concatenation of all obtained feature for nonlinear dimensionality reduction and feature extraction of the fluid systems.

3. Future state estimation of a nonlinear dynamical system using LSTM network from it's the features extracted by the deep fully connected and convolutional autoencoders and POD method.

4. Comparison of linear and nonlinear NIROM methods.

Even though the idea of developing nonintrusive ROM based on replacing Galerkin projection with LSTM is similar to the proposed FAE-CAE-LSTM method, while previous works use fully connected autoencoder (FAE) or convolutional autoencoder (CAE), or

POD method separately, not a combination of them for nonlinear dimensionality reduction and feature extraction of the fluid systems, the proposed FAE-CAE-LSTM method concatenate the all obtained features from three models (FAE, CAE, and POD) for nonlinear dimensionality reduction. Moreover, other works are mostly focused on exploring the capability of LSTM in modeling the flow in reduced order space for data sets with less randomness. Indeed, the data sets with less randomness have more memory in them, i.e., there are persistent or anti-persistent trends and thus, are more controllable through LSTM hyper parameters. Whereas, the performance of the proposed method in future state prediction of the flow was investigated by both convective unsteady flow problem (the Kelvin–Helmholtz instability), and periodic limit cycle states (flow past a square cylinder).

In the following, the second section describes POD and DMD methods. Then, in the third section, the deep autoencoder and LSTM networks are expressed and in the fourth section, proposed FAE-CAE-LSTM method for NIROM is given. The obtained results are presented in the fifth section and finally the conclusion is in the sixth section.

2 POD and DMD

Proper orthogonal decomposition (POD) is a widely used technique for feature reduction. POD tries to find a linear subspace smaller than the original feature space so that the new features have the most variance. POD can be used to data from experiments and numerical simulations. In numerical simulations, the numerical data are collected into a matrix, X, whose rows are individual snapshots relevant to the solution of the high-fidelity model for a particular set of parameters. For a problem with D spatial degrees of freedom and N snapshots, the snapshots' matrix will be:

$$X=\{x_1\cdot x_2\cdots x_N\}, \tag{2.1}$$

where $x_i\in\mathbb{R}^D$ is i-th snapshot of the flow field, and $X\in\mathbb{R}^{N\times D}$. POD projects the data onto a M-dimensional subspace ($M<D$). Therefore, POD method is applied to the X matrix in order to obtain the reduced features. According to POD, the projection is denoted as:

$$y=Ax,\quad \text{where}\quad A=\left[u_1^T,\cdots,u_m^T\right]\quad \text{and}\quad u_k^Tu_k=1\quad \text{for}\quad k=1,\cdots,m \tag{2.2}$$

The goal of POD is to maximize the variance of $\{y_i\}$, which is the trace of the covariance matrix of $\{y_i\}$. Thus:

$$A^*=\underset{A}{\operatorname{argmax}}\operatorname{tr}\left(S_y\right), \tag{2.3}$$

where

$$S_y = \frac{1}{N}\sum_{i=1}^{N}(y_i-\bar{y})(y_i-\bar{y})^T,$$
$$\bar{y} = \frac{1}{N}\sum_{i=1}^{N}x_i.$$

Let S_y be the covariance matrix of $\{y_i\}$. Since $\mathrm{tr}(S_y)=\mathrm{tr}(AS_xA^T)$ by using the Lagrangian multiplier and taking the derivative get:

$$S_x u_k = \lambda_k u_k, \tag{2.4}$$

which means that u_k is an eigenvector of S_x. Now x_i can be represented as:

$$x_i = \sum_{k=1}^{D}\left(x_i^T u_k\right)u_k \tag{2.5}$$

can be also approximated by:

$$\tilde{x_i} = \sum_{k=1}^{M}\left({x_i}^T u_k\right)u_k. \tag{2.6}$$

Where, u_k is the eigenvector of S_x corresponding to the kth largest eigenvalue.

In fluid dynamics, Dynamic Mode Decomposition (DMD) is one of the well-known modelfree reduced order modeling techniques and also a method to decompose complex flows into dominant spatiotemporal coherent structures using the power of the SVD. The first step for utilizing the DMD analysis is to construct the lagged matrix X' from the snapshots' matrix X:

$$X' = \{x_2 \cdot x_3 \cdot \cdots x_{N+1}\}. \tag{2.7}$$

The locally linear approximation can be formulated as follows in terms of X and X' matrices:

$$X' \approx AX. \tag{2.8}$$

As a result, the best fit operator A is provided by:

$$A = X'X^{\dagger}, \tag{2.9}$$

where † is the Moore-Penrose pseudoinverse. The high-dimensional matrix A is not calculated directly; rather, it is projected onto a low-rank subspace using SVD first. The SVD of X could be determined as follows:

$$X = U\sum V^*. \tag{2.10}$$

The matrix A is then derived from:

$$A = X'V\sum^{-1}U^* \tag{2.11}$$

As a result, $\tilde{A}$ matrix could be produced as follows:

$$\tilde{A} = U^*AU = U^*X'V\Sigma^{-1}, \tag{2.12}$$

where $\tilde{A}$ matrix is the $r \times r$ projection of the matrix A onto the low-ranked subspace r is the truncation rank. Eigen decomposition of A obtain a set of eigenvectors ω and eigenvalues λ, where:

$$\tilde{A}\omega = \lambda\omega. \tag{2.13}$$

A DMD mode can be defined as:

$$\phi = \frac{1}{\lambda}X'V\sum^{-1}\omega. \tag{2.14}$$

Finally, the approximate solution of at the future times could be determined using:

$$x(t) \approx \sum_{k=1}^{r} \phi_k e^{(\omega_k t)} b_k, \tag{2.15}$$

where, $\omega_k = \ln(\frac{\lambda_k}{\Delta t})$ and $b = \Phi^{\dagger}x_1$ is the initial amplitude of each mode. x_1 is the initial snapshot, and Φ is the matrix of DMD eigenvectors.

3 Deep neural networks

In the following subsections, we introduce deep neural network architectures for establishing NIROM. First, the two types of autoencoder used in this article are described: a standard, fully connected autoencoder (FAE) and a convolutional autoencoder (CAE). Then, a type of recurrent neural network called long short-term memory network (LSTM) is explained.

3.1 Fully connected autoencoder

Autoencoder is a type of artificial neural network that is used as a generative model, feature extractor, representation learning in unsupervised learning, and to pre-train discriminative multilayer neural networks in supervised learning. Autoencoder was first introduced by Bourlard [80]. An autoencoder is a neural network that implements two encoder $(x): R^n \to R^k$ and decoder $(x): R^k \to R^n$ mappings. The goal of autoencoder is to obtain the kdimensional representation of the data x so that the error measure between

x and decode $(\text{encode}(x))$ is minimized. Autoencoder is a great tool for Dimensionality Reduction and can be considered as a generalization of POD. An autoencoder without non-linear activations and only with code layer should be able to learn POD transformation in the encoder if trained to optimize mean squared error (MSE) loss.

A fully connected autoencoder is a special type of feed-forward neural network that is trained to learn the identity map. A bottleneck at its central layer forces the autoencoder to learn a reduced representation of the data. The simplest autoencoder is fully connected neural network that has an input layer with n neurons, a hidden layer with m neurons where $m < n$, and an output layer with n neurons. These models became widely used after deep learning. These networks are trained to create a new representation of input data. During training, it is expected that the data will be restored to the original data from this new representation. In other words, the goal is to include all the knowledge contained in the data in the resulting representation.

The number of visible layer neurons is equal to the number of input data. The number of hidden layer neurons can be greater or less than the number of visible neurons. If the hidden layer is larger than the visible layer, the model can simply copy all the properties to the hidden layer. If the number of hidden layer neurons is less, the autoencoder tries to find the best possible representation. For this purpose, various methods have been proposed to adjust the parameters of autoencoder. In denoising autoencoder (DAE) [81], noise is artificially added to the data. The loss function is defined in such a way that the model eliminates this noise. To do this, the autoencoder must learn the general structure of the data, and it is not possible to remove the noise just by memorizing the data. Sparse autoencoder (SAE) [82] may include more (rather than fewer) hidden units than inputs, but only a small number of the hidden units are allowed to be active at once. This sparsity constraint forces the model to respond to the unique statistical features of the input data used for training. Contractive autoencoder [83] adds an explicit regularizer in their objective function that forces the model to learn a function that is robust to slight variations of input values. This regularizer corresponds to the Frobenius norm of the Jacobian matrix of the encoder activations with respect to the input.

An autoencoder consists of two parts, the encoder and the decoder, which can be defined as transitions f and g such that:

$$\begin{cases} \boldsymbol{f}: \boldsymbol{X} \to \boldsymbol{Z}, \\ \boldsymbol{g}: \boldsymbol{X} \to \boldsymbol{X}, \\ g(\boldsymbol{f}(\boldsymbol{x})) = x. \end{cases} \tag{3.1}$$

The purpose of the autoencoder neural network is to reduce the difference between the input and output of the network. The following equation shows the loss function of the autoencoder neural network:

$$\arg\min \|(g \circ f)X - X\|_2^2. \tag{3.2}$$

By stacking autoencoder, models with more parameters can be built for more complex data. Training this stack will also be layer by layer using a greedy algorithm. Autoen-

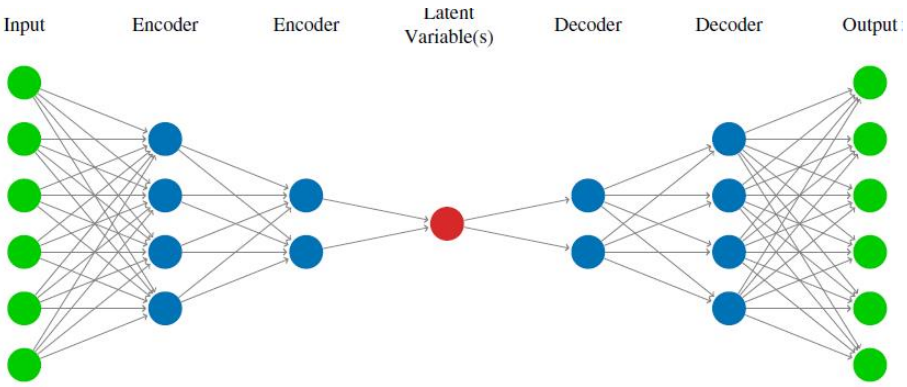

Figure 1: A fully connected autoencoder with six layers [62].

coders are often trained with only a single layer encoder and a single layer decoder, but using deep encoders and decoders offers many advantages. Depth can exponentially reduce the computational cost of representing some functions. Depth can exponentially decrease the amount of training data needed to learn some functions [83]. Experimentally, deep Autoencoders yield better compression compared to shallow or linear Autoencoders [84].

Fig. 1 shows an FAE with five hidden layers. Neurons can have connections with sending neurons from the previous layer and with receiving neurons from the following layer. In this network, the input and output layers have six neurons each and the latent space is in dimension one. The input to a neuron is calculated as a weighted sum of the outputs of the sending neurons to which it is connected and a bias term. The output of the neuron is the evaluation of the activation function for this weighted sum.

3.2 Convolutional autoencoder

Convolutional neural networks were first introduced as an alternative to fully connected networks for data structured as multiple arrays e.g., 1D signals and sequences, 2D images or spectrograms, and 3D video. Local connections, and shared weights are two key properties of convolutional neural networks [83,25]. In arrayed data, often local groups of values are highly correlated, assembling into distinct features that can be easily detected using a local approach. Additionally, weight sharing across the input domain works to detect location invariant features.

Convolutional Autoencoders (CAE) wrapping the fully-connected autoencoder with a convolutional neural network that have some convolutional layers. In CAE, rather than applying a fully-connected network directly to data, apply it to a vectorized feature map of a much lower-dimension obtained from a deep convolutional network acting directly on the high-dimensional data. In a convolutional layer, instead of learning a matrix that connects all m neurons of layer's input to all n neurons of the layer's output, learn a set

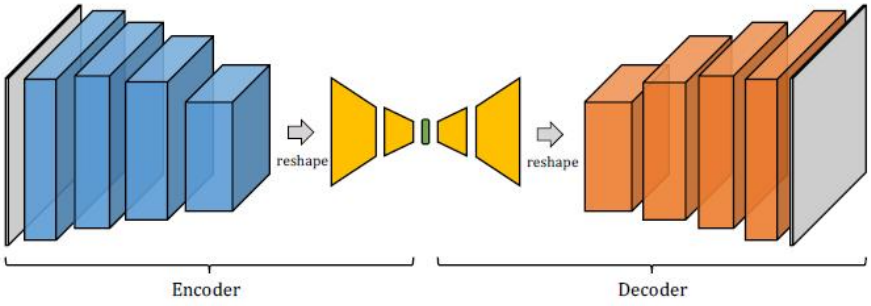

Figure 2: Network architecture of the convolutional autoencoder [40].

of filters. Each filter is convolved with patches of the layer's input. In CAEs, layers are organized into feature maps, where each unit in a feature map is connected to a local domain of the previous layer through a filter bank. Consider a $2D$ input $X \in \mathbb{R}^{N_x \times N_y}$, a convolutional layer consists of a set of F filters $K^f \in \mathbb{R}^{a \times b}$, $f = 1, \cdots, F$, each of which generates a feature map $Y^f \in \mathbb{R}^{N'_x \times N'_y}$ by a $2D$ discrete convolution:

$$Y^f = \sum_{k=0}^{a-1} \sum_{l=0}^{b-1} K^f_{a-k,b-l} X_{1+s(i-1)-k,1+s(j-1)}, \tag{3.3}$$

where $N'_x = 1 + \frac{N_x+a-2}{s}$, $N'_y = 1 + \frac{N_y+b-2}{s}$ and $s \geq 1$ is an integer value called the stride.

