Available online at www.sciencedirect.com
ScienceDirect
Comput. Methods Appl. Mech. Engrg. 372 (2020) 113375
Computer methods in applied mechanics and engineering
www.elsevier.com/locate/cma

An advanced hybrid deep adversarial autoencoder for parameterized nonlinear fluid flow modelling

M. Cheng[a], F. Fang[a,*], C.C. Pain[a], I.M. Navon[b]

[a] *Applied Modelling and Computation Group, Department of Earth Science and Engineering, Imperial College London, SW7 2BP, UK*
[b] *Department of Scientific Computing, Florida State University, Tallahassee, FL, 32306-4120, USA*

Received 8 October 2019; received in revised form 29 June 2020; accepted 7 August 2020
Available online 25 August 2020

Abstract

Considering the high computation cost required in conventional computation fluid dynamic simulations, machine learning methods have been introduced to flow dynamic simulations in years, aiming on reducing CPU time. In this work, we propose a hybrid deep adversarial autoencoder (VAE-GAN) to integrate generative adversarial network (GAN) and variational autoencoder (VAE) for predicting parameterized nonlinear fluid flows in spatial and temporal dimensions. High-dimensional inputs are compressed into the low-dimensional representations by nonlinear functions in a convolutional encoder. In this way, the predictive fluid flows reconstructed in a convolutional decoder contain the dynamic fluid flow physics of high nonlinearity and chaotic nature. In addition, the low-dimensional representations are applied to the adversarial network for model training and parameter optimization, which enables fast computation process. The capability of the hybrid VAE-GAN is illustrated by varying inputs on a flow past a cylinder test case as well as a second case of water column collapse. Numerical results show that this hybrid VAE-GAN has successfully captured the spatio-temporal flow features with CPU speed-up of three orders of magnitude. These promising results suggest that the hybrid VAE-GAN can play a critical role in efficiently and accurately predicting complex flows in future research efforts.

Keywords: Parameterized; Nonlinear fluid flows; Generative adversarial networks; Variational autoencoder; Model reduction

1. Introduction

Numerical simulations of nonlinear fluid flows are used to describe the complex evolution of many physical processes. Accurate simulations of nonlinear fluid flows are of great importance to many fields such as atmosphere, flood and ocean modelling, which systems often exhibit rich flow dynamics in both space and time [1,2]. The numerical simulations have benefited from the availability of high-resolution spatio-temporal data due to recent advances in measurement techniques [3], which make the studies of more complex flows possible. However, the computational cost involved in solving complex problems is intensive which still precludes the development in these areas. In order to address the issue of the high computational cost, this paper proposes a hybrid deep adversarial autoencoder to solve fluid problems in an efficient manner.

* Corresponding author.
E-mail address: f.fang@imperial.ac.uk (F. Fang).

https://doi.org/10.1016/j.cma.2020.113375

Recent advances in machine learning technologies are increasingly of interest for efficiently simulating flows in dynamical systems [4–6]. Machine learning has demonstrated its potential capability in fluid flow applications [7], such as feature extraction [8–10], dimensionality reduction [5,11] and superresolution [12]. For example, Kim et al. [13] adopted a convolutional neural network (CNN) to extract flow features and reconstruct fluid fields. Gonzalez and Balajewicz [11] developed a deep convolutional recurrent autoencoder for nonlinear model reduction, which identified the low-dimensional feature dynamics by the high-dimensional datasets and modelled flow dynamics on the underlying manifold space. Xie et al. [12] proposed a temporally coherent generative model to produce the super-resolution fluid flows. Despite the fact that these research efforts demonstrated how to successfully reconstruct flow dynamics, it is noted that they commonly do not take into account the temporal and spatial evolution of inputs or parameters, which is crucial for realistic dynamical systems [14].

Most recently, it has been shown that Generative Adversarial Network (GAN) is an efficient approach for reconstructing the spatial distribution of tracers and flow fields [15]. GAN introduced by Goodfellow et al. [16], has emerged as having a leading role in recreating the distributions of complex data [12,17]. The key feature of GAN is the adversarial strategy consisting of two modules. GAN defines a learning generative network by transforming a latent variable into a state variable using nonlinear functions. Then GAN drives the learning process by discriminating the observed data from the generated data in a discriminator network. Because of the special adversarial architecture, GAN has demonstrated great capability in producing high-resolution samples, mostly in images, e.g., image synthesis [18], semantic image editing [19], style transfer [20] etc.

For efficient GAN training and accurate spatio-temporal fluid flow prediction, variational autoencoder (VAE) [21] has been introduced to GAN in this study. VAE has been widely used in various research areas, such as text generation [22], facial attribute prediction [23], image generation [24,25], graph generation [26], music synthesis [27], and speech emotion classification [28], etc. The hybrid deep adversarial autoencoder (VAE-GAN) developed here takes advantage of both GAN and VAE. GAN allows for training on large datasets and is fast to yield visually high-resolution images, but the flexible architecture is easy to result in the model collapse problem and generate unreal results [29]. The VAE is attractive for achieving better log-likelihoods than GAN [30,31], this suggests a hybrid VAE-GAN to better represent all the training data and discourage model-collapse problem in GAN [29].

The hybrid VAE-GAN developed here is a robust and efficient numerical tool for accurate prediction of parameterized nonlinear fluid flows. The advantages of the hybrid VAE-GAN include:

- The proposed method exploits spatial features by use of convolutional neural networks. It will be advantageous over the traditional reduced order models (ROMs) based on proper orthogonal decomposition (POD) [32,33], Galerkin regression [34,35], cluster-based reduced order model (CROM) [36,37] and dynamic mode decomposition (DMD) [38], which are used for model reduction of dynamical systems by linear combinations of the original snapshots. In convolutional neural networks, the high-dimensional datasets are compressed into the low-dimensional representations by nonlinearity functions in a convolutional encoder. In this way, the predictive fluid flows containing both high nonlinearity and chaotic nature can be represented by a convolutional decoder.
- The low-dimensional representations, several orders of magnitude smaller than the dimensional size of the original datasets, are applied into the adversarial network for representation learning and parameter optimization, thus accelerating the computation process.
- With the trained hybrid VAE-GAN, for given unseen inputs, the spatio-temporal features can be automatically extracted in the encoder. Consequently, accurate predictive nonlinear fluid fields can be further obtained in the decoder in an efficient manner.

This is the first time that the hybrid VAE-GAN is adopted to address parameterized nonlinear fluid flow problems. It is expected to make a breakthrough in predicting accurate nonlinear fluid flows with the high-speed computation.

The remainder of this paper is organized as follows. Section 2 presents the governing equation for nonlinear fluid flows in dynamic systems. Methodologies of VAE and GAN are briefly introduced in Sections 3 and 4, respectively. The hybrid VAE-GAN for parameterized nonlinear fluid fields is described in detail in Section 5. Section 6 demonstrates the performance of the hybrid VAE-GAN using a flow past a cylinder and water column collapse as test cases. Finally in Section 7, conclusions are presented.

2. Governing equations

In the computational fluid dynamic models, nonlinear fluid flows are usually simulated by solving nonlinear partial differential equations, such as Navier–Stokes equations. In general, a parameterized partial differential equation for a spatio-temporal fluid flow problem can be written as:

$$\mathcal{M}(\hbar(x,\mu,t),x,\mu,t)=\mathcal{F}(\hbar(x,\mu,t),x,\mu,t), \tag{1}$$

where $\mathcal{M}$ denotes a nonlinear partial differential operator, $\hbar(x,\mu,t)$ is the state variable vector (for example, pressure, density, velocity, etc.), x represents the spatial coordinate system, μ denotes the parameter vector (for example, model input and boundary condition), t is the time and $\mathcal{F}$ is the source term.

3. Variational autoencoder for model reduction

VAE was introduced by Kingma and Welling [21], which combines Bayesian inference with deep learning. The VAE is a generative model which aims to produce the desired local variable $\hbar$ from the underlying latent variable ζ (as shown in Fig. 1). Mathematically, let $p(\zeta)$ be a prior distribution of ζ, and the probability of the local state variable $\hbar$ be modelled by

$$\hbar\sim p_\theta(\hbar|\zeta),\quad p_\theta(\hbar)=\int p_\theta(\hbar|\zeta)p_\theta(\zeta)d\zeta, \tag{2}$$

where $p_\theta(\hbar|\zeta)$ is the conditional distribution of the local state variable $\hbar$ given ζ, which is modelled by a deep neural network (called decoder) with parameters θ (for example, weights and bias in the decoder).

Since the true posterior $p_\theta(\zeta|\hbar)=p_\theta(\hbar|\zeta)p_\theta(\zeta)/p_\theta(\hbar)$ is intractable, the integral of the marginal likelihood of $p_\theta(\hbar)$ is also intractable [21]. To evaluate $p_\theta(\hbar)$, the distribution of the latent state vector $p_\theta(\zeta)$ is usually modelled by $q_\phi(\zeta|\hbar)$ as:

$$\zeta\sim q_\phi(\zeta|\hbar), \tag{3}$$

where $q_\phi(\zeta|\hbar)$ represents the conditional distribution of the ζ given local state variable $\hbar$ (as shown in Fig. 1), which is a deep neural network (called encoder) with parameters ϕ (for example, weights and bias in the encoder). The encoder enables model reduction for large-scale datasets in dynamic fluid systems [5].