As before, the feature map can be passed through an element-wise nonlinear function. Typically, the dimension of the feature map is reduced by using a pooling layer, in which a single value is computed from small $a' \times b'$ patch of the feature map either by taking the maximum value or averaging.

Consider the following 12-layer CAE model showed in Fig. 2. A 2D arrayed input $X \in \mathbb{R}^{N_x \times N_y}$, with $N_x \times N_y = 128$, is first passed through 4-layer convolutional encoder. Each convolutional encoder layer uses a filter bank $K^f \in \mathbb{R}^{5 \times 5}$, with the first layer having a dilation rate of 2 and the number of filters f increasing from 4 in the first layer to 32 in the fourth layer using Eq. (2.8). At the opposite end of the convolutional autoencoder network use a 4-layer decoder network consisting of transpose convolutional layers. Often erroneously referred to as "deconvolutional" layers, transpose convolutional layers multiply each element of the input with a filter K^f and sum over the resulting feature map, effectively swapping the forward and backward passes of a regular convolutional layer.

3.3 Long short-term memory

THe recursive neural network can improve its performance by understanding the dependencies of the sequence. Memory generated from recursive connections can be constrained by algorithms for training recursive neural networks. The major challenge and

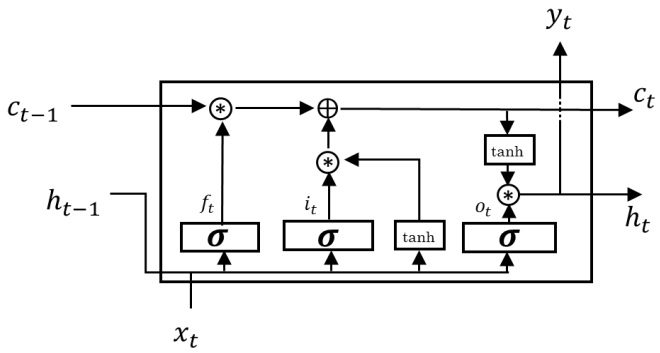

Figure 3: The LSTM cell [86].

disadvantage of standard recursive neural networks (RNN) is the inability to work with long sequence dependencies. Long short-term memory (LSTM) [85] is one of the most popular and common solutions to this challenge. The LSTM network is a specific type of recursive neural network capable of learning long-term dependencies. All recursive neural networks are repeated in the form of chains of modules. In RNNs, this iteration module usually has a simple structure. While in LSTM, the iteration module is more complex. Instead of having one layer of neural network, four layers interact. In addition, it has two states: hidden and cell. In the RNN structure, the range of content available in practice is very limited. The problem is that the impact of a given input on the hidden layer, and consequently on the network output, exponentially decreases and disappears. This problem is known as the vanishing gradient. The LSTM structure consists of a set of recursively connected subnets, called memory blocks. Each block contains one or more self-recursive memory cells and three units of multiplication, input, output, and forget gates, which provide continuous analogs of writing, reading, and rearranging cell functions. Fig. 3 is an example of a cell of this type of network.

The cell remembers values over arbitrary time intervals and the three gates regulate the flow of information into and out of the cell. Therefore, the problem of vanishing gradient is reduced.

Intuitively, the cell is responsible for keeping track of the dependencies between the elements in the input sequence. The input gate controls the extent to which a new value flows into the cell, the forget gate controls the extent to which a value remains in the cell and the output gate controls the extent to which the value in the cell is used to compute the output activation of the LSTM unit. The activation function of the LSTM gates is often the logistic sigmoid function. There are connections into and out of the LSTM gates, a few of which are recurrent. The weights of these connections, which need to be learned during training, determine how the gates operate.

In the equations below, the lowercase variables represent vectors. Matrices W_q and U_q contain, respectively, the weights of the input and recurrent connections, where the subscript q can either be the input gate i, output gate o, the forget gate f or the memory

cell c, depending on the activation being calculated. In this section, we are thus using a "vector notation". So, for example, $c_t \in \mathbb{R}^h$ is not just one cell of one LSTM unit, but contains h LSTM unit's cells. The compact forms of the equations for the forward pass of an LSTM unit are [87]:

$$f_t = \sigma_g\left(W_f x_t + U_f h_{t-1} + b_f\right), \quad i_t = \sigma_g(W_i x_t + U_i h_{t-1} + b_i), \tag{3.4a}$$

$$o_t = \sigma_g(W_o x_t + U_o h_{t-1} + b_o), \quad c_t = f_t \circ c_{t-1} + i_t \circ \sigma_c(W_c x_t + U_c h_{t-1} + b_c), \tag{3.4b}$$

$$h_t = o_t \circ \sigma_h(c_t). \tag{3.4c}$$

Where the initial values are $c_0 = 0$ and $h_0 = 0$ and the operator $\circ$ denotes the Hadamard product (element-wise product). The subscript t indexes the time step. $f_t \in \mathbb{R}^h$ is forget gate's activation vector, $i_t \in \mathbb{R}^h$ input/update gate's activation vector, o o $\in \mathbb{R}^h$ output gate's activation vector, $W \in \mathbb{R}^{h\times d}$. $W \in \mathbb{R}^{h\times h}$. $b \in \mathbb{R}^h$ are weight matrices and bias vector parameters which need to be learned during training where the superscripts d and h refer to the number of input features and number of hidden units, respectively. $x_t \in \mathbb{R}^h$ is input vector to the LSTM unit and $h_t \in \mathbb{R}^h$ hidden state vector also known as output vector of the LSTM unit. $c_t \in \mathbb{R}^h$ is cell state vector. σ_g is sigmoid function, σ_c. σ_h are hyperbolic tangent function.

4 FAE-CAE-LSTM method for ROM

In this study, the ability of deep autoencoder and LSTM networks in nonlinear dimension reduction and learning of the sequential data and the power of POD to identify the most important linear features of flow are leveraged for non-intrusive reduced order modeling of unsteady flows. In the proposed method, two types of autoencoder networks, namely fullyconnected autoencoder (FAE) and Convolutional autoencoder (CAE), have been used to take advantage of the capability of FAE in learn more complicated and nonlinear patterns and ability of CAE in exploit local correlations in field values simultaneously.

The essential and initial component of the proposed method requires a high-fidelity series of snapshots of the flow field which can be obtained by full-order simulations, experiments or field measurements. Our data-driven surrogate model shall rely on the use of CAE, FAE and POD for dimensionality reduction and LSTM for latent space temporal evolution of the state. Let $X = \left\{x_n, x_{n+1}, \cdots, x_{n+p}\right\} \in \mathbb{R}^{N\times M}$ be the flow field data set where $x_i \in \mathbb{R}^m$ is the flow field snapshot at time t_i and n is the number of snapshots. m is the number of data points, for example, number of probes in an experiment or field measurements or the number of mesh nodes in a numerical simulation? The target is to predict the future flow fields: $x_{n+1} \cdot x_{n+2}, \cdots$ using the data set X. The schematic of the proposed method, namely FAE-CAE-LSTM, is presented in Fig. 4. The following are the steps of the FAE-CAE-LSTM method:

1. The first step is to train FAE and SAE networks and perform POD to obtain the reduced features using the snapshot data collected from the flow-field with the aim of

nonlinear dimension reduction and feature extraction. In this study, the input and output data of the FAE and SAE networks and POD are the velocity magnitudes V and pressure magnitudes U on the computational grid points obtained from CFD simulation of the flow field over the test cases. For train FAE and perform POD, the flow field snapshot data $x_i \in \mathbb{R}^m$ fed to FAE and POD method. For train CAE, the flow field snapshot data $x_i \in \mathbb{R}^m$ is first converted to the two-dimensional $x_i \in \mathbb{R}^{m_x \times m_y}$ and then fed to the network.

2. Then, the features obtained from each of methods FAE, CAE and POD for each time step data x_i, which is a vector of state variables at time t, are mapped to the latent spaces through the functions as $g_{FAE}(t) \in \mathbb{R}^{k_{FAE}}$, $g_{CAE}(t) \in \mathbb{R}^{k_{CAE}}$ and $g_{PCA}(t) \in \mathbb{R}^{k_{PCA}}$ are, $k_{FAE}+k_{CAE}+k_{PCA}=k \ll M$. In each of the feature extraction methods (FAE, CAE and POD), features in latent spaces $(g_{FAE}(t), g_{CAE}(t), g_{PCA}(t))$ are projected back to the original space to reconstruct the input data $\tilde{x}_i$.

3. The data in the latent space, $g(t)=[g_{FAE}(t), g_{CAE}(t), g_{POD}(t)]$ include features from all models, is then used as the input to train the LSTM network with the aim of future prediction. $g(t)$ includes features from all models $(g(FAE), g(CAE), g(POD))$. The input of the LSTM network is a sequence of g as $\left[g(t_n), \cdots, g\left(t_{n+p}\right)\right]$ with the length of p and the output is the flow field data at the next time step of the input sequence $\tilde{x}_{n+p+1}$. Note that in this way, p previous time units are used for prediction of the next time step.

4. For testing and using the FAE-CAE-LSTM method for reduced order modeling, an iterative procedure is conducted. An initial sequence is first mapped to the latent space using the encoder parts of FAE and CAE and modes of POD. The data on the latent space is fed to the LSTM network to predict the flow field data for the next time step of the input sequence. The predicted value is then stacked to the input sequence ignoring the first snapshot data to prepare a sequence one step further in time. The new sequence is fed again to the network to predict the next time step, and the procedure is performed iteratively for future prediction.

The autoencoder and LSTM networks are trained by the feedforward and backpropagation algorithms. In the FAE and the CAE networks, use a stack version of autoencoders and learn a deep architecture with scaled exponential linear units (SELU) [88].The decoder architectures in both model are symmetrically encoders and thus the number of parameters is halved. It is very important for the activity function f in hidden layers of FAE to contain non-zero negative and unbounded positive parts, and for this reason SELU units are used. Fig. 5 shows the SELU activation function. The following is the formula for this activation function:

$$\text{selu}(x)=\begin{cases}\lambda x, & \text{if} \quad x>0,\\ \lambda\alpha(e^x-1), & \text{if} \quad x\leq 0.\end{cases} \tag{4.1}$$

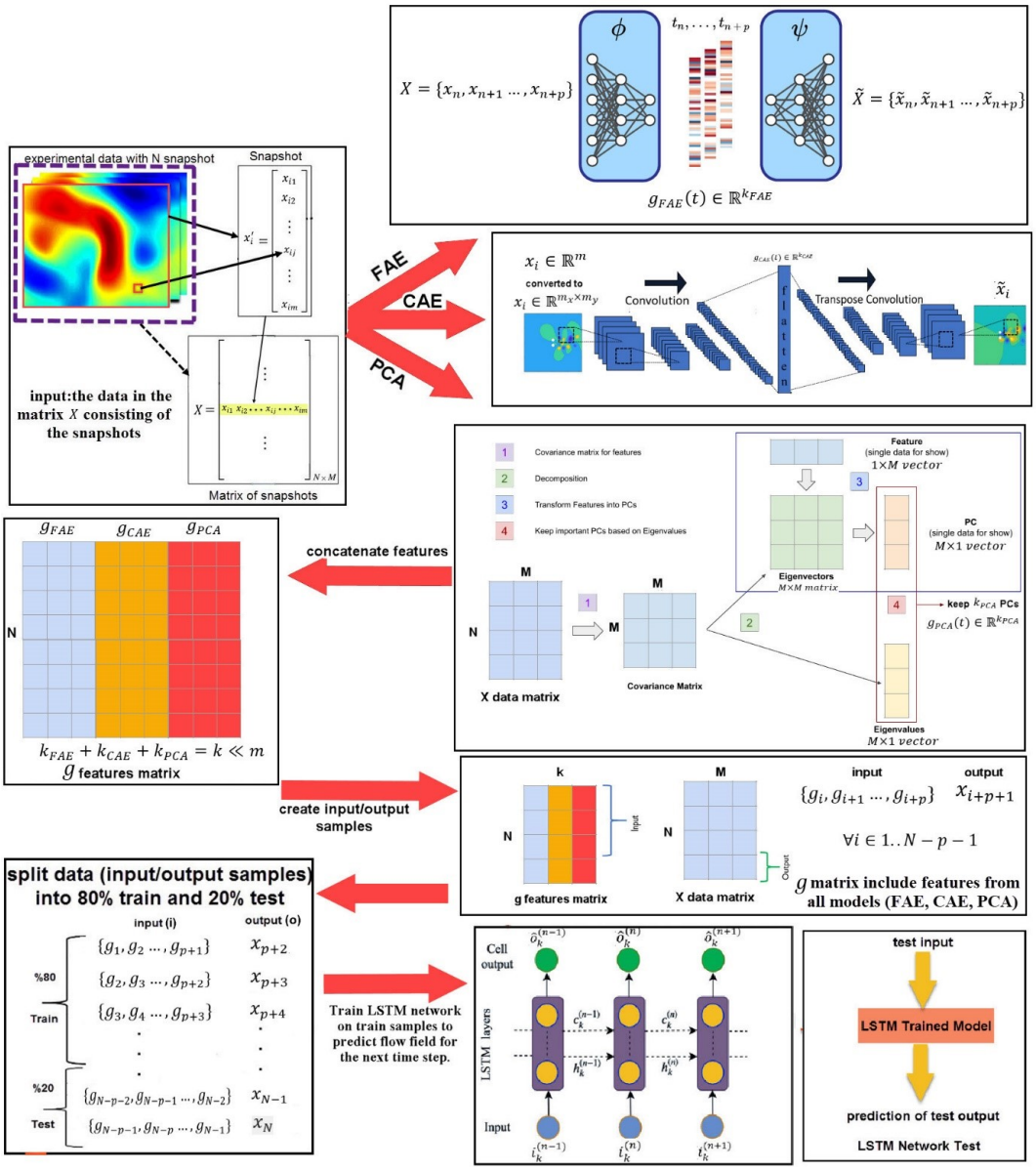

Figure 4: FAE-CAE-LSTM reduced order modeling framework.

If decoder mirrors encoder architecture as it does in the FAE model, then one can constrain decoder's weights W_d^l to be equal to transposed encoder weights W_e^l from the corresponding layer and has almost two times less free parameters.

If decoder mirrors encoder architecture as it does in the FAE model, then one can constrain decoder's weights W_d^l to be equal to transposed encoder weights W_e^l from the

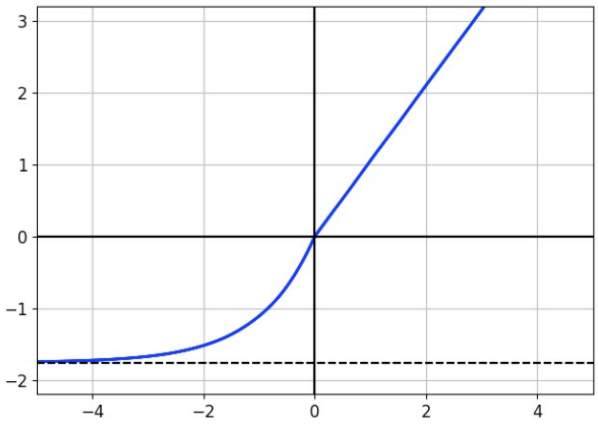

Figure 5: The SELU activation function.

corresponding layer and has almost two times less free parameters.

In the FAE-CAE-LSTM method, the root mean square error (RMSE) is used as the loss function for training of the networks:

$$RMSE = \left[\frac{1}{n}\sum_{i=1}^{n}(y_i - \tilde{y}_i)^2\right]^{1/2}. \tag{4.2}$$

Where y_i is the real data, $\tilde{y}_i$ represents the predicted data. In addition, n is the number of samples, which in this study is the number of points (grids) in the computational domain.

5 Results and discussion

In this work, we use two well-known canonical cases: 1) Kelvin-Helmholtz Instability, 2) flow past a cylinder. Both test cases are based on structured (Cartesian) grids. The specifications of the datasets used are given in Table 1. Python 3.7 was used to implement and execute the NIROM using the FAE-CAE-LSTM method in a Spyder environment on an 8GB computer with a 2.4GHz quad-core processor. For experiments, no pre-training has been used.

First, the performance of the FAE-CAE-LSTM method was investigated by solving a strong shear flow exhibiting the Kelvin–Helmholtz instability, known as the Marsigli flow

Table 1: Specifications of the datasets used.

Dataset name	# snapshots	grid resolution	time step (Δt)	Reynolds number
Cylinder	470	267×94	0.005	100
Kelvin-Helmholtz	46	800×100	0.0005	7000

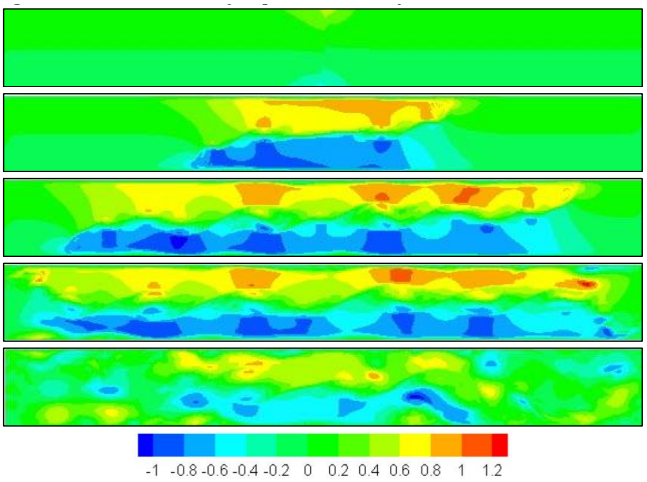

Figure 6: Evolution of the pressure variable in Kelvin-Helmholtz dataset.

or the lock-exchange problem. This flow problem is a convective unsteady flow problem without time periodicity. Thus, the lock-exchange flow is a challenging benchmark problem for constructing accurate ROMs. The physical process in the Marsigli flow problem explains how differences in density (i.e., using the Boussinesq approximation, the density difference can be shown to be inversely proportional to temperature difference) cause currents to form in the ocean and seas. When waters of two different densities meet, the higher density water will slide below the lower density water. These varying densities form a density current in the water. This is one of the primary mechanisms by which ocean currents are formed [89]. The evolution of the pressure and velocity variables in Kelvin-Helmholtz dataset are shown in Figs. 6 and 7. The numerical simulation of these convective rolling structures in flows that exhibit Kelvin–Helmholtz instabilities is quite challenging and requires high resolution to accurately capture the flow dynamics.

In the second numerical example, a flow past a cylinder was simulated. We consider a 2D flow around a square cylinder [90] which velocity inlet boundary condition is used at the domain inlet and Convection condition is applied at the outflow. Furthermore, the top and bottom boundaries are set to be far away from the cylinder with the slip condition to prevent the interaction between the cylinder surface and the boundaries. The size of the computational domain is $(L_x,L_y)=(24.0,20.0)$, and the cylinder center is located at $(x,y)=(5.5,9.5)$. The flow equations are solved on the Cartesian grid system with the grid spacing of $\Delta x=0.06$, $\Delta y=0.08$ and the time interval of flow field data is $\Delta t=0.005$. The number of grid points used for DNS is $(N_x,N_y)=(267,94)$ and 470 snapshots were obtained at regularly spaced time intervals for each of the velocity and pressure solution variables. The dynamics of the cylinder is considered the limit-cycle states. In Fig. 8, the coefficient lift versus nondimensional time is shown. The evolution of the pressure and velocity variables in Cylinder dataset is shown in Fig. 9.

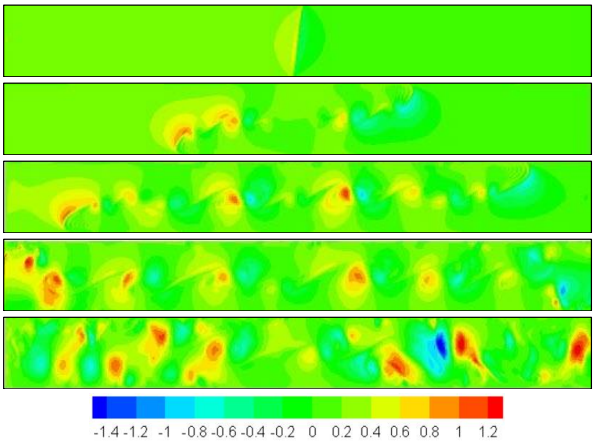

Figure 7: Evolution of the velocity variable in Kelvin-Helmholtz dataset.

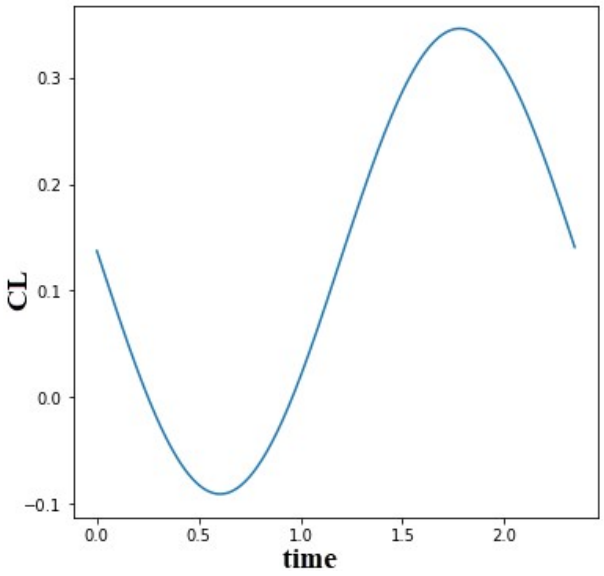

Figure 8: Temporal evolution of coefficient lift (CL) at cylinder dataset.

The CFD snapshots for the flow past a square cylinder test case were generated using a two-dimensional incompressible Navier-Stokes solver based on a finite volume method. We show the comparison against reference (i.e., Strouhal number) and mean Lift Coefficient with Re=100 for the square cylinder in Table 2 to justify the soundness of the simulation.

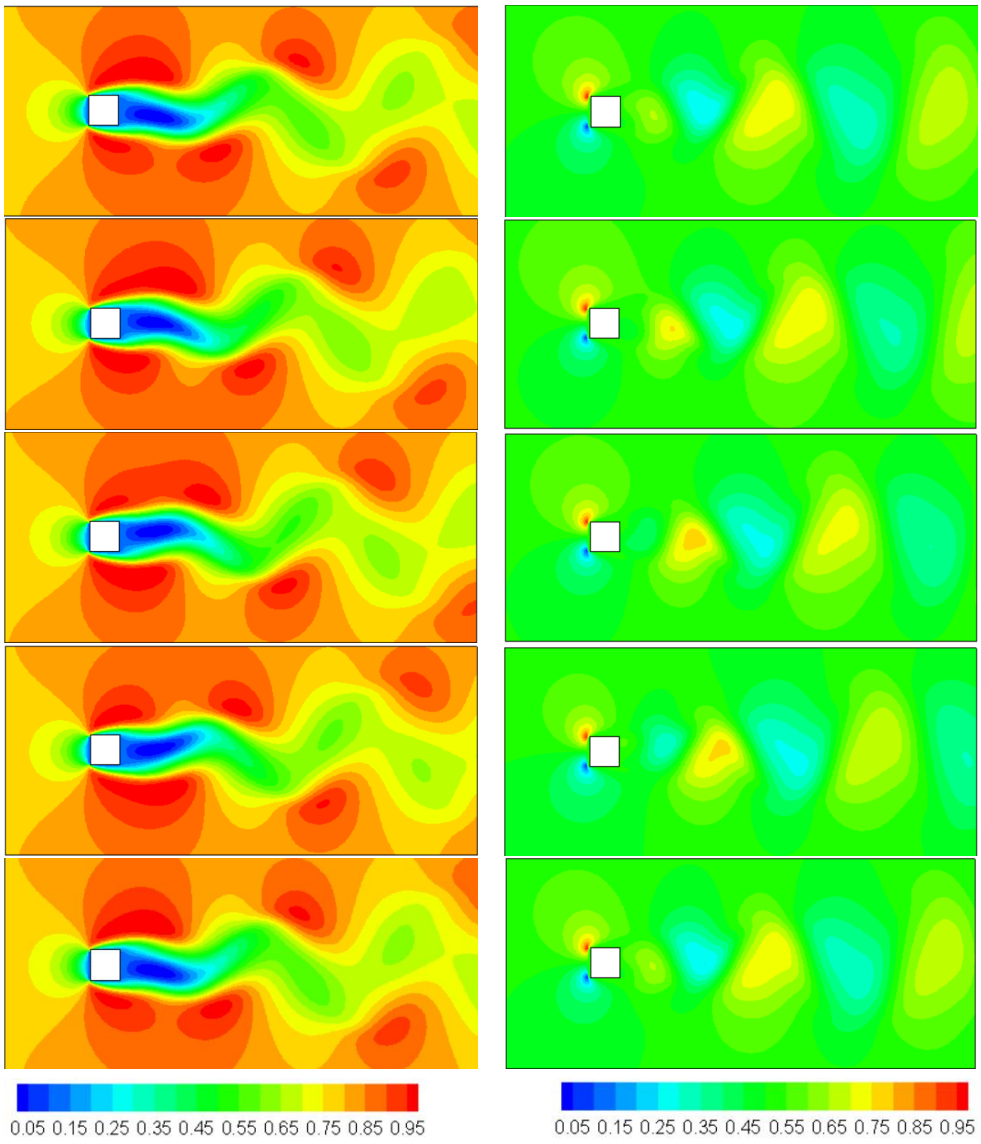

Figure 9: Evolution of the pressure and velocity variables in cylinder dataset.