To achieve the sample reconstruction, the reconstruction loss $\mathcal{L}_{rec}$ as the negative expected log-likelihood of the samples needs to be maximized, as follows:

$$\mathcal{L}_{rec}=E_{q_\phi(\zeta|\hbar)}(\log p_\theta(\hbar|\zeta)). \tag{4}$$

The difference (called Kullback–Leibler divergence) between the distribution of $q_\phi(\zeta|\hbar)$ and the intractable true posterior $p_\theta(\zeta|\hbar)$ can be quantified as

$$\mathcal{L}_{KL}=D_{KL}(q_\phi(\zeta|\hbar)\|p_\theta(\zeta|\hbar)). \tag{5}$$

The total loss $\mathcal{L}_{vae}$ contains the loss from the sample reconstruction (Eq. (4)) and the difference between the generated and true posteriors (Eq. (5)) ($\mathcal{L}_{vae}=\mathcal{L}_{rec}+\mathcal{L}_{KL}$) ($\mathcal{L}_{KL}$ is replaced by the adversarial loss $\mathcal{L}_{adv}$ in Eq. (27) of the hybrid VAE-GAN, see Section 5.1.2), which can be minimized by gradient descent algorithms [21]. Since the KL divergence is non-negative, the total loss $\mathcal{L}_{vae}$ can be expressed as:

$$\mathcal{L}_{vae}=-D_{KL}(q_\phi(\zeta|\hbar)\|p_\theta(\zeta|\hbar))+E_{q_\phi(\zeta|\hbar)}(\log p_\theta(\hbar|\zeta)). \tag{6}$$

Correspondingly, the parameters of the encoder and decoder in VAE are optimized by maximizing the expectation of the lower bound of the log marginal likelihood as follows:

$$\max_\theta\max_\phi E_{p_{data}(\hbar)}[-D_{KL}(q_\phi(\zeta|\hbar)\|p_\theta(\zeta|\hbar))+E_{q_\phi(\zeta|\hbar)}(\log p_\theta(\hbar|\zeta))], \tag{7}$$

where $p_{data}(\hbar)$ is the prior distribution of $\hbar$.

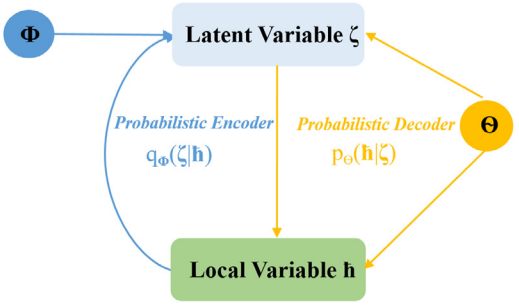

Fig. 1. Illustration of a VAE model. Yellow lines denote the generative process in a probabilistic decoder $p_\theta(\hbar|\zeta)$, blue lines represent the variational approximation $q_\phi(\zeta|\hbar)$ in a probabilistic encoder to the intractable posterior $p_\theta(\hbar|\zeta)$. ϕ and θ are parameters which are learned jointly (see [21]). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

4. Generative adversarial network

The GAN is generally implemented with a minimax game in a system of two players [39]. One player is responsible for generating the new samples $\hbar$ from a random dataset μ, while another player aims to discriminate the real samples $\hbar_d$ from the generated samples $\hbar$. The former player is called the generator $\mathcal{G}$ and the latter is the discriminator $\mathcal{D}$.

In the discriminator, $\mathcal{D}(\hbar_d) = 1$ if the real samples $\hbar_d$ are accepted while $\mathcal{D}(\mathcal{G}(\mu)) = 0$ if the generated samples $\hbar$ ($\hbar = \mathcal{G}(\mu)$) are rejected. Unlike the autoencoder, GAN is a two-player game rather than optimizing one loss function $\mathcal{L}_{vae}$ (as in Eq. (7)) in VAE. During the training process of GAN, the parameters in the discriminator are updated by maximizing $\mathcal{L}_D$ as:

$$\mathcal{L}_D = E_{\hbar_d \sim p_{data}(\hbar_d)}[\log \mathcal{D}(\hbar_d)] + E_{\mu \sim p_\mu(\mu)}[\log(1 - \mathcal{D}(\mathcal{G}(\mu)))], \tag{8}$$

while the parameters in the generator are updated by minimizing $\mathcal{L}_G$ as:

$$\mathcal{L}_G = E_{\mu \sim p_\mu(\mu)}[\log(1 - \mathcal{D}(\mathcal{G}(\mu)))], \tag{9}$$

where $p_\mu(\mu)$ is a prior distribution for the random dataset μ, and $p_{data}(\hbar_d)$ is the corresponding probability data distribution for the real datasets $\hbar_d$.

Generally, the objective function for GAN is written as follows:

$$\min_{\mathcal{G}} \max_{\mathcal{D}} \mathcal{L}(\mathcal{G}, \mathcal{D}) = E_{\hbar_d \sim p_{data}(\hbar_d)}[\log \mathcal{D}(\hbar_d)] + E_{\mu \sim p_\mu(\mu)}[\log(1 - \mathcal{D}(\mathcal{G}(\mu)))], \tag{10}$$

In practice, the optimization of $\mathcal{L}(\mathcal{G}, \mathcal{D})$ is performed using gradient-based methods, e.g. the adaptive Moment Estimation (Adam) optimizer [40]. Given enough capacity, the game converges to a global optimum where $\mathcal{D}(\hbar_d) = \frac{1}{2}$ everywhere (details shown in [16]).

5. Hybrid deep adversarial autoencoder for nonlinear fluid flow modelling

In this paper, for nonlinear fluid flow modelling, a hybrid deep learning fluid model based on deep adversarial autoencoder (VAE-GAN) is proposed by combining a VAE and a GAN.

In spatio-temporal fluid flow simulations, the state variable vector $\hbar$ represents the fluid flow distribution as

$$\hbar = \begin{bmatrix} \hbar_{t_1}^1 & \hbar_{t_1}^2 & \cdots & \hbar_{t_1}^{N_x} \\ \hbar_{t_2}^1 & \hbar_{t_2}^2 & \cdots & \hbar_{t_2}^{N_x} \\ \vdots & \vdots & & \vdots \\ \hbar_{t_{N_t}}^1 & \hbar_{t_{N_t}}^2 & \cdots & \hbar_{t_{N_t}}^{N_x} \end{bmatrix}, \tag{11}$$

and the state variable vector $\hbar$ in Eq. (11) can be rewritten as:

$$\hbar_{t_{n_t}} = (\hbar_{t_{n_t}}^1, \ldots, \hbar_{t_{n_t}}^{n_x}, \ldots, \hbar_{t_{n_t}}^{N_x}), \quad \hbar^{n_x} = (\hbar_{t_1}^{n_x}, \ldots, \hbar_{t_{n_t}}^{n_x}, \ldots, \hbar_{t_{N_t}}^{n_x})^T, \tag{12}$$

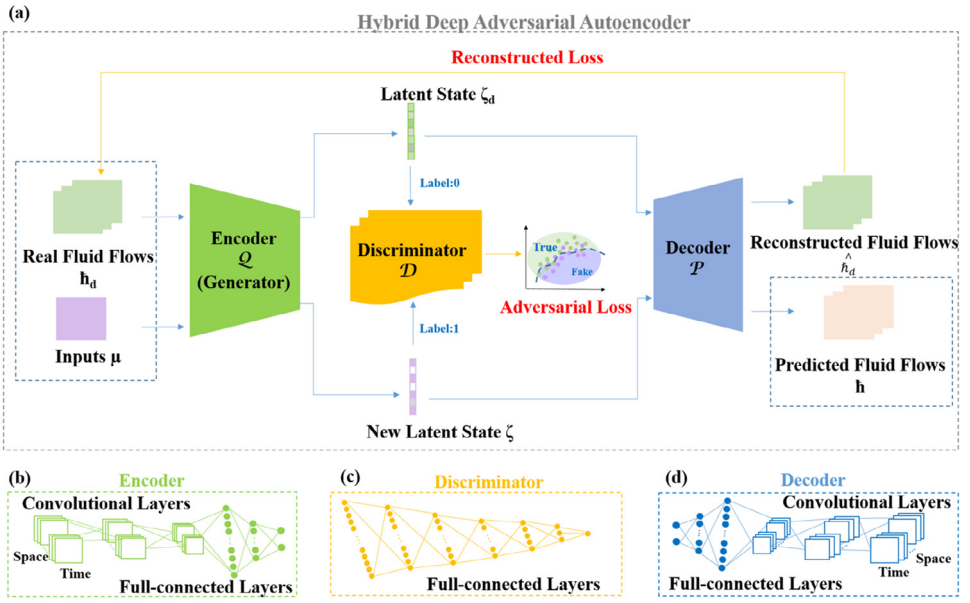

Fig. 2. (a) Illustration of a hybrid deep adversarial autoencoder (VAE-GAN), which consists of an encoder (acted as a generator) (b), a discriminator (c), and a decoder (d). The encoder and decoder compose of convolutional layers and full-connected layers. The discriminator is a stack of full-connected layers.

where $\hbar_{t_{n_t}}$ is the state variable spatial solution at time level t_{n_t} ($t_{n_t} \in [0, t_{N_t}]$, N_t is the number of timesteps), $\hbar^{n_x}$ is the state variable spatial solution at the point n_x ($n_x \in [0, N_x]$, N_x denotes the number of points in scalar grids of the computational domain Ω).