5.1 Hyperparameter analysis

The number of layers, the number of neurons in the hidden layers and activation functions of the DAE network and the number and size of the filters in each convolution layer, the size of the stride, and the size of the filters in the pooling layer of CAE network and number of cells of LSTM network are the hyperparameters of the FAE-CAE-LSTM

Table 2: The comparison against reference and mean lift coefficient with $\mathrm{Re}=100$ for square cylinder.

existing literatures	Strouhal number	Mean Lift Coefficient
Sohankar et al. [91]	0.146	0.15
Cheng et al. [92]	0.144	0.15
Berrone et al. [93]	0.145	0.17
Lam et al. [94]	0.141	0.18
Our simulation	0.141	0.18

method. An analysis of the hyperparameters is conducted, and the results for the most important ones are presented for the velocity field of the cylinder dataset. Here, the performance of the FAE, CAE and LSTM networks in the reconstruction and prediction of the testing data is investigated using some of the most popular choices in deep learning: sigmoid, rectified linear units (RELU), hyperbolic tangent (TANH), exponential linear units (ELU), leaky relu (LRELU), and scaled exponential linear units (SELU) on the three hidden layer with 128 nodes in each hidden layer, five different number of hidden layers, $N_h \in [1,2,3,4,5]$ for the FAE network, three different number of filters $N_f \in [10,20,30]$ in convolution layers with size 3×3, three different size of filters $S_f \in [3\times3,5\times5,7\times7]$ in four convolution layers with 30 filters in the CAE network, and four different numbers of LSTM cells $N_{LSTM} \in [10,100,300,500]$ in the LSTM network. Note that the error for the DAE and CAE networks are the error in the reconstruction of the flow field from its dominant features, and the error for the LSTM is the prediction error. As authors reported in [95], a checkerboard artifact can be seen due to the unsuccessful training by using the transposed convolution with the large filter size. They demonstrated that the L2 error norm of the models with up-sampling operation gradually decreases as the filter size increases. In contrast, the model error with the transposed convolutional operation is reported significantly worse with $S_f > 9$. We note that the proposed model with the transposed convolution has less size of filters, i.e., $S_f \leq 7$.

Figs. 10(a) and (b) show the validation loss versus number of epochs for the FAE and the LSTM networks, respectively. It can be seen that the SELU activation function performs better for the FAE and the LSTM networks. Here, results are reported for the FAE network with three hidden layers and $k_{FAE}=50$. The first and third hidden layers contain 500 neurons. Note that we use linear activation functions for the Input, output, and the bottleneck layers and nonlinear activation functions for other layers. There are two properties which seems to separate activations which perform well from those which do not: a) non-zero negative part and b) unbounded positive part. Hence, we conclude, that in this setting these properties are important for successful training. Thus, we use SELU activation units and tune SELU-based networks for performance. Moreover, Fig. 10(c) shows the results for the LSTM networks with various numbers of LSTM cells. It can be seen that the increase in the numbers of LSTM cells leads to better predictions. Also, the effect of the number of hidden layers of the FAE on the accuracy of the FAE reconstructions is investigated, and the results are reported in Fig. 10(d). It can be observed

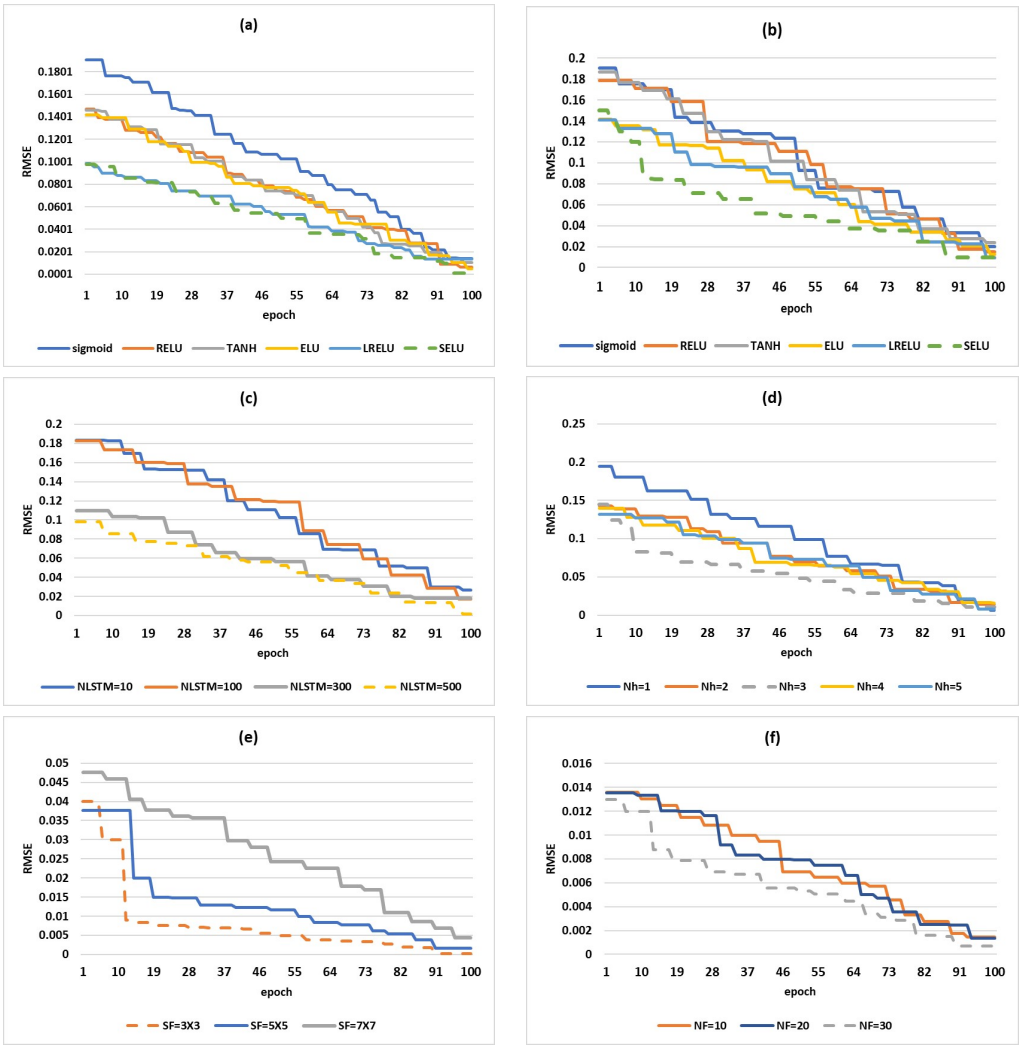

Figure 10: Results of the hyperparameter analysis; (a) validation loss versus epochs of the reconstruction of the test data set for SAE network with different activation functions, (b) validation loss versus epochs of the prediction of the test data set for LSTM network with different activation functions, (c) validation loss versus epochs of the prediction of the test data set for LSTM network with different numbers of LSTM cells, (d) effect of the number of hidden layers of the SAE network on the predictions with FAE-CAE-LSTM, (e) validation loss versus epochs of the reconstruction of the test data set for CAE network with different size of filters, (f) validation loss versus epochs of the reconstruction of the test data set for CAE network with different number of filters.

that the FAE with three hidden layers obtains the best results. It also should be noted that to avoid overfitting, the training process is stopped where more training epochs do not lead to further reduction of the validation loss. While making layers wider helps bring training loss down, adding more layers is often correlated with a network's abil-

ity to generalize. Going from one layer in the encoder and decoder to three layers in both provides a good improvement in validation RMSE. After that, blindly adding more layers does help, however it provides diminishing returns. Figs. 10(e) and (f) show the validation loss versus number of epochs of the reconstruction of the test data set for the CEA with different size of filters and with various numbers of filters, respectively. Based on the comparison Fig. 9(e), we can conclude that smaller filter sizes are and should be a popular choice over larger sizes. From Fig. 9(f) results, it can be seen that the increase in the number of filters leads to better reconstructions.

Based on these results, the FAE network with three hidden layers and SELU activation function and the CAE network with 30 filters 3×3 in four convolution layers and the LSTM network with one hidden layer containing 500 LSTM cells with SELU activation function are chosen as the network architectures for the FAE-CAE-LSTM method. We also observed that these hyperparameters are robust for all the test cases, showing the possible applicability of the SELU activation function, for dimension reduction and future prediction in the domain of fluid dynamics.

5.2 Reconstruction comparison

Here, the performance of the proposed NIROM framework, FAE-CAE-LSTM, in the reconstruction of the flow-field from its extracted features of the test cases is compared with the DMD, POD, FAE and CAE methods. To evaluate the similarity between the flow field data set (X, snapshot data) and the reconstructed data ($\tilde{X}$) the root mean square error (i.e., RMSE) was calculated in this study. Data loss is an inevitable consequence of dimension reduction, and the amount of compression which leads to a reasonable approximation error in the reconstructed data is significant. Therefore, the number of neurons at the bottleneck layer of the autoencoder networks, or the number of dominant modes of the POD/DMD methods,k, is an important hyperparameter and should be chosen based on thorough experiments. Since DMD relies on singular value decomposition, we can specify the number of the largest singular values used to approximate the input data where the dominant DMD modes are the low-rank structures individuated. Here, methods with the size of the latent space (extracted features) of 5, 10, 15, 20, and 25 are trained and tested with the training and testing data set of the Kelvin-Helmholtz dataset and with the size of the latent space of 10, 20, 30,50, and 100 are trained and tested with the training and testing data set of Cylinder dataset. For all the models, the CAE and FAE network parameters are set as in Table 3.

RMSE of the testing data is shown in Fig. 11. As shown in Figs. 11(a)–(d), the RMSE of each method follows the order of Proposed<CAE<FAE<POD<DMD. Compared with AE models, the POD/DMD models in this study show larger RMSE on evaluation. Comparing the DMD with the POD method, the RMSE of these methods implies that the capabilities of POD/DMD models with similar model size are almost identical. By contrast, the CAE model improves the RMSE in the reconstruction of flow field. The improvement caused by CAE models demonstrates the superiority of convolutional neural networks

Table 3: CAE and FAE network parameters.

	#Hidden Layers	Batch size	Learning Rate	#Epochs	Activation function	#Nerouns per layer
FAE	3	32	0.0001	50	SELU	128
	# Hidden Layers	Batch size	Learning Rate	#Epochs	Flter size	#Filter per layer
CAE	4	32	0.0001	50	3×3	30

in spatial pattern generalizations. We find that the ability of nonlinear methods to reproduce snapshots is generally better than that of linear methods. In terms of nonlinear methods, CAE is better than the FAE autoencoder. As it is expected, an increase in the size of the extracted features leads to an increase in the accuracy of the reconstruction and a decrease in the approximation error. Since the AE needs to discard the information while extracting dominant features from high-dimensional flow fields into a limited number of latent spaces. But the decrease of RMSE slows up with extracted features grow. It can be seen that even with $k=10$, the data has been reconstructed with acceptable accuracy by FAE-CAE-LSTM. Considering that models with smaller hidden dimensions require less training data and computing resources. It was clearly seen that the mapping ability of methods highly depend on the target flows, i.e., complexity. Although the strength of the AEs could be seen through the investigation, one of issues of the present form of the AEs is the interpretability of latent space. Since the AE-based modes are not orthogonal with each other, it is still hard to understand the role of each latent vector for reconstruction.

At the latent space, the proposed NIROM framework, FAE-CAE-LSTM, extracts the main features of the flow-field with concatenate the features obtained from each of methods FAE, CAE and POD, which is adequate for the reconstruction of the flow field. To visualize the main features of the flow for the aforementioned test cases, an input array of g is given to the latent space, and the results are taken from the the original space. Figs. 12 and 13 show the reconstruction contours of the velocity and pressure magnitudes with 15 and 30 features obtained by different methods for a snapshot over Kelvin-Helmholtz and Cylinder datasets, respectively (22th snapshot of Kelvin-Helmholtz and 235th snapshot of Cylinder).

It has recently been reported that performing POD for the decoded flow fields by autoencoders tells us how much POD modal structures are contained in the nonlinear autoencoder modes [96,97,98]. To this end, we predict the whole of time span of each test case using models with the latent dimension of five (POD rank truncation of five for the POD-RNN [68] and five neurons at the bottleneck layer for the autoencoder-LSTM [55], and the proposed FAE-CAE-LSTM method) and obtain the predicted velocity snapshots. We then perform a POD of the predicted snapshots to see how many POD bases are captured in the predictions. Figs. 14 and 15 present the results of the decomposition for the predicted and the real data for both Kelvin-Helmholtz and Cylinder datasets, respectively. For the POD-RNN method, the predictions based on five modes only contain

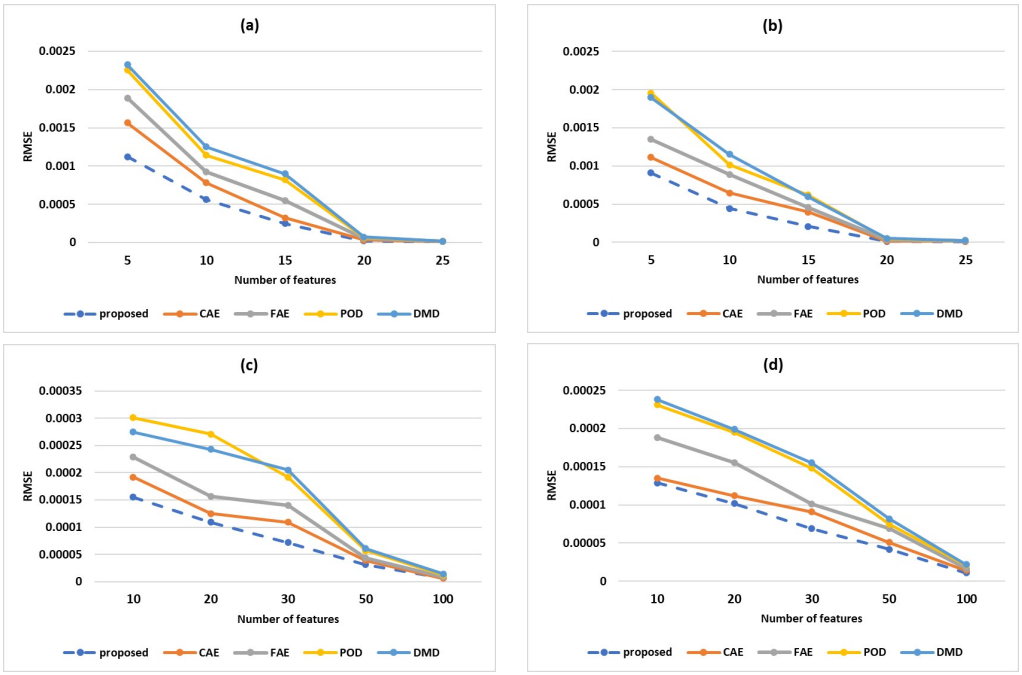

Figure 11: Results of the relationship between the number of latent space (features) and RMSE; (a) velocity magnitudes V of Kelvin-Helmholtz dataset, (b) pressure magnitudes U of Kelvin Helmholtz dataset, (c) velocity magnitudes V of Cylinder dataset, (d) pressure magnitudes U of cylinder dataset.

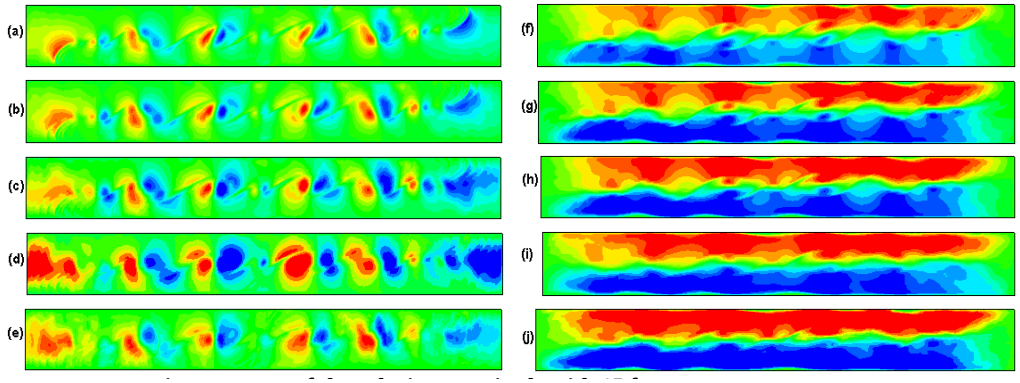

Figure 12: The reconstruction contours of the velocity magnitude with 15 features obtained by (a) FAE-CAE-LSTM, (b) CAE, (c) FAE, (d) POD, (e) DMD methods and the pressure magnitude with 30 features obtained by (f) FAE-CAE-LSTM, (g) CAE, (h) FAE, (i) POD, (j) DMD methods for 22th snapshot of Kelvin-Helmholtz.

five POD basis as is expected. These modes are the same as the ones for the real data, so they are not depicted in Figs. 14 and 15. For the autoencoder-LSTM and proposed FAE-CAE-LSTM method, the interesting result is that the predictions based on only five

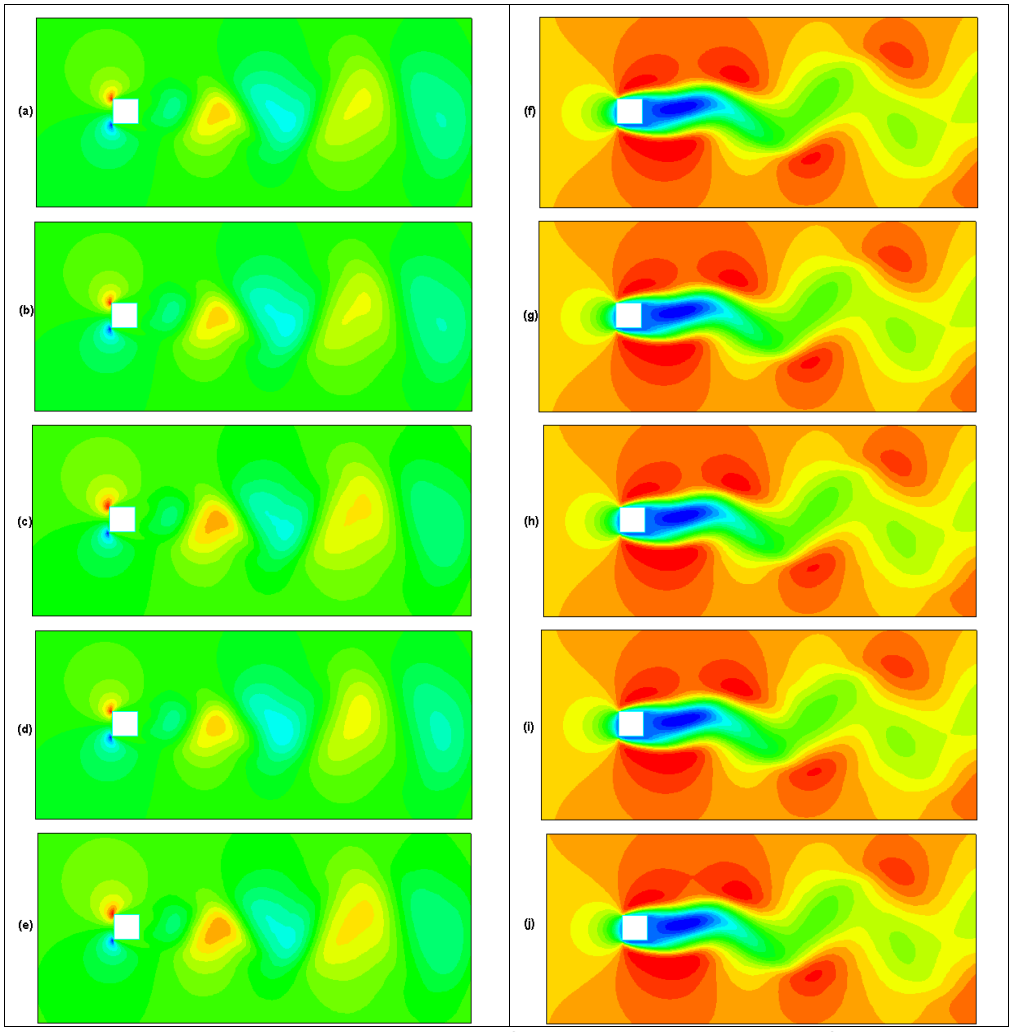

Figure 13: The reconstruction contours of the velocity magnitude with 30 features obtained by (a) FAE-CAE-LSTM, (b) CAE, (c) FAE, (d) POD, (e) DMD methods and the pressure magnitude with 30 features obtained by (f) FAE-CAE-LSTM, (g) CAE, (h) FAE, (i) POD, (j) DMD methods for 235th snapshot of cylinder.

modes contain more than five POD bases. As it can be observed for the autoencoder-LSTM method in Fig. 14 for up to mode six and in Fig. 15 for up to mode five and for the proposed FAE-CAE-LSTM method in Fig. 14 for up to mode eight and in Fig. 15 for up to mode six. In reality, the autoencoder may compress a greater number of POD modes into only five modes. Furthermore, the energy of the modes is approximate to that of the original data, particularly for the higher energy modes.

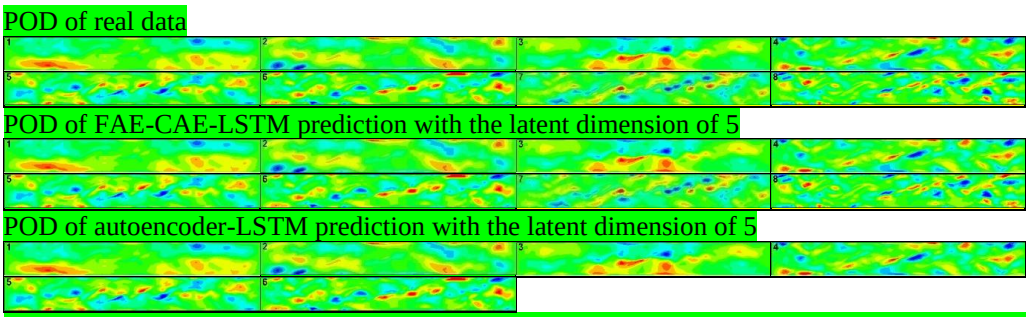

Figure 14: POD of the predicted velocity fields by autoencoder-LSTM and FAE-CAE-LSTM methods and the original data over Kelvin-Helmholtz dataset.

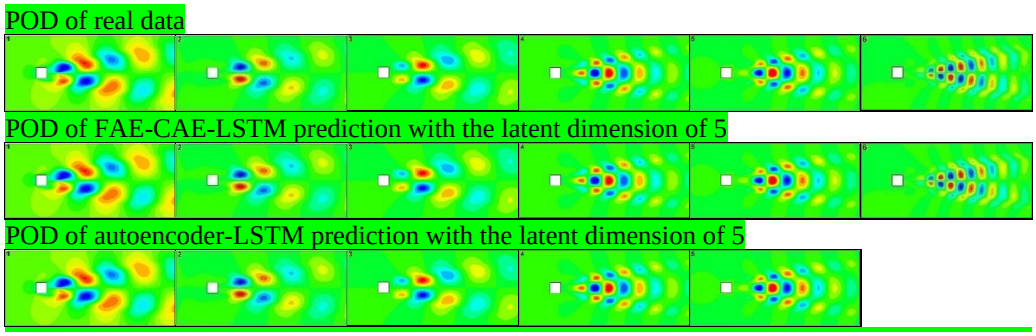

Figure 15: POD of the predicted velocity fields by autoencoder-LSTM and FAE-CAE-LSTM methods and the original data over cylinder dataset.

5.3 Prediction performance

In this section, the performance of the proposed NIROM method, FAE-CAE-LSTM, in the prediction of the future estate of the test cases (Kelvin-Helmholtz Instability, and flow past a cylinder) is compared with other NIROM methods such as CAE-LSTM [58], autoencoder-LSTM [55], autoencoder-DMD [55] and POD-RNN [68] based models. Results are reported for the train and test of Kelvin-Helmholtz Instability, and flow past a cylinder data sets, where the train data set is reconstructed and the test data set is predicted by the models. However, the main focus is on the comparison of the prediction skills of the models. For all the models, the size of the latent space is equal to 15 for the Kelvin-Helmholtz dataset and 30 for the cylinder dataset and the CAE, FAE, LSTM network parameters are set as in Tables 3 and 4, respectively.

RMSE obtained from the models in the prediction of the training and testing data sets are reported in Table 5.

As can be seen in Table 4, the FAE-CAE-LSTM method has achieved lower RMSE than the other methods in both datasets, which indicates the excellent performance of this method in the prediction of the future state of the ow, only from past measurements.

Table 4: LSTM network parameters.

batch-size	Number of Hidden Layers	Number of Cells	Learning Rate	epochs	Input size	output-size
32	1	100	0.0005	100	5	5

Table 5: RMSE of the examined models for the prediction of the testing and training datasets.

	dataset	Kelvin-Helmholtz		Cylinder	
Method		pressure	velocity	pressure	velocity
FAE-CAE-LSTM	train	0.000211	0.000249	0.000069	0.000072
	test	0.000392	0.000384	0.000085	0.000096
CAE-LSTM	train	0.000399	0.000325	0.000091	0.000109
	test	0.000541	0.000601	0.000114	0.000152
AE-LSTM	train	0.000458	0.000552	0.000101	0.000140
	test	0.000598	0.000808	0.000146	0.000186
AE-DMD	train	0.000596	0.000899	0.000155	0.000205
	test	0.000754	0.001057	0.000190	0.000249
POD-RNN	train	0.000622	0.000821	0.000148	0.000192
	test	0.000825	0.001004	0.000179	0.000227

From these results, it can be seen that since AE-DMD and POD-RNN do not use LSTM for the prediction of the future estate, these perform worse than the other methods in both datasets. Whereas, FAE-CAE-LSTM, CAE-LSTM, and AE-LSTM methods, in addition to using LSTM, also use Autoencoder to obtain the reduced features, achieve better results than AE-DMD and POD-RNN. It can be concluded that it is necessary to consider the LSTM network in the prediction of the transient dynamics of a low-order model and capability of AE in learn more complicated patterns. On the other hand, the results of Convolutional-based methods (FAE-CAE-LSTM and CAE-LSTM) are also better than the other methods which can be concluded that the efficiency of the dimensionality reduction process with Convolutional AE network also improves prediction of the future estate. The proposed method (FAE-CAE-LSTM), not only takes the advantages of all the above approaches but also employs POD technique.