5.1. Hybrid deep adversarial autoencoder

When the dimension of $\hbar \in \Re^{N_t \times N_x}$ is large, efficiently predicting Eq. (1) for given inputs (e.g. boundary conditions) becomes challenging. Therefore, in the hybrid VAE-GAN (as described in Fig. 2(a)), the inputs μ and the real fluid fields $\hbar_d$ (the targeted outputs) are fed into the encoder, which transforms the high-dimensional datasets into low-dimensional representations ζ and ζ_d respectively. The discriminator can discriminate the low-dimensional representations ζ and ζ_d in an efficient way. The latent states ζ_d and ζ are then transformed into the reconstructed fluid fields $\hat{\hbar}_d$ (the reconstructed outputs) and predicted fluid fields $\hbar$ (the predicted outputs) respectively in the decoder. In the backward propagation process, the parameters in modules: encoder, decoder and discriminator are updated by minimizing the reconstruction loss and maximizing the adversarial loss.

As shown in Fig. 2(a), the encoder is acting as a generator in the hybrid VAE-GAN. In comparison to Eq. (3) in VAE, the generator here defines an aggregated posterior distribution of $q(\zeta)$ on the latent state as follows:

$$q(\zeta) = \int q_\phi(\zeta|\hbar) q_{data}(\hbar) d\hbar, \tag{13}$$

where $q_{data}(\hbar)$ is the data distribution for the datasets $\hbar$. The autoencoder in the hybrid VAE-GAN is regularized by matching the aggregated posterior $q(\zeta)$ to the prior $p(\zeta)$ in Eq. (2).

Given the parameter vector $\mu = [\mu^1, \mu^2, \ldots, \mu^{N_x}]$ as inputs and the state variable vector $\hbar_d = [\hbar_d^1, \hbar_d^2, \ldots, \hbar_d^{N_x}]$ as the targeted outputs, the model architecture, loss function and offline training and online predictive processes of the hybrid VAE-GAN are detailedly introduced as follows.

5.1.1. Model architecture

As shown in Fig. 2(b), (c) and (d), the modules: encoder, decoder and discriminator, consist of deep neural networks (containing convolutional layers and full-connected layers), which are used for extracting spatio-temporal

features. Let h^{l-1} and h^l be the input and output for the lth layer, respectively. The output of a layer is computed as

$$h_j^l = \sigma^l(\sum_{i=1}^{N^{l-1}} h_i^{l-1} * w_{ij}^l + b_j^l), \tag{14}$$

where h_j^l denotes the jth output-feature-map of layer l, σ represents the activation function, N^{l-1} is the number of input-feature-maps of layer $(l-1)$, w_{ij}^l is the weight linking the ith input map to jth output map of layer l, and b_j^l is the bias of jth output map of layer l.

For the activation function σ, the rectifier non-linearity (ReLU) activation function ($\mathcal{I}(\alpha) = \max(0, \alpha)$) is normally recommended for the hidden layers, and the hyperbolic tangent function ($\mathcal{I}(\alpha) = \tanh(\alpha)$) and the sigmoid function ($\mathcal{I}(\alpha) = 1/(1 + \exp^{-\alpha})$) are chosen for the output layer [41,42].

For simplifying the Eq. (14), it can be rewritten as:

$$h_l = \mathcal{H}(h_{l-1}, \sigma_l, \delta_l), \tag{15}$$

where $\delta_l \equiv (w_l, b_l)$ represents the parameters in the lth layer.

Design of the Encoder: Mapping the targeted outputs $\hbar_d$ and inputs μ to the latent state ζ_d and ζ respectively as:

$$\mathcal{Q} : \hbar_d \in \Re^{N_{\hbar_d,el}} \mapsto \zeta_d \in \Re^{N_{\zeta_d,el}}, \mu \in \Re^{N_{\mu,el}} \mapsto \zeta \in \Re^{N_{\zeta,el}}, \tag{16}$$

where $\Re^{N_{\hbar_d,el}}$ and $\Re^{N_{\mu,el}}$ are the dimensions of the targeted outputs and inputs respectively, $\Re^{N_{\zeta_d,el}}$ and $\Re^{N_{\zeta,el}}$ are the dimensions of the latent states respectively.

The encoder $\mathcal{Q}$ forms the deep convolutional neural networks with N_e layers, producing the latent state ζ_d as:

$$\zeta_d = \mathcal{Q}(\hbar_d, \theta) = \mathcal{H}_{N_e}^{(E)}(\sigma_{N_e}^{(E)}; \delta_{N_e}^{(E)}) \circ \ \mathcal{H}_{N_e-1}^{(E)}(\sigma_{N_e-1}^{(E)}; \delta_{N_e-1}^{(E)}) \circ \cdots \circ \mathcal{H}_1^{(E)}(\hbar_d; \delta_1^{(E)}), \tag{17}$$

where $\mathcal{H}_l^{(E)}(\sigma_l^{(E)}; \delta_l^{(E)})$ represents the layer l of the neural network, $l = 1, \ldots, N_e$, $\sigma_l^{(E)}$ represents the activation function at the layer l, $\delta_l^{(E)}$ denotes the weights and bias at the layer l, and $\theta \equiv (\delta_1^{(E)}, \ldots, \delta_{N_e}^{(E)})$.

Similarly, the latent state ζ is generated by the encoder as:

$$\zeta = \mathcal{Q}(\mu, \theta) = \mathcal{H}_{N_e}^{(E)}(\sigma_{N_e}^{(E)}; \delta_{N_e}^{(E)}) \circ \ \mathcal{H}_{N_e-1}^{(E)}(\sigma_{N_e-1}^{(E)}; \delta_{N_e-1}^{(E)}) \circ \cdots \circ \mathcal{H}_1^{(E)}(\mu; \delta_1^{(E)}). \tag{18}$$

Design of the Discriminator: The discriminator aims to judge the latent states ζ_d and ζ as:

$$\mathcal{D} : \zeta \in \Re^{N_{\zeta,el}} \mapsto p_\zeta, \zeta_d \in \Re^{N_{\zeta,el}} \mapsto q_{\zeta_d}. \tag{19}$$

The discriminator forms deep neural networks with M_d layers can be expressed as:

$$\mathcal{D}(\zeta, \zeta_d, \psi) = \mathcal{H}_{M_d}^{(M)}(\sigma_{M_d}^{(M)}; \delta_{M_d}^{(M)}) \circ \ \mathcal{H}_{M_d-1}^{(M)}(\sigma_{M_d-1}^{(M)}; \delta_{M_d-1}^{(M)}) \circ \cdots \circ \mathcal{H}_1^{(M)}(\zeta; \zeta_d; \delta_1^{(M)}), \tag{20}$$

where $\mathcal{H}_l^{(M)}(\sigma_l^{(M)}; \delta_l^{(M)})$ represents the layer l of the neural network, $l = 1, \ldots, M_d$, $\sigma_l^{(M)}$ represents the activation function at the layer l, $\delta_l^{(M)}$ denotes the weights and bias at the layer l, and $\psi \equiv (\delta_1^{(M)}, \ldots, \delta_{M_d}^{(M)})$.

Design of the Decoder: Reconstructing the new latent state ζ_d to the targeted output $\hat{\hbar}_d$ and predicting the outputs $\hbar$ by latent state ζ as:

$$\mathcal{P} : \zeta_d \in \Re^{N_{\zeta_d,el}} \mapsto \hat{\hbar}_d \in \Re^{N_{\hat{\hbar}_d,el}}, \zeta \in \Re^{N_{\zeta,el}} \mapsto \hbar \in \Re^{N_{\hbar,el}}, \tag{21}$$

where $\Re^{N_{\hbar,el}}$ is the dimension of the outputs.

The decoder $\mathcal{P}$ with N_d layers takes the form:

$$\hat{\hbar}_d = \mathcal{P}(\zeta_d, \phi) = \mathcal{H}_{N_d}^{(D)}(\sigma_{N_d}^{(D)}; \delta_{N_d}^{(D)}) \circ \ \mathcal{H}_{N_d-1}^{(D)}(\sigma_{N_d-1}^{(D)}; \delta_{N_d-1}^{(D)}) \circ \cdots \circ \mathcal{H}_1^{(D)}(\zeta_d; \delta_1^{(D)}), \tag{22}$$

where $\mathcal{H}_l^{(D)}(\sigma_l^{(D)}; \delta_l^{(D)})$ represents the layer l of the neural network, $l = 1, \ldots, N_d$, $\sigma_l^{(D)}$ represents the activation function at the layer l, $\delta_l^{(D)}$ denotes the weights and bias at the layer l, and $\phi \equiv (\delta_1^{(D)}, \ldots, \delta_{N_d}^{(D)})$.

Similarly, the outputs $\hbar$ can be expressed as:

$$\hbar = \mathcal{P}(\zeta, \phi) = \mathcal{H}_{N_d}^{(D)}(\sigma_{N_d}^{(D)}; \delta_{N_d}^{(D)}) \circ \ \mathcal{H}_{N_d-1}^{(D)}(\sigma_{N_d-1}^{(D)}; \delta_{N_d-1}^{(D)}) \circ \cdots \circ \mathcal{H}_1^{(D)}(\zeta; \delta_1^{(D)}). \tag{23}$$

5.1.2. Loss function

As introduced by Makhzani et al. [43], the encoder, decoder and discriminator of the hybrid VAE-GAN are trained jointly in the reconstruction phase and the regularization phase. In the reconstruction phase, the hybrid VAE-GAN updates the encoder $\mathcal{Q}$ and the decoder $\mathcal{P}$ to minimize the errors between the reconstructed targeted outputs $\hat{\hbar}_d$ and targeted outputs $\hbar_d$. In the regularization phase, the adversarial network updates the discriminator $\mathcal{D}$ to distinguish the true latent state ζ (as Eq. (18)) from the latent state ζ_d (as Eq. (17)) generated by encoder. The loss functions in two phases are as follows.