To illustrate the performance of the models in the prediction of the flow field, contours of velocity and pressure magnitudes at the testing data for last snapshot over Kelvin-Helmholtz and Cylinder datasets are shown in Figs. 16 and 17 (46th snapshot of Kelvin-Helmholtz and 470th snapshot of Cylinder). It can be seen that the FAE-CAE-LSTM method, can acquire trustworthy results in the prediction of the time evolution of a complex fluid system.

To depict the performance of CAE-LSTM and FAE-CAE-LSTM models in the prediction of velocity and pressure evolution through time, we selected a point in each dataset that has the most changes during all snapshots and predictions of CAE-LSTM and FAE-CAE-LSTM models against the real data are presented in Fig. 18. It can be seen that for Cylinder test case, which is periodic dynamical system, the proposed FAE-CAE-LSTM

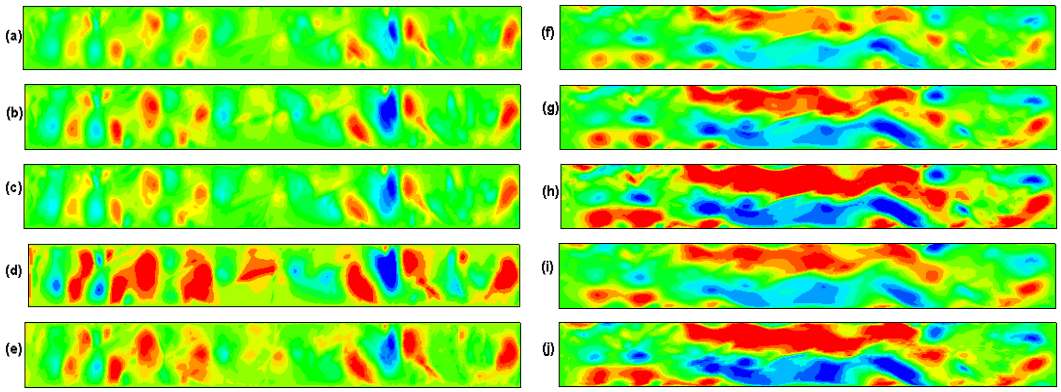

Figure 16: The prediction contours of the velocity magnitude with 15 features obtained by (a) FAE-CAE-LSTM, (b) CAE-LSTM, (c) autoencoder-LSTM, (d) autoencoder-DMD, (e) POD-RNN methods and the pressure magnitude with 15 features obtained by (f) FAE-CAE-LSTM, (g) CAE-LSTM, (h) autoencoder-LSTM, (i) autoencoder-DMD, (j) POD-RNN methods for 46th snapshot of Kelvin-Helmholtz.

model provides accurate results and for Kelvin Helmholtz test case, the FAE-CAE-LSTM method performs much better than the CAE-LSTM method in prediction of velocity and pressure magnitudes. Furthermore, the time history in the latent space obtained by FAE-CAE-LSTM for real and the reconstructed flow fields over Kelvin-Helmholtz and Cylinder datasets are shown in Fig. 19.

5.4 Time cost comparison

Here, we investigate the computational efficiency of the proposed NIROM method, FAE-CAE-LSTM, and compare it with CAE-LSTM [58], autoencoder-LSTM [55], autoencoder-DMD [55] and POD-RNN [68] based models. The essential time cost of the proposed NIROM is the online computation required for the training of the CAE, FAE, and LSTM neural networks. In Table 6, we report the needed time for both training (online) and prediction (offline) computations for each method. We note that the computing time assessments documented in this table are based on Python executions and for the experiments, the CAE, FAE, and LSTM network parameters are set as in Tables 3 and 4, respectively. The size of the latent space is equal to 15 for the Kelvin-Helmholtz dataset and 30 for Cylinder datasets. As can be seen in the table, FAE-CAE-LSTM is the slowest. However, we can observe that the computation time of FAE-CAE-LSTM is acceptable and comparable and the performance of FAE-CAE-LSTM implementation is better than other approaches. On other hand, with the advancement of hardware computing power, this amount of time can be saved by using more powerful hardware.

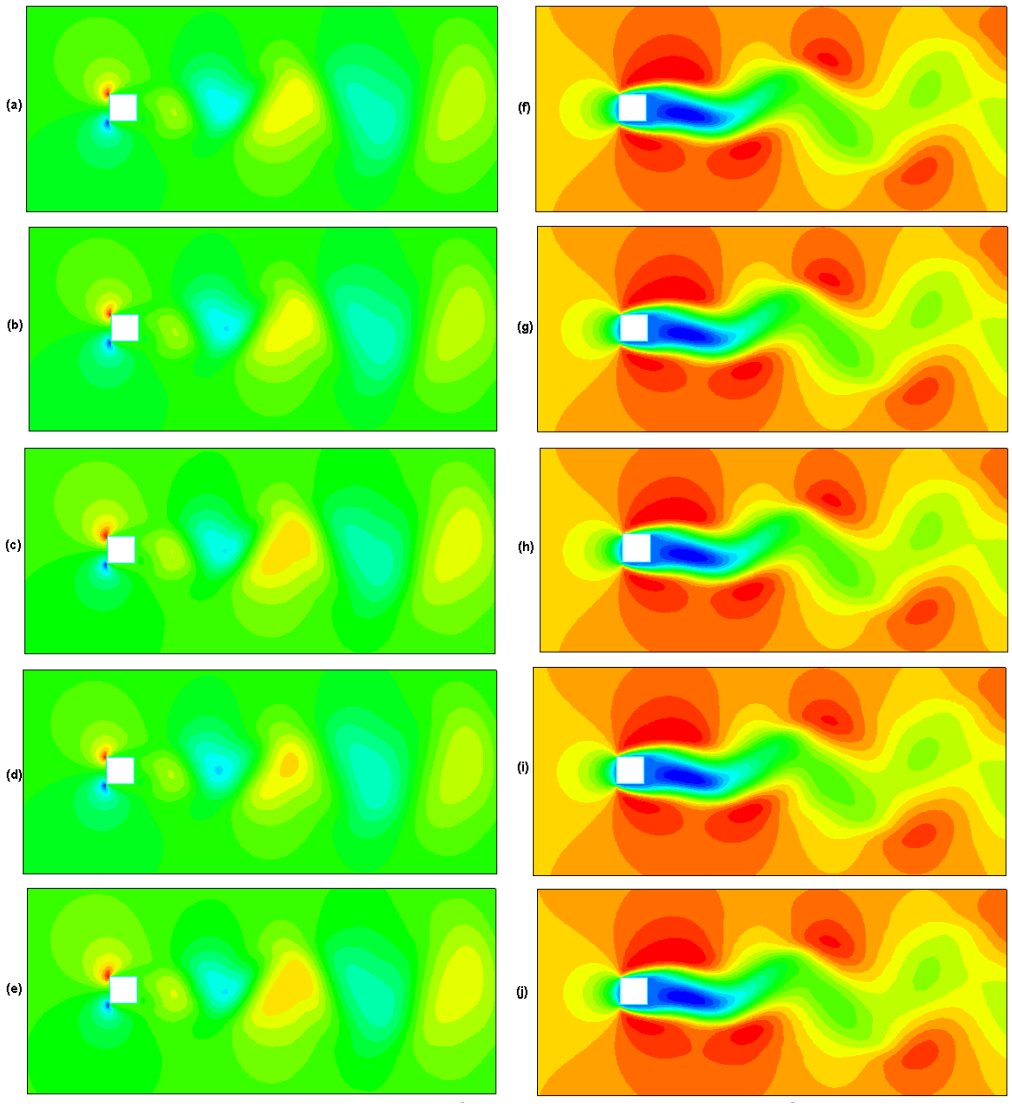

Figure 17: The prediction contours of the velocity magnitude with 30 features obtained by (a) FAE-CAE-LSTM, (b) CAE-LSTM, (c) autoencoder-LSTM, (d) autoencoder-DMD, (e) POD-RNN methods and the pressure magnitude with 30 features obtained by (f) FAE-CAE-LSTM, (g) CAE-LSTM, (h) autoencoder-LSTM, (i) autoencoder-DMD, (j) POD-RNN methods for 470th snapshot of Cylinder.

6 Conclusions

The global approximation properties of neural networks have been studied in many recent studies, which show that deep networks are powerful and comprehensive in approximating a wide class of functions. According to studies and research, deep learning

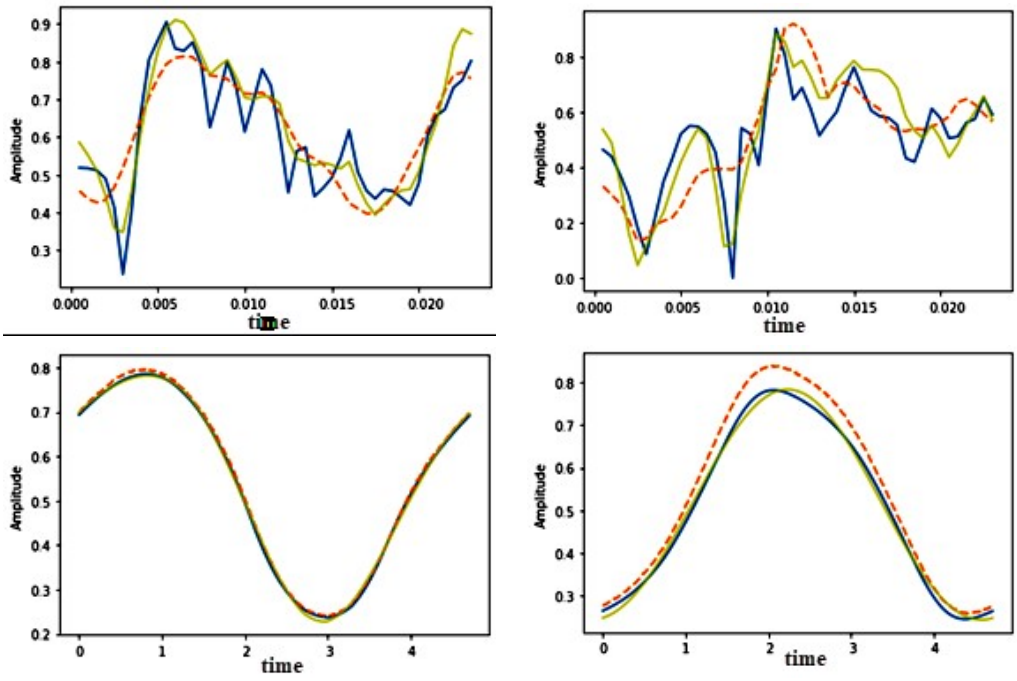

Figure 18: Comparison between time history of the values predictions of CAE-LSTM and FAE-CAE-LSTM models against the real data for a) pressure magnitude in Kelvin-Helmholtz, b) velocity magnitude in Kelvin-Helmholtz, c) pressure magnitude in Cylinder, d) velocity magnitude in Cylinder Dataset at a point with the most changes. Blue lines represent the results for the real data, and red and green shows the results obtained from the predictions of the FAE-CAE-LSTM and CAE-LSTM methods, respectively.

has been able to solve many challenges in the field of model order reduction. This approach has been able to overcome the challenge of nonlinear problems and provide the appropriate accuracy and speed to reduce order model. Unlike conventional neural networks, deep learning networks have the ability to work with labeled and unlabeled data, and because they can also automatically extract features, they have the ability to do so.

In this paper, a data-driven NIROM method based on the ability of the autoencoder neural network in dimensionality reduction and feature extraction and the ability of the LSTM network in the prediction of the sequential data is presented. The proposed method trains FAE and SAE networks and performs POD to obtain the reduced features. Then, it concatenates the all obtained feature and predicts future state of a nonlinear dynamical system using LSTM network from it's the features obtained. The proposed method was compared with other NIROM methods i.e., CAE-LSTM, autoencoder-LSTM, autoencoder-DMD, and POD-RNN based models and the results showed that the proposed method with a lower number of modes, offers higher accuracy in input reconstruction and the prediction of the flow field in future time instances.

There are a number of natural extensions and future directions suggested by this

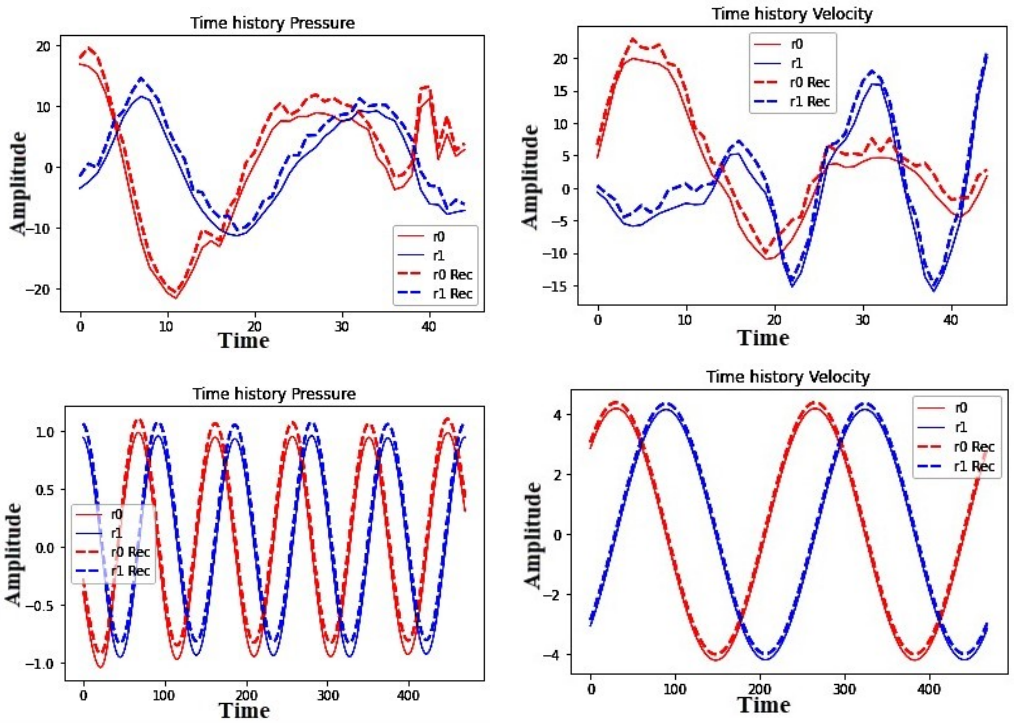

Figure 19: Time history of the latent vector obtained by FAE-CAE-LSTM for real and the reconstructed flow fields at a) pressure magnitude in Kelvin-Helmholtz, b) velocity magnitude in Kelvin-Helmholtz, c) pressure magnitude in Cylinder, d) velocity magnitude in cylinder. The latent vector obtained by FAE-CAE-LSTM for real data is shown by $r=(r_0,r_1)$ and for reconstructed flow fields is shown by $Rec=(r_0Rec,r_1Rec)$.

work. Three-dimensional flows and experimental observations would put the approach to the test and may reveal weaknesses that need to be addressed further. Although the present work only considered input from velocity and pressure fields, incorporating information and parameters of fluid flow e.g., angle of attack, Reynolds number, the Mach number, the diameter ratio of two cylinders, distance between two cylinders, and bluff body shapes is the subject of ongoing research. Applications of CNN to unstructured mesh fluid flow data are one of the remaining challenges of the present work. For many fluid flow studies, such as flows around an airfoil at practical Reynolds values, we frequently face unstructured data. Unfortunately, the standard CNN utilized in the proposed method cannot be directly applied to these unstructured mesh data. Several approaches, such as PhyGeoNet [99], graph CNN [100], Voronoi-tessellation aided [101] and, PointNet [102] have recently been presented to address this issue. One of the drawbacks in the current NIROM creation toward applications to more complicated flows, such as higher Reynolds number turbulence, is the requirement for a large number of latent modes. To address this issue, Fukami et al. [97] have created a hierarchical autoencoder, which can accomplish more efficient low-dimensionalization than traditional AEs

Table 6: Computing time (in seconds) required for training and prediction computations of FAE-CAE-LSTM, CAE-LSTM, autoencoder-LSTM, autoencoderDMD, and POD-RNN methods.

Datasets	Kelvin-Helmholtz		Cylinder	
Methods	Training time	Prediction time	Training time	Prediction time
FAE-CAE-LSTM	298.53	62.45	596.82	91.52
CAE-LSTM	172.08	47.53	407.19	77.16
Autoencoder-LSTM	153.76	34.21	381.92	70.85
Autoencoder-DMD	86.44	24.63	240.19	54.43
POD-RNN	59.71	20.49	209.98	47.19

and POD. Although their experiment was with a two-dimensional sectional velocity of turbulent flow, this approach could be integrated with the proposed NIROM technique.

References

[1] G. DIMITRIU, R. ŞTEFĂNESCU, AND I. M. NAVON, *Comparative numerical analysis using reduced-order modeling strategies for nonlinear large-scale systems,* Journal of Computational and Applied Mathematics, 310 (2017), pp. 32–43.

[2] Z. LIN, D. XIAO, F. FANG, C. PAIN, AND I. M. NAVON, *Non-intrusive reduced order modelling with least squares fitting on a sparse grid,* International Journal for Numerical Methods in Fluids, 83 (2017), pp. 291–306.

[3] F. FANG, C. PAIN, I. M. NAVON, AND D. XIAO, *An efficient goal-based reduced order model approach for targeted adaptive observations,* International Journal for Numerical Methods in Fluids, 83 (2017), pp. 263–275.

[4] F. FANG, T. ZHANG, D. PAVLIDIS, C. PAIN, A. BUCHAN, AND I. NAVON, *Reduced order modelling of an unstructured mesh air pollution model and application in 2D/3D urban street canyons,* Atmospheric Environment, 96 (2014), pp. 96–106.

[5] M. DIEZ, E. F. CAMPANA, AND F. STERN, *Design-space dimensionality reduction in shape optimization by Karhunen–Loève expansion,* Computer Methods in Applied Mechanics and Engineering, 283 (2015), pp. 1525–1544.

[6] A. MANZONI, F. SALMOIRAGHI, AND L. HELTAI, *Reduced Basis Isogeometric Methods (RB-IGA) for the real-time simulation of potential flows about parametrized NACA airfoils,* Computer Methods in Applied Mechanics and Engineering, 284 (2015), pp. 1147–1180.

[7] A. CORIGLIANO, M. DOSSI, AND S. MARIANI, *Model order reduction and domain decomposition strategies for the solution of the dynamic elastic–plastic structural problem,* Computer Methods in Applied Mechanics and Engineering, 290 (2015), pp. 127–155.

[8] A. G. BUCHAN ET AL., *A POD reduced order model for resolving angular direction in neutron/photon transport problems,* Journal of Computational Physics, 296 (2015), pp. 138–157.

[9] W. H. SCHILDERS, H. A. VAN DER VORST, AND J. ROMMES, Model Order Reduction: Theory, Research Aspects and applications, Springer, 2008.

[10] B. R. NOACK, M. MORZYNSKI, AND G. TADMOR, Reduced-Order Modelling for Flow Control. Springer Publishing Company, Incorporated, 2013.

[11] J. R. KOZA, *Genetic programming as a means for programming computers by natural selection,* Statistics and Computing, 4 (1994), pp. 87–112.

[12] M. F. HAMZA, H. J. YAP, AND I. A. CHOUDHURY, *Genetic algorithm and particle swarm optimization based cascade interval type 2 fuzzy PD controller for rotary inverted pendulum system,* Mathematical Problems in Engineering, 15 (2015).
[13] P. GUTMAN, C. MANNERFELT, AND P. MOLANDER, *Contributions to the model reduction problem,* IEEE Transactions on Automatic Control, 27 (1982), pp. 454–455.
[14] E. S. GOPI, Algorithm Collections for Digital Signal Processing Applications Using Matlab. Springer Publishing Company, Incorporated, 2007.
[15] M. AOKI, *Control of large-scale dynamic systems by aggregation,* IEEE Transactions on Automatic Control, 13 (1968), pp. 246–253.
[16] Y. SHAMASH, *Stable reduced-order models using Padé-type approximations,* IEEE Transactions on Automatic Control, 19 (1974), pp. 615–616.
[17] N. K. SINHA AND B. KUSZTA, *Modeling and identification of dynamic systems,* IEEE Transactions on Systems, Man, and Cybernetics, SMC-15, (1985), pp. 449–450.
[18] M. HUTTON AND B. FRIEDLAND, *Routh approximations for reducing order of linear, time-invariant systems,* IEEE Transactions on Automatic Control, 20 (1975), pp. 329–337.
[19] P. J. SCHMID, *Dynamic mode decomposition of numerical and experimental data,* Journal of Fluid Mechanics, 656 (2010), pp. 5–28.
[20] C. W. ROWLEY, I. MEZIĆ, S. BAGHERI, P. SCHLATTER, AND D. S. HENNINGSON, *Spectral analysis of nonlinear flows,* Journal of Fluid Mechanics, 641 (2009), pp. 115–127, 2009.
[21] S. A. L. GLEGG AND W. J. DEVENPORT, *Proper orthogonal decomposition of turbulent flows for aeroacoustic and hydroacoustic applications,* Journal of Sound and Vibration, 239 (2001), pp. 767–784.
[22] J. BURKARDT, M. GUNZBURGER, AND H.-C. LEE, *POD and CVT-based reduced-order modeling of Navier–Stokes flows,* Computer Methods in Applied Mechanics and Engineering, 196 (2006), pp. 337-355.
[23] Y. BENGIO, I. J. GOODFELLOW, AND A. COURVILLE, *Deep learning,* Nature, 521 (2015), pp. 436–444.
[24] J. SCHMIDHUBER, *Deep learning in neural networks: An overview,* Neural Networks, 61 (2015), pp. 85–117.
[25] Y. LECUN, Y. BENGIO, AND G. HINTON, *Deep learning*, Nature, 521 (2015), pp. 436–444.
[26] T. N. SAINATH, A. MOHAMED, B. KINGSBURY, AND B. RAMABHADRAN, *Deep convolutional neural networks for LVCSR,* in 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, (2013), pp. 8614–8618.
[27] H. Y. XIONG ET AL., *RNA splicing. The human splicing code reveals new insights into the genetic determinants of disease,* Science, 347 (2015), pp. 1254806–1254806.
[28] J. KOU AND W. ZHANG, *Data-driven modeling for unsteady aerodynamics and aeroelasticity,* Progress in Aerospace Sciences, 125 (2021), 100725.
[29] S. E. AHMED, O. SAN, D. A. BISTRIAN, AND I. M. NAVON, *Sampling and resolution characteristics in reduced order models of shallow water equations: Intrusive vs nonintrusive,* International Journal for Numerical Methods in Fluids, 92 (2020), pp. 992–1036.
[30] B. M. AFKHAM AND J. S. HESTHAVEN, *Structure preserving model reduction of parametric Hamiltonian systems,* SIAM Journal on Scientific Computing, 39 (2017), pp. A2616–A2644.
[31] D. SIPP, M. F. DE PANDO, AND P. J. SCHMID, *Nonlinear model reduction: A comparison between POD-Galerkin and POD-DEIM methods,* Comput. Fluids, 208 (2020), 104628.
[32] D. XIAO ET AL., *Non-linear model reduction for the Navier–Stokes equations using residual DEIM method,* Journal of Computational Physics, 263 (2014), pp. 1–18.
[33] E. J. PARISH, C. R. WENTLAND, AND K. DURAISAMY, *The Adjoint Petrov–Galerkin method*

for non-linear model reduction, Computer Methods in Applied Mechanics and Engineering, 365 (2020), 112991.

[34] R. REYES AND R. CODINA, *Projection-based reduced order models for flow problems: A variational multiscale approach,* Computer Methods in Applied Mechanics and Engineering, 363 (2020), 112844.

[35] D. XIAO ET AL., *Non-linear Petrov–Galerkin methods for reduced order modelling of the Navier–Stokes equations using a mixed finite element pair,* Computer Methods In Applied Mechanics and Engineering, 255 (2013), pp. 147–157.

[36] R. JIANG AND L. J. DURLOFSKY, *Implementation and detailed assessment of a GNAT reduced-order model for subsurface flow simulation,* Journal of Computational Physics, 379 (2019), pp. 192–213.

[37] R. ABGRALL AND R. CRISOVAN, *Model reduction using L 1-norm minimization as an application to nonlinear hyperbolic problems,* International Journal for Numerical Methods in Fluids, 87 (2018), pp. 628–651.

[38] T. BUI-THANH, M. DAMODARAN, AND K. WILLCOX, *Proper orthogonal decomposition extensions for parametric applications in compressible aerodynamics,* in 21st AIAA Applied Aerodynamics Conference, (2003), 4213.

[39] D. XIAO, F. FANG, C. PAIN, I. NAVON, P. SALINAS, AND Z. WANG, *Non-intrusive model reduction for a 3D unstructured mesh control volume finite element reservoir model and its application to fluvial channels,* International Journal of Oil, Gas and Coal Technology, 19 (2018), pp. 316–339.

[40] F. J. GONZALEZ AND M. BALAJEWICZ, *Deep convolutional recurrent autoencoders for learning low-dimensional feature dynamics of fluid systems,* 2018.

[41] S. WIEWEL, M. BECHER, AND N. THUEREY, *Latent space physics: Towards learning the temporal evolution of fluid flow,* in Computer Graphics Forum, 38 (2019), 38, pp. 71–82.

[42] J. S. HESTHAVEN AND S. UBBIALI, *Non-intrusive reduced order modeling of nonlinear problems using neural networks,* Journal of Computational Physics, 363 (2018), pp. 55–78.

[43] M. GUO AND J. S. HESTHAVEN, *Data-driven reduced order modeling for time-dependent problems,* Computer methods in applied mechanics and engineering, 345 (2019), pp. 75–99.