Reconstruction loss $\Rightarrow$ With jointing the encoder and the decoder, the autoencoder is trained using the reconstruction loss as:

$$\mathcal{L}_{rec}(\theta, \phi) = \|\hbar_d - \mathcal{P}(\mathcal{Q}(\hbar_d, \theta), \varphi)\| . \tag{24}$$

Adversarial loss $\Rightarrow$ The discriminator is trained to distinguish between the latent states ζ and ζ_d based the loss function:

$$\mathcal{L}_d(\theta, \phi, \psi) = -E_{\zeta\sim p_\zeta(\zeta)}(\log \mathcal{D}(\mathcal{Q}(\mu))) - E_{\zeta_d\sim p_{\zeta_d}(\zeta_d)}(\log(1 - \mathcal{D}(\mathcal{Q}(\hbar_d)))), \tag{25}$$

while the encoder (generator) tries to fool the discriminator using the following loss function:

$$\mathcal{L}_g(\theta, \phi, \psi) = -E_{\zeta_d\sim p_{\zeta_d}(\zeta_d)}(\log(1 - \mathcal{D}(\mathcal{Q}(\hbar_d)))). \tag{26}$$

The adversarial loss function is:

$$\mathcal{L}_{adv}(\theta, \phi, \psi) = -\mathcal{L}_d(\theta, \phi, \psi) = E_{\zeta\sim p_\zeta(\zeta)}(\log \mathcal{D}(\mathcal{Q}(\mu))) + E_{\zeta_d\sim p_{\zeta_d}(\zeta_d)}(\log(1 - \mathcal{D}(\mathcal{Q}(\hbar_d)))). \tag{27}$$

Hybrid loss $\Rightarrow$ The hybrid VAE-GAN attempts to minimize the reconstruction loss and maximize the adversarial loss. The hybrid loss can be obtained as:

$$\min_{\theta,\phi}\max_{\psi} E_{\zeta\sim p_\zeta(\zeta)}(\log \mathcal{D}(\mathcal{Q}(\mu))) + E_{\zeta_d\sim p_{\zeta_d}(\zeta_d)}(\log(1 - \mathcal{D}(\mathcal{Q}(\hbar_d)))) + \|\hbar_d - \mathcal{P}(\mathcal{Q}(\hbar_d, \theta), \varphi)\| . \tag{28}$$

The above Eqs. (24) to (28) show that the hybrid VAE-GAN is intuitively similar to the VAE, except for the KL divergence penalty of the VAE being replaced by the adversarial loss. The adversarial training procedure makes the hybrid VAE-GAN more flexible in choice of the latent variable ζ, which has been discussed in the work of Makhzani et al. [43]. The hybrid model is trained using the stochastic gradient descent by performing alternative updates of each component as Algorithm 1.

5.1.3. Offline training and online predictive processes

The hybrid VAE-GAN applied for nonlinear fluid flow modelling consists of the following three processes (as shown in Fig. 3):

(1) Data preparation process: To develop an input–output relationship, the initial and boundary conditions $\{\mu^1, \mu^2, \ldots, \mu^{N_x}\}$ are chosen as the inputs and the spatio-temporal distributions of flow variables $\{\hbar_d^1, \hbar_d^2, \ldots, \hbar_d^{N_x}\}$ are acted as the outputs, where N_x represents the number of nodes in the domain area Ω. It is worth noting that the outputs are collected in the whole timesteps.

(2) Offline training and validation process: Using the input–output data pairs $(\mu, \hbar_d)$ collected in (1), the hybrid VAE-GAN is trained with the inputs ϑ_{tr} and corresponding outputs δ_{tr} (where ϑ_{tr} and δ_{tr} denote the training inputs and the outputs, respectively). Once the training and validation processes are completed, the relationship between the input–output pair is determined.

(3) Online predictive process: Given a new series of predictive inputs $\mu \in (\vartheta \backslash \vartheta_{tr})$ (where ϑ is the whole inputs), the predictive outputs $\hbar$ are directly predicted by the hybrid VAE-GAN.

6. Numerical examples

In this section, two examples, flow past a cylinder and water column collapse, are used to illustrate the capabilities of the hybrid VAE-GAN in resolving nonlinear fluid flow problem governed by the Navier–Stokes equations. The input–output datasets in both examples were obtained by running the unstructured mesh finite element fluid model ($Fluidity$) [44] (referred as the original high fidelity model). In Table 1, the module architecture of the encoder, decoder and discriminator used for two numerical examples are summarized. The dimensional size of the latent state is set to 10 using the trial and error method. The Adam optimizer is employed to optimize the values of parameters (θ, ϕ, ψ), with a learning rate $\eta = 0.0002$. With the training and validated input–output pairs ϑ_{tr} and ϕ_{tr} in two examples, the hybrid VAE-GAN are trained using Algorithm 1.

Algorithm 1 Adversarial training of the hybrid deep adversarial autoencoder.

- Sample $\{\mu^1, \mu^2, \ldots, \mu^{N_x}\}$ from the inputs.
- Sample $\{\hbar_d^1, \hbar_d^2, \ldots, \hbar_d^{N_x}\}$ from the real datasets.

for P = 1 to N_P **do**

for K = 1 to N_K **do**

- Compute the latent state ζ_d in the encoder: $\zeta_d = \mathcal{Q}(\hbar_d, \theta)$.
- Compute the reconstructed outputs $\hbar_d$ in the decoder: $\hat{\hbar}_d = \mathcal{P}(\mathcal{Q}(\hbar_d, \theta), \phi)$.
- Compute the new latent state ζ in the encoder: $\zeta = \mathcal{Q}(\mu, \theta)$.
- Compute the outputs $\hbar$ in the decoder: $\hbar = \mathcal{P}(\mathcal{Q}(\mu, \theta), \phi)$.
- Compute the probability in the discriminator: $\mathcal{D}(\mathcal{Q}(\mu, \theta), \mathcal{Q}(\hbar_d, \theta), \psi)$
- Update ψ in the discriminator by performing its stochastic gradient ascent:

$$\psi \leftarrow \psi + \eta \cdot \nabla_\psi(\mathcal{L}_{rec}(\theta, \phi) + \mathcal{L}_{adv}(\theta, \phi, \psi))$$

end for

- Update θ in the encoder by performing its stochastic gradient descent:

$$\theta \leftarrow \theta - \eta \cdot \nabla_\theta(\mathcal{L}_{rec}(\theta, \phi) + \mathcal{L}_{adv}(\theta, \phi, \psi))$$

- Update ϕ in the decoder by performing its stochastic gradient descent:

$$\phi \leftarrow \phi - \eta \cdot \nabla_\phi(\mathcal{L}_{rec}(\theta, \phi) + \mathcal{L}_{adv}(\theta, \phi, \psi))$$

end for

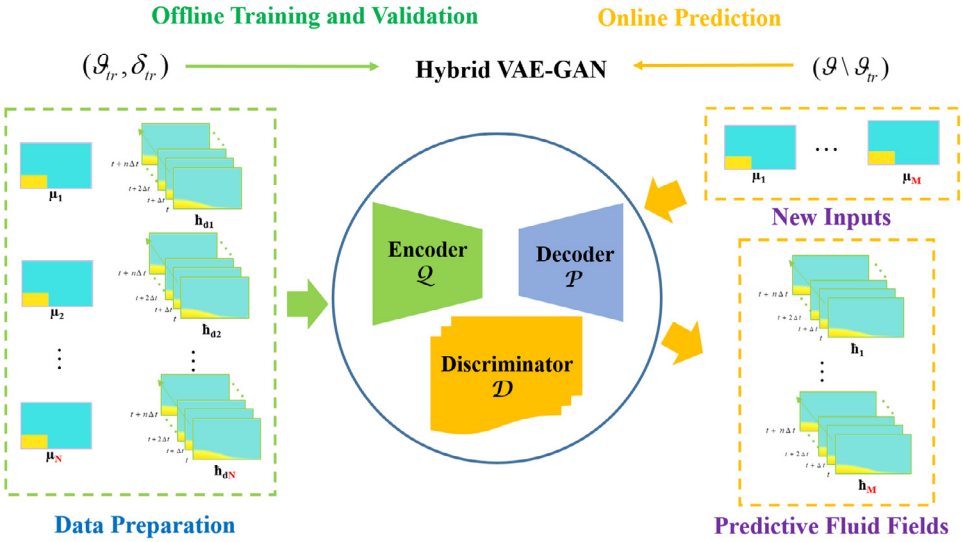

Fig. 3. The offline training and online predictive processes.

6.1. Case 1: flow past a cylinder

The first case for numerical illustration of proposed methods is a two dimensional flow past a cylinder. In Fig. 4 shows a rectangle with the length = 2.2 m and width = 0.41 m, and a cylinder with a radius of 0.05 m. The fluid flow simulations are driven by an inlet velocity from the left boundary of the domain area. The computational

Table 1
Architectures of $\mathcal{Q}$, $\mathcal{P}$, and $\mathcal{D}$.

Module	Case 1: flow past a cylinder		Case 2: water column collapse	
Layer in Encoder $\mathcal{Q}$	Output shape	Parameter number	Output shape	Parameter number
InputLayer	(50, 1, 3213)	0	(75, 1, 19097)	0
$\mathcal{H}^{(E)}$ (leaky ReLU)	(50, 1, 128)	3,701,504	(75, 1, 128)	21,999,872
$\mathcal{H}^{(E)}$ (leaky ReLU)	(50, 1, 16)	18,448	(75, 1, 16)	18,448
$\mathcal{H}^{(E)}$ (leaky ReLU)	(800)	8,010	(1200)	12,010
OutputLayer	(10)	0	(10)	0
Layer in Decoder $\mathcal{P}$	Output shape	Parameter number	Output shape	Parameter number
InputLayer	(10)	0	(10)	0
$\mathcal{H}^{(D)}$ (leaky ReLU)	(50)	550	(75)	825
$\mathcal{H}^{(D)}$ (leaky ReLU)	(800)	40,800	(1200)	91,200
$\mathcal{H}^{(D)}$ (leaky ReLU)	(50, 1, 128)	18,560	(75, 1, 128)	18,560
OutputLayer(Tanh)	(50, 1, 3213)	3,704,589	(75, 1, 19097)	22,018,841
Layer in Discriminator $\mathcal{D}$	Output shape	Parameter number	Output shape	Parameter number
InputLayer	(10)	0	(10)	0
$\mathcal{H}^{(M)}$(leaky ReLU)	(512)	5,632	(512)	5,632
$\mathcal{H}^{(M)}$(leaky ReLU)	(256)	131,328	(256)	131,328
$\mathcal{H}^{(M)}$(leaky ReLU)	(64)	16,448	(64)	16,448
$\mathcal{H}^{(M)}$(leaky ReLU)	(32)	2,080	(32)	2,080
OutputLayer(Sigmoid)	(1)	33	(1)	33

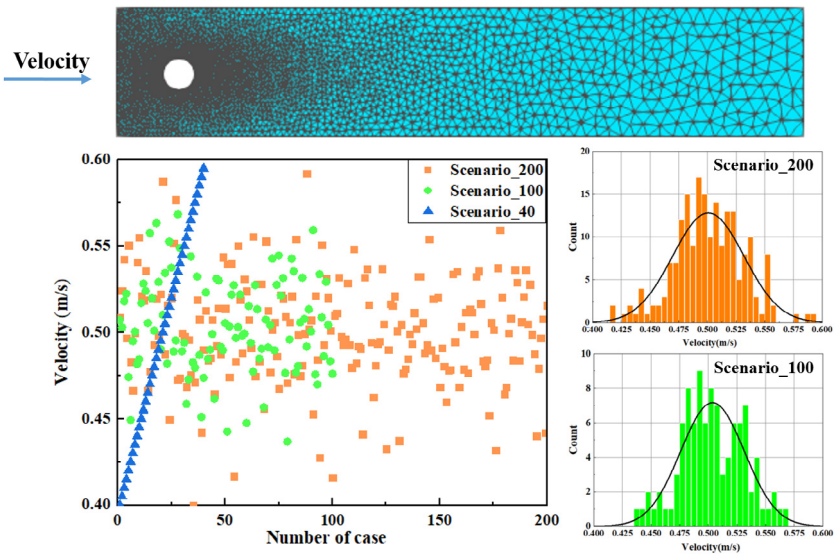

Fig. 4. The sketch of flow past a cylinder and the settings of boundary velocities.

domain consists of 3213 nodes. The whole simulation period was 10 s with a timestep of 0.05 s. The targeted outputs $\hbar_d$, 50 velocity solution snapshots, were obtained by running the original high fidelity model at the regularly spaced time intervals 0.2 s.

6.1.1. Sensitivity analysis

In this example, the inlet velocity is selected as the input parameter (as shown in Fig. 4). To estimate the sensitivity of results from the hybrid VAE-GAN with respect to the number of the input–output datasets, three scenarios have been set up, where 200, 100 and 40 input–output datasets are used for scenario_200, scenario_100 and scenario_40 respectively (shown in Fig. 4). In scenario_200 and scenario_100, the input parameter (inlet velocity) is ranged from 0.4 m/s to 0.6 m/s with the Gaussian distribution $\mathcal{N}(\bar{\mu}, \sigma^2)$, where the mean $\bar{\mu}$ is 0.5 m/s and the

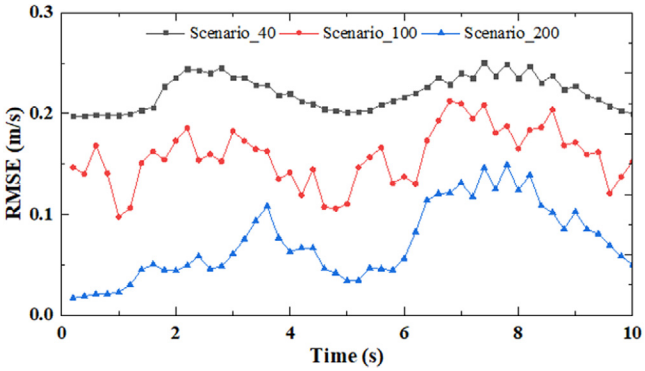

Fig. 5. The RMSE of velocity between the generated fields and original high fidelity fields during whole simulational period.

variance of the distribution σ^2 is 0.001 m/s. In scenario_40, the input data is obtained from the linear distribution between 0.4 m/s to 0.6 m/s.

N_a pairs of input and output datasets for training and validation were obtained by running the high fidelity model. The inputs and targeted outputs ϑ, ϕ can be re-written in a discretized form in space and time:

$$\vartheta = [\mu_a^1, \mu_a^2, \ldots, \mu_a^{N_{b,x}}], \tag{29}$$

$$\phi = \begin{bmatrix} \hbar_{a,t_1}^1 & \hbar_{a,t_1}^2 & \cdots & \hbar_{a,t_1}^{N_x} \\ \hbar_{a,t_2}^1 & \hbar_{a,t_2}^2 & \cdots & \hbar_{a,t_2}^{N_x} \\ \vdots & \vdots & & \vdots \\ \hbar_{a,t_{N_t}}^1 & \hbar_{a,t_{N_t}}^2 & \cdots & \hbar_{a,t_{N_t}}^{N_x} \end{bmatrix}, \tag{30}$$

where $\mu_a \in \mathbb{R}^{N_x}$ represents the input parameter (here, the inlet velocity) while the spatio-temporal distributions of flow variables $\hbar_a \in \mathbb{R}^{N_x}$ (here, the flow velocity) are acted as the outputs, the number of nodes over the domain $N_x = 3213$, the number of nodes at the inlet boundary $N_{b,x} = 39$, the number of time steps $N_t = 50$, and $a \in (1, \ldots, N_a)$, where N_a represents the total number of the input–output pairs used for training, validation and model prediction, $N_a = 200$, 100 and 40 pairs of input–output datasets are setup for scenario_200, scenario_100 and scenario_40 respectively.

Given a new inlet velocity $\mu = 0.52 \in (\vartheta \backslash \vartheta_{tr})$, we evaluate the accuracy of the hybrid VAE-GAN generated with different number of training datasets. The RMSE of the predictive results from the hybrid VAE-GAN and the original high fidelity model is shown in Fig. 5. It can be noticed that the RMSE values in three scenarios are below 0.25 m/s during the whole simulation period. The maximum RMSE values are 0.25 m/s, 0.21 m/s and 0.14 m/s for scenario_40, scenario_100 and scenario_200 respectively. The corresponding relative error values are shown in Fig. 6. We can see that most of the relative errors are below 0.15 during the simulation period except for some moments $t = 2$–3.5 s and $t = 6$–8 s. It is worth noting that the magnitude and trend of relative values in scenario_40 are comparable to those in scenario_100 and scenario_200 except for $t = 2$–3.5 s, although the RMSE values in scenario_40 are higher than other two scenarios. This means that the hybrid VAE-GAN generated with a small number of training datasets could provide a reasonably accurate prediction of nonlinear fluid flows by using the hybrid VAE-GAN.

6.1.2. Model prediction

Fig. 7 displays the velocity solutions (given the inlet velocity $\mu = 0.52 \in (\vartheta \backslash \vartheta_{tr})$) obtained from the hybrid VAE-GAN in scenario_40 and high fidelity model at time levels $t = 1.8, 4, 5.6, 9.2$ s. It is shown that there are small visual differences between the hybrid VAE-GAN and high fidelity model except for upper and lower edges of the domain area. The hybrid VAE-GAN can capture the details of the flow very well in comparison to the high fidelity model. In addition, the magnitude of the spatial velocity is also in closer agreement to the high fidelity model solutions.

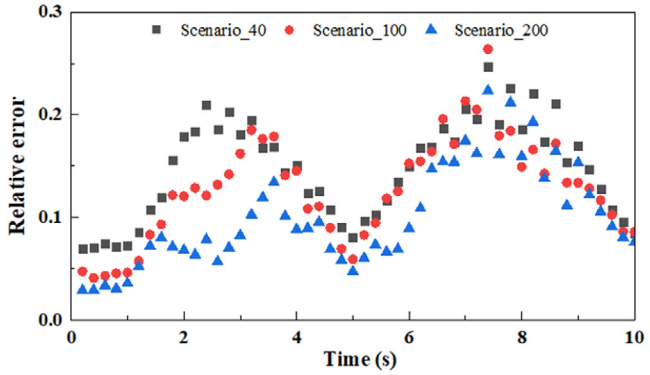

Fig. 6. The relative error of velocity solutions between the generated fields and original high fidelity fields during whole simulational period.

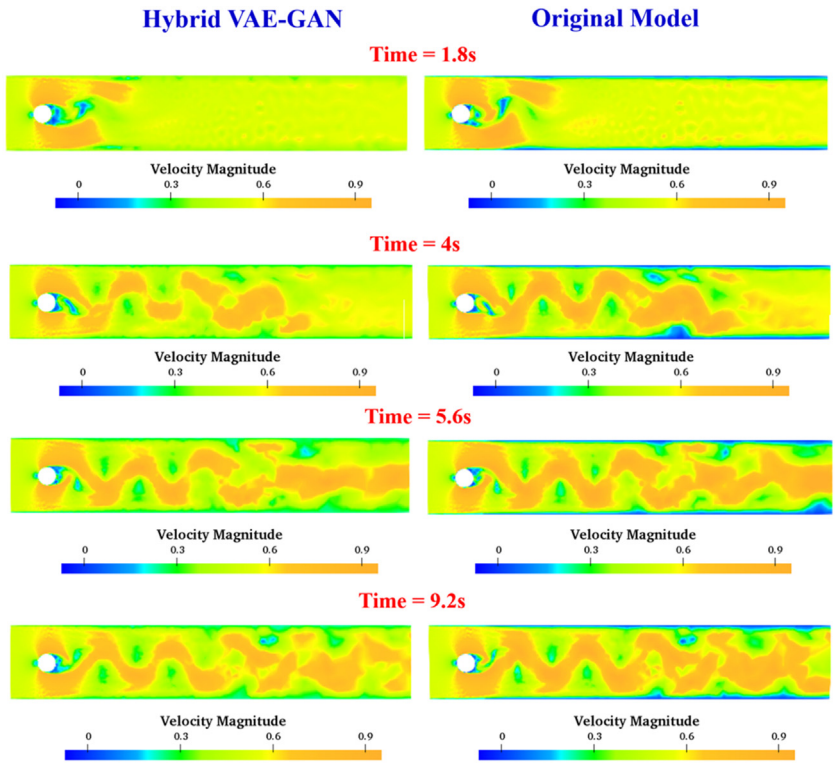

Fig. 7. Comparison of the spatial distribution of velocity fields obtained from the hybrid VAE-GAN (left) and the original high fidelity model (right) at time levels $t = 1.8, 4, 5.6, 9.2$ s.

6.2. Case 2: water column collapse

The hybrid VAE-GAN has been further validated with a case of water column collapse. It is a benchmark test case for multi-material models, also known as a dam break problem. The dam-break flow problem has been of great importance in hydraulic engineering and hydropower generation. Dam-break flows are complex and generally discontinuous with abrupt variations of flow velocity and water depth [45]. The flooding induced by the dam-break flows causes great loss of human life and property as well as damaging the ecosystem in the downstream area.

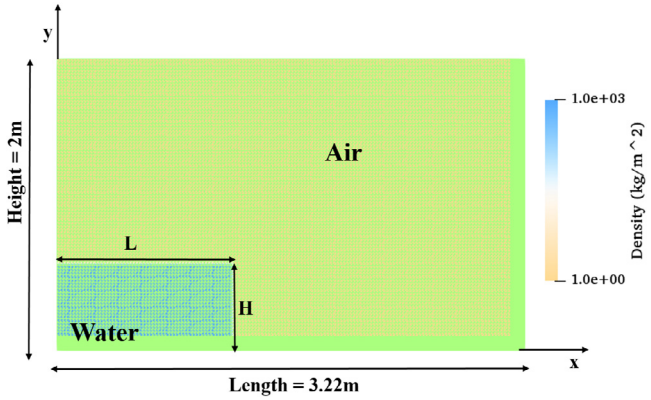

Fig. 8. The sketch of water collapse experiment. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

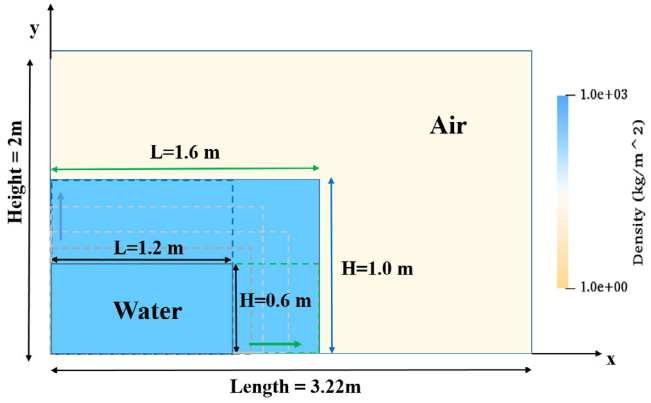

Fig. 9. The settings of length and height in water collapse experiment.

In this example, the dam-break experiment is conducted in a tank with the length 3.22 m, the height 2 m, and the depth 1 m [46]. The simulated reservoir of water is held behind a barrier at one end of the tank. When no variations are introduced in the third dimension, the experiment is reproduced in the horizontal and vertical dimensions within the domain area Ω. Fig. 8 shows the mesh in the domain area Ω. The densities of water and air are $1,000\ \mathrm{kgm}^{-2}$ and $1\ \mathrm{kgm}^{-2}$ respectively. The scalar fields a_k representing the volume fraction are introduced to distinguish the two materials. As shown in Fig. 8, the interface between the water (the yellow area, $a_k = 1$) and air (the blue area, $a_k = 0$) is delineated by contours at a_k of 0.025, 0.5 and 0.975. In this case, the simulation period is [0, 1.85] s, with a timestep = 0.025 s. Thus, the targeted outputs (here, pressure solutions in time and space) $\hbar_d$ were obtained by running the original high fidelity model, with a unstructured mesh of 19097 nodes.

6.2.1. Model application

In this case, for training the hybrid VAE-GAN, the length L and height H of the tank are selected as the input parameters (as shown in Fig. 9). The length of the tank L is ranged from 1.2 m to 1.6 m, and the height H of the tank varies from 0.6 m to 1.0 m. A series of sizes (see Table 2) as inputs μ and the corresponding solution snapshots $\hbar_d$ (a total of 3375 snapshots of 45 pairs of input–output) were obtained by running the high fidelity model. The training and validated input–output pairs ϑ_{tr} and ϕ_{tr} (where $\mu_a \in \mathbb{R}^{N_x}$, $\hbar_{d,a} \in \mathbb{R}^{N_x \times N_t}$, $a \in [0, 39]$, $N_{b,x} = N_x = 75$, $N_t = 19097$ in Eqs. (29) and (30)) are selected for model training and parameter optimization, while the remained inputs $(\vartheta \backslash \vartheta_{tr})$ (where $\mu_a \in \mathbb{R}^{N_x}$, $a \in [0, 6]$, $N_x = 19097$) are used for model prediction.

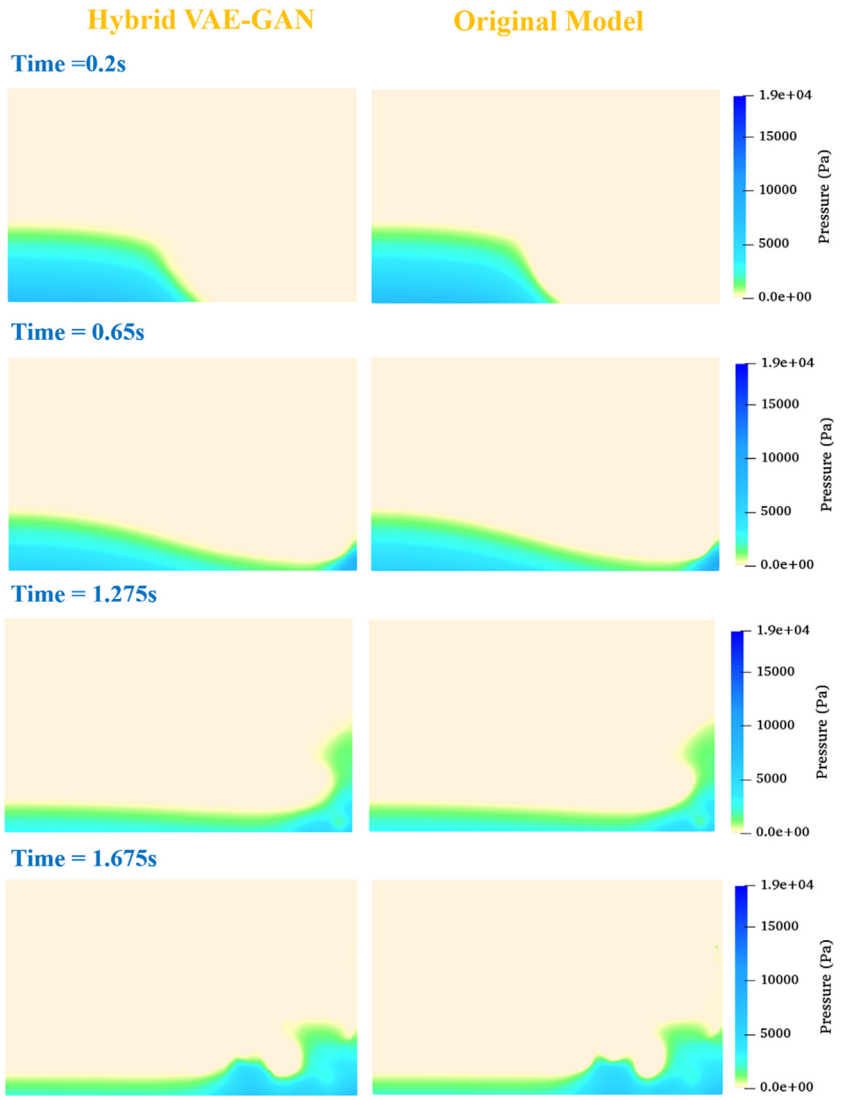

Fig. 10. Comparison of the spatial distribution of pressure fields obtained from the hybrid VAE-GAN (left) and the original high fidelity model (right) at time levels $t = 0.2, 0.65, 1.275, 1.675$ s.

6.2.2. Model prediction

Prediction of spatial nonlinear fluid flows: To evaluate the predictive ability of the hybrid VAE-GAN, given the new input $\mu \in (\vartheta \backslash \vartheta_{tr})$, comparison of the predictive results from the hybrid VAE-GAN and the original high fidelity model are shown in Fig. 10. It can be seen that the hybrid VAE-GAN predicted the flow fields well, which captures the most pressure features at time levels $t = 0.2, 0.65, 1.275, 1.675$ s. Visually, very little difference between the hybrid VAE-GAN and the original high fidelity model can be noticed. In order to compare the differences between the predictive and original fluid fields, the absolute error and correlation coefficient of pressure solutions within the computational domain area Ω are illustrated in Fig. 11. It is observed that the absolute errors are small over the whole domain area at different time levels, and the correlation coefficient between the hybrid VAE-GAN and the high fidelity model is higher than 0.99. The predictive results in spatial space suggest the hybrid VAE-GAN is able to obtain reasonable and accurate solutions in spatial space for nonlinear fluid flows.

Prediction of temporal nonlinear fluid flows: Fig. 12 shows the temporal variation of pressure solution predicted by the two models. Before the water collapse happens, the points P1, P2, P5 and P6 are located in the water area

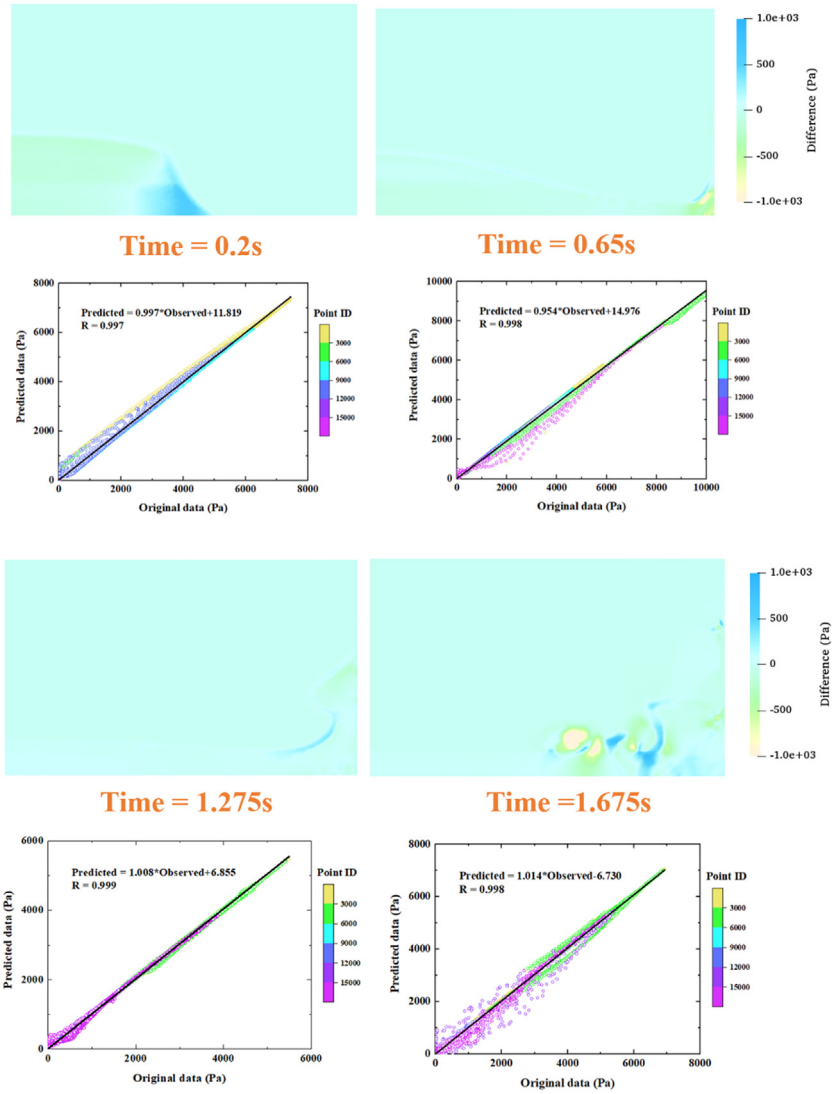

Fig. 11. Differences and correlation coefficients of pressure fields between the hybrid VAE-GAN and the original high fidelity model at time levels $t = 0.2, 0.65, 1.275, 1.675$ s.

(as marked with a blue rectangle in Fig. 9), while other four points P3, P4, P7 and P8 are placed in the area filled with air. Model performance has been evaluated by comparison of pressure variation in these detector points. In Fig. 12(a), (b), (e) and (f), the pressure values at four points are slightly increased and then gradually decreased as the water collapse happens. As for detector P3, it is beyond the scale of water front motions before $t = 1.5$ s, and the pressure trend in Fig. 12(c) is not so obvious. After $t = 1.5$ s, influenced by the overturning water, the pressure at the detector P3 starts to fluctuate. Compared to detector P3, it can be noted that the pressure values at detector P4, P7 and P8 are uprising when the water drops down the horizon into rightward direction. The sequent water experiences dropping, travelling, rising up along the right vertical wall, and dropping again processes. It can be observed that the curves of pressure from the hybrid VAE-GAN achieve a good agreement with those of the original high fidelity model, except for the under-predictions of dropping again process after $t = 1.5$ s. However, it is also a challenging problem for numerical simulation when the water hits the right wall and again meets the horizontal free surface [47].

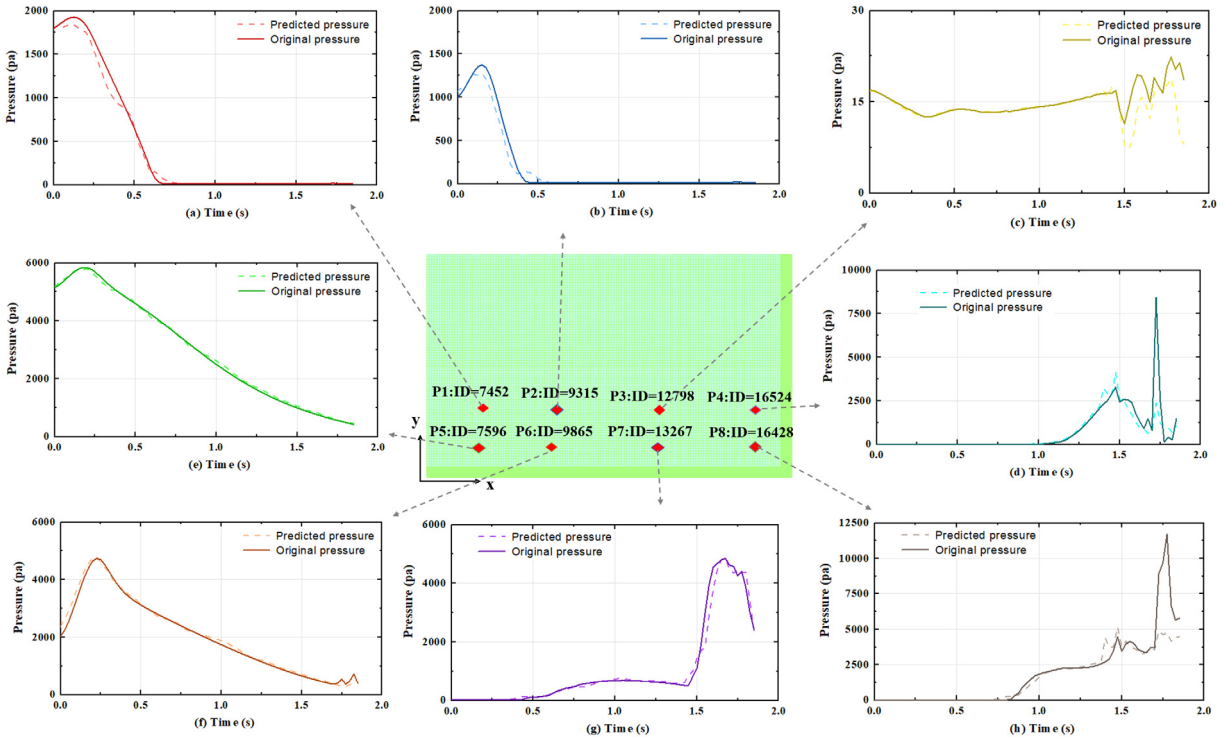

Fig. 12. Comparison of the temporal variation of pressure at points ID = 7452, 7596, 9315, 9865, 12798, 13267, 16428, 16524.

Table 2
The variation of length and height of the tank in cases.

Cases	L(m)	H(m)	Cases	L(m)	H(m)	Cases	L(m)	H(m)
A1	1.2	0.6	A16	1.3	0.9	A31[a]	1.5	0.75
A2	1.2	0.65	A17	1.3	0.95	A32[a]	1.5	0.8
A3	1.2	0.7	A18	1.3	1.0	A33[a]	1.5	0.85
A4	1.2	0.75	A19	1.4	0.6	A34	1.5	0.9
A5	1.2	0.8	A20	1.4	0.65	A35	1.5	0.95
A6	1.2	0.85	A21	1.4	0.7	A36	1.5	1.0
A7	1.2	0.9	A22[a]	1.4	0.75	A37	1.6	0.6
A8	1.2	0.95	A23[a]	1.4	0.8	A38	1.6	0.65
A9	1.2	1.0	A24[a]	1.4	0.85	A39	1.6	0.7
A10	1.3	0.6	A25	1.4	0.9	A40	1.6	0.75
A11	1.3	0.65	A26	1.4	0.95	A41	1.6	0.8
A12	1.3	0.7	A27	1.4	1.0	A42	1.6	0.85
A13	1.3	0.75	A28	1.5	0.6	A43	1.6	0.9
A14	1.3	0.8	A29	1.5	0.65	A44	1.6	0.95
A15	1.3	0.85	A30	1.5	0.7	A45	1.6	1.0

[a]Represents the validated datasets.

To further evaluate the performance of the hybrid VAE-GAN, the correlation coefficient and RMSE are calculated in the temporal-spatio space. The error analysis of pressure solutions in the temporal space is shown in Fig. 13. It is evident that most of RMSE values are between 0 pa and 100 pa and the correlation coefficients values are close to 99% during the whole simulation period except for some moments including $t = 0.5$ s and after $t = 1.5$ s. The water front hits the right wall at $t = 0.5$ s and water drops again towards the left wall after $t = 1.5$ s, which causes obvious errors as discussed above. The maximum value of RMSE is 2102 pa at $t = 1.775$ s, and the corresponding minimum value of correlation coefficient is 96%. In light of fluid field prediction in the temporal space, these results

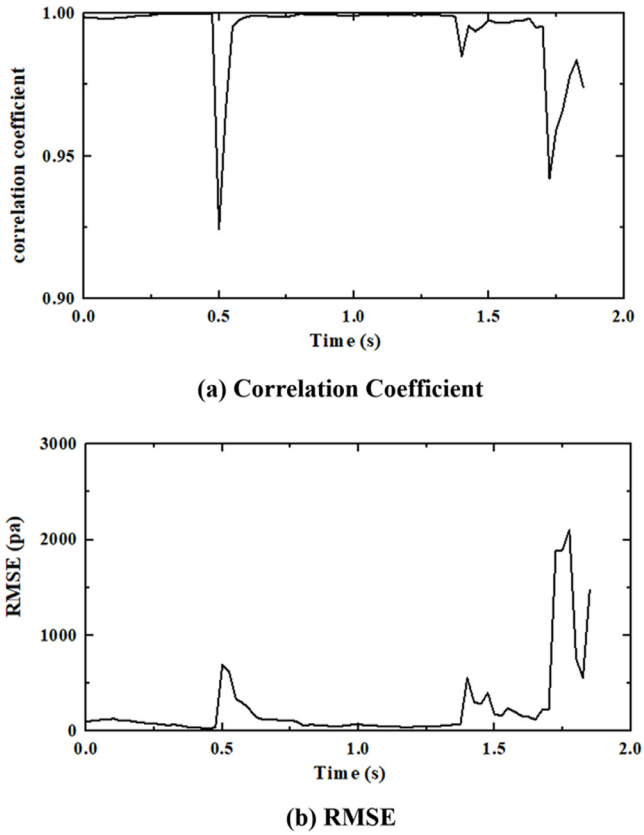

Fig. 13. The correlation coefficients and RMSE of pressure between the generated fields and original high fidelity fields during whole simulational period.

demonstrate that the hybrid VAE-GAN performs well and the predicted fluid fields are in good agreement with the true fluid fields during the whole process.

Fig. 14 illustrates the model performance in the spatial space. It can be noted that most values of RMSE are below 500 pa in the most domain area Ω, except for the area close to the right wall, where the largest errors exist during the process of water rising up along the right vertical wall and maximum value of RMSE is 1940 pa. In addition, it is clear that the values of correlation coefficient in most area exceed 0.99, although the correlation coefficient is negative in some area influenced by the interface of multi-materials. This can be explained that when the water front strikes the right vertical wall, the water jet causes a sudden rise of pressure and air entrainment [47]. The phenomena is shown in Fig. 12(c), where depicts the pressure fluctuation after $t = 1.5$ s. The above results demonstrate the capability of hybrid VAE-GAN to predict accurate fluid flows.

6.3. Model accuracy and efficiency of hybrid VAE-GAN

In this section, the model accuracy and computational cost of the hybrid VAE-GAN are provided for all the case studies. Table 3 shows the online prediction accuracy of the hybrid VAE-GAN at different timesteps. It can be seen that the relative errors are between 0.03 to 0.16 in all the cases.

The simulations of the hybrid VAE-GAN and the original high fidelity model were preformed on Intel Xeon(R) CPU@ 3.60 GHz with a 449.5 GB memory. Table 4 lists the computation time required for the hybrid VAE-GAN and high fidelity model to simulate the flow past a cylinder and water column collapse in the online prediction process, where the CPU time required for the offline training is not included. It can be seen that the CPU time for running the hybrid VAE-GAN is reduced drastically by three orders of magnitude in comparison to the original high fidelity model in two examples.

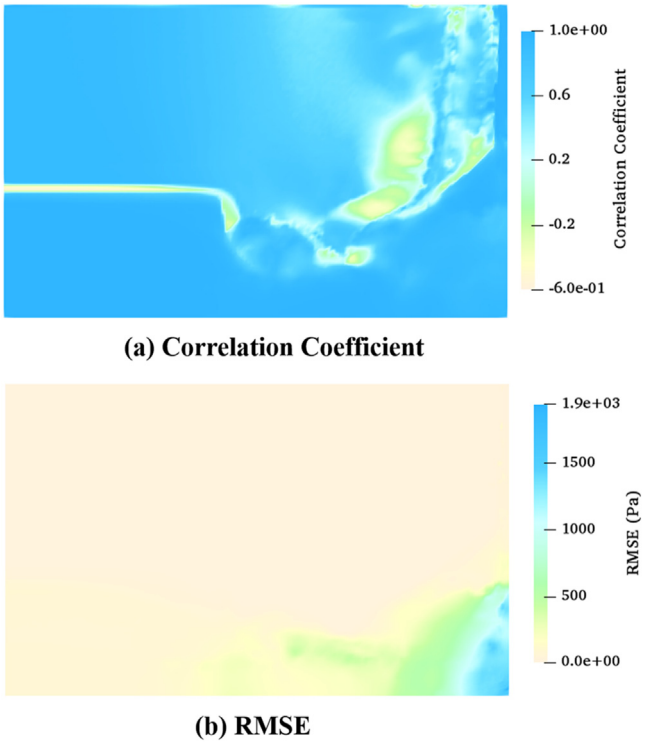

Fig. 14. The correlation coefficients and RMSE of pressure in time series in the computational domain area Ω.

Table 3
The relative error of variables between the generated fields and original high fidelity fields during whole simulational period for all the cases.

Timestep	2	10	25	44
Case 1: flow past a cylinder (time interval = 0.2 s)				
scenario_200	0.03	0.1	0.04	0.11
scenario_100	0.04	0.14	0.05	0.13
scenario_40	0.07	0.14	0.08	0.15
Case 2: water column collapse (time interval = 0.025 s)	0.14	0.09	0.14	0.16

Table 4
Online CPU time required for running the hybrid VAE-GAN and high fidelity model during the whole simulation period for two cases.

Case	Hybrid VAE-GAN (Seconds)	High fidelity model (Seconds)
Case 1: flow past a cylinder	1	427
Case 2: water column collapse	22	8837

7. Conclusions

In this work, a hybrid VAE-GAN method has been, for the first time, used to predict nonlinear fluid flows in varying parameterized space. For any given input parameters μ, the hybrid VAE-GAN is capable of predicting accurately dynamic nonlinear fluid flows and remains highly efficient in simulations since it combines both advantages of VAE and GAN.

The performance of the hybrid VAE-GAN has been demonstrated by a flow past a cylinder test case and a second case of water column collapse. To evaluate the model accuracy, a detailed comparison between the original

high fidelity model (Fluidity) and the hybrid VAE-GAN has been undertaken. The accuracy assessment has also been performed through the correlation coefficient, RMSE and relative error. The numerical simulations show that the hybrid VAE-GAN exhibits good agreement with the original high fidelity model, in both space and time. Additionally, a significant CPU speed-up has been achieved by the hybrid VAE-GAN for all the test cases.

The hybrid VAE-GAN is an efficient and robust tool for simulations of parameterized nonlinear fluid flows. It provides a wide range of applications, for instance, risk response management in emergencies and natural hazards (e.g. flooding, dam break, etc.), real-time decision making. Future work will be focused on the predictive ability for a long lead-time prediction (beyond the trained period) by the hybrid VAE-GAN and more complex fluid problems will also be examined with this model.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

The Authors acknowledge the support of: China Scholarship Council (No. 201806270238) and funding from the EPSRC (MAGIC) (EP/N010221/1) and INHALE (EP/T003189/1), and the Royal Society (IEC/ NS- FC/170563) in the UK, as well as the joint KAUST-Imperial research project (EACPR_P83206).

References

[1] N.B. Erichson, M. Muehlebach, M.W. Mahoney, Physics-informed autoencoders for Lyapunov-stable fluid flow prediction, 2019, arXiv preprint arXiv:1905.10866.
[2] R. Hu, F. Fang, C.C. Pain, I.M. Navon, Rapid spatio-temporal flood prediction and uncertainty quantification using a deep learning method, J. Hydrol. (2019).
[3] K. Taira, M.S. Hemati, S.L. Brunton, Y. Sun, K. Duraisamy, S. Bagheri, S. Dawson, C.-A. Yeh, Modal analysis of fluid flows: Applications and outlook, 2019, arXiv preprint arXiv:1903.05750.
[4] B. Lusch, J.N. Kutz, S.L. Brunton, Deep learning for universal linear embeddings of nonlinear dynamics, Nature Commun. 9 (1) (2018) 4950.
[5] K. Lee, K. Carlberg, Model reduction of dynamical systems on nonlinear