[44] D. XIAO ET AL., *A reduced order model for turbulent flows in the urban environment using machine learning,* Building and Environment, 148 (2019), pp. 323–337.

[45] O. SAN AND R. MAULIK, *Extreme learning machine for reduced order modeling of turbulent geophysical flows,* Physical Review E, 97 (2018), 042322.

[46] C. LU AND X. ZHU, *Bifidelity data-assisted neural networks in nonintrusive reduced-order modeling,* Journal of Scientific Computing, 87 (2021), pp. 1–30.

[47] S. PAWAR, S. RAHMAN, H. VADDIREDDY, O. SAN, A. RASHEED, AND P. VEDULA, *A deep learning enabler for nonintrusive reduced order modeling of fluid flows,* Physics of Fluids, 31 (2019), 085101.

[48] O. SAN, R. MAULIK, AND M. AHMED, *An artificial neural network framework for reduced order modeling of transient flows,* Communications in Nonlinear Science and Numerical Simulation, 77 (2019), pp. 271–287.

[49] S. PAWAR, S. E. AHMED, O. SAN, AND A. RASHEED, *Data-driven recovery of hidden physics in reduced order modeling of fluid flows,* Physics of Fluids, 32 (2020), 036602.

[50] K. LI, J. KOU, AND W. ZHANG, *Deep neural network for unsteady aerodynamic and aeroelastic modeling across multiple Mach numbers,* Nonlinear Dynamics, 96 (2019), pp. 2157–2177.

[51] Z. WANG, D. XIAO, F. FANG, R. GOVINDAN, C. PAIN, AND Y. GUO, *Model identification of reduced order fluid dynamics systems using deep learning,* (2017).

[52] M. WANG, H. LI, X. CHEN, AND Y. CHEN, *Deep learning-based model reduction for distributed parameter systems,* IEEE Transactions on Systems, Man, and Cybernetics: Systems, 46 (2016), pp. 1664–1674.

[53] S. E. AHMED, O. SAN, A. RASHEED, AND T. ILIESCU, *A long short-term memory embedding for hybrid uplifted reduced order models,* Physica D: Nonlinear Phenomena, (2020), 132471.

[54] P. R. VLACHAS, W. BYEON, Z. Y. WAN, T. P. SAPSIS, AND P. KOUMOUTSAKOS, *Data-driven forecasting of high-dimensional chaotic systems with long short-term memory networks,* Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences, 474 (2018), 20170844.

[55] A. T. MOHAN AND D. V. GAITONDE, *A deep learning based approach to reduced order modeling for turbulent flow control using LSTM neural networks,* 2018.

[56] S. M. RAHMAN, S. PAWAR, O. SAN, A. RASHEED, AND T. ILIESCU, *Nonintrusive reduced order modeling framework for quasigeostrophic turbulence,* Physical Review E, 100 (2019), 053306.

[57] M. KHERAD, M. K. MOAYYEDI, AND F. FOTOUHI, *Reduced order framework for convection dominant and pure diffusive problems based on combination of deep long short-term memory and proper orthogonal decomposition/dynamic mode decomposition methods,* International Journal for Numerical Methods in Fluids, 93 (2021), pp. 853–873.

[58] H. EIVAZI, H. VEISI, M. H. NADERI, AND V. ESFAHANIAN, *Deep neural networks for nonlinear model order reduction of unsteady flows,* Physics of Fluids, 32 (2020), 105104.

[59] K. FUKAMI, K. HASEGAWA, T. NAKAMURA, M. MORIMOTO, AND K. FUKAGATA, *Model order reduction with neural networks: Application to laminar and turbulent flows,* SN Computer Science, 2 (2021), pp. 1–16.

[60] S. XIANG ET AL., *Non-intrusive reduced order model of urban airflow with dynamic boundary conditions,* Building and Environment, 187 (2021), 107397.

[61] R. MAULIK, B. LUSCH, AND P. BALAPRAKASH, *Reduced-order modeling of advection-dominated systems with recurrent neural networks and convolutional autoencoders,* Physics of Fluids, 33 (2021), 037106.

[62] T. R. PHILLIPS, C. E. HEANEY, P. N. SMITH, AND C. C. PAIN, *An autoencoder-based reduced-order model for eigenvalue problems with application to neutron diffusion,* International Journal for Numerical Methods in Engineering, 2020.

[63] K. HASEGAWA, K. FUKAMI, T. MURATA, AND K. FUKAGATA, *CNN-LSTM based reduced order modeling of two-dimensional unsteady flows around a circular cylinder at different Reynolds numbers,* Fluid Dynamics Research, 52 (2020), 065501.

[64] T. NAKAMURA, K. FUKAMI, K. HASEGAWA, Y. NABAE, AND K. FUKAGATA, *Convolutional neural network and long short-term memory based reduced order surrogate for minimal turbulent channel flow,* Physics of Fluids, 33 (2021), 025116.

[65] K. HASEGAWA, K. FUKAMI, T. MURATA, AND K. FUKAGATA, *Machine-learning-based reduced-order modeling for unsteady flows around bluff bodies of various shapes,* Theoretical and Computational Fluid Dynamics, 34 (2020), pp. 367–383.

[66] C. QUILODRÁN-CASAS, R. ARCUCCI, C. PAIN, AND Y. GUO, *Adversarially trained LSTMs on reduced order models of urban air pollution simulations,* 2021.

[67] K. T. CARLBERG, A. JAMESON, M. J. KOCHENDERFER, J. MORTON, L. PENG, AND F. D. WITHERDEN, *Recovering missing CFD data for high-order discretizations using deep neural networks and dynamics learning,* Journal of Computational Physics, 395 (2019), pp. 105–124.

[68] R. MAULIK, T. BOTSAS, N. RAMACHANDRA, L. R. MASON, AND I. PAN, *Latent-space time evolution of non-intrusive reduced-order models using Gaussian process emulation,* Physica D: Nonlinear Phenomena, 416 (2021), 132797.

[69] Y. GUAN, S. L. BRUNTON, AND I. NOVOSSELOV, *Sparse nonlinear models of chaotic electroconvection,* Royal Society Open Science, 8 (2021), 202367.
[70] K. FUKAMI, T. MURATA, K. ZHANG, AND K. FUKAGATA, *Sparse identification of nonlinear dynamics with low-dimensionalized flow representations,* Journal of Fluid Mechanics, 926 (2021).
[71] M. O. WILLIAMS, I. G. KEVREKIDIS, AND C. W. ROWLEY, *A data–driven approximation of the koopman operator: Extending dynamic mode decomposition,* Journal of Nonlinear Science, 25 (2015), pp. 1307–1346.
[72] H. ARBABI AND I. MEZIC, *Ergodic theory, dynamic mode decomposition, and computation of spectral properties of the Koopman operator,* SIAM Journal on Applied Dynamical Systems, 16 (2017), pp. 2096–2126.
[73] B. LUSCH, J. N. KUTZ, AND S. L. BRUNTON, *Deep learning for universal linear embeddings of nonlinear dynamics,* Nature Communications, 9 (2018), pp. 1–10.
[74] N. TAKEISHI, Y. KAWAHARA, AND T. YAIRI, *Learning Koopman invariant subspaces for dynamic mode decomposition,* Advances in Neural Information Processing Systems, 30 (2017).
[75] Q. LI, F. DIETRICH, E. M. BOLLT, AND I. G. KEVREKIDIS, *Extended dynamic mode decomposition with dictionary learning: A data-driven adaptive spectral decomposition of the Koopman operator,* Chaos: An Interdisciplinary Journal of Nonlinear Science, 27 (2017), 103111.
[76] J. MORTON, A. JAMESON, M. J. KOCHENDERFER, AND F. WITHERDEN, *Deep dynamical modeling and control of unsteady fluid flows,* Advances in Neural Information Processing Systems, 31 (2018).
[77] M. A. KHODKAR, P. HASSANZADEH, AND A. ANTOULAS, *A Koopman-based framework for forecasting the spatiotemporal evolution of chaotic dynamics with nonlinearities modeled as exogenous forcings,* 2019.
[78] H. EIVAZI, L. GUASTONI, P. SCHLATTER, H. AZIZPOUR, AND R. VINUESA, *Recurrent neural networks and Koopman-based frameworks for temporal predictions in a low-order model of turbulence,* International Journal of Heat and Fluid Flow, 90 (2021), 108816.
[79] S. R. BUKKA, R. GUPTA, A. R. MAGEE, AND R. K. JAIMAN, *Assessment of unsteady flow predictions using hybrid deep learning based reduced-order models,* Physics of Fluids, 33 (2021), 013601.
[80] H. BOURLARD, *Auto-association by multilayer perceptrons and singular value decomposition,* IDIAP2000.
[81] P. VINCENT, H. LAROCHELLE, I. LAJOIE, Y. BENGIO, AND P.-A. MANZAGOL, *Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion,* Journal of machine learning research, 11 (2010), pp. 3371–3408.
[82] C. POULTNEY, S. CHOPRA, AND Y. L. CUN, *Efficient learning of sparse representations with an energy-based model,* Advances in Neural Information Processing Systems, (2007), pp. 1137–1144.
[83] I. GOODFELLOW, Y. BENGIO, AND A. COURVILLE, Deep Learning, MIT press, 2016.
[84] G. E. HINTON AND R. R. SALAKHUTDINOV, *Reducing the dimensionality of data with neural networks,* Science, 313 (2006), pp. 504–507.
[85] H. SALEHINEJAD, S. SANKAR, J. BARFETT, E. COLAK, AND S. VALAEE, *Recent advances in recurrent neural networks,* 2017.
[86] N. ANDREW, Sequence Models, (2015).
[87] F. A. GERS, J. SCHMIDHUBER, AND F. CUMMINS, Learning to Forget: Continual Prediction with LSTM, 1999.
[88] G. KLAMBAUER, T. UNTERTHINER, A. MAYR, AND S. HOCHREITER, *Self-normalizing neural networks,* Advances in Neural Information Processing Systems, (2017), pp. 971–980.

[89] A. E. GILL, Atmosphere—Ocean Dynamics, Elsevier, 2016.
[90] M. G. LARSON AND F. BENGZON, The Finite Element Method: Theory, Implementation, and Applications, Springer Science & Business Media, 2013.
[91] A. SOHANKAR, C. NORBERGB, AND L. DAVIDSON, *Numerical simulation of unsteady low-Reynolds number flow around rectangular cylinders at incidence,* Journal of Wind Engineering and Industrial Aerodynamics, 69 (1997), pp. 189–201.
[92] M. CHENG, D. WHYTE, AND J. LOU, *Numerical simulation of flow around a square cylinder in uniform-shear flow,* Journal of Fluids and Structures, 23 (2007), pp. 207–226.
[93] S. BERRONE, V. GARBERO, AND M. MARRO, *Numerical simulation of low-Reynolds number flows past rectangular cylinders based on adaptive finite element and finite volume methods,* Comput. Fluids, 40 (2011), pp. 92–112.
[94] K. LAM, Y. LIN, L. ZOU, AND Y. LIU, *Numerical study of flow patterns and force characteristics for square and rectangular cylinders with wavy surfaces,* Journal of Fluids and structures, 28 (2012), pp. 359–377.
[95] M. MORIMOTO, K. FUKAMI, K. ZHANG, A. G. NAIR, AND K. FUKAGATA, *Convolutional neural networks for fluid flow analysis: toward effective metamodeling and low dimensionalization,* Theoretical and Computational Fluid Dynamics, 35 (2021), pp. 633–658.
[96] T. MURATA, K. FUKAMI, AND K. FUKAGATA, *Nonlinear mode decomposition with convolutional neural networks for fluid dynamics,* Journal of Fluid Mechanics, 882 (2020).
[97] K. FUKAMI, T. NAKAMURA, AND K. FUKAGATA, *Convolutional neural network based hierarchical autoencoder for nonlinear mode decomposition of fluid field data,* Physics of Fluids, 32 (2020), 095110.
[98] H. EIVAZI, S. LE CLAINCHE, S. HOYAS, AND R. VINUESA, *Towards extraction of orthogonal and parsimonious non-linear modes from turbulent flows,* Expert Systems with Applications, 202 (2022), 117038.
[99] H. GAO, L. SUN, AND J.-X. WANG, *PhyGeoNet: Physics-informed geometry-adaptive convolutional neural networks for solving parameterized steady-state PDEs on irregular domain,* Journal of Computational Physics, 428 (2021), 110079.
[100] J. TENCER AND K. M. POTTER, *Enabling nonlinear manifold projection reduced-order models by extending convolutional neural networks to unstructured data,* Sandia National Lab.(SNL-NM), Albuquerque, NM, 2020.
[101] K. FUKAMI, R. MAULIK, N. RAMACHANDRA, K. FUKAGATA, AND K. TAIRA, *Global field reconstruction from sparse sensors with Voronoi tessellation-assisted deep learning,* Nature Machine Intelligence, 3 (2021), pp. 945–951.
[102] A. KASHEFI, D. REMPE, AND L. J. GUIBAS, *A point-cloud deep learning framework for prediction of fluid flow fields on irregular geometries,* Physics of Fluids, 33 (2021), 027104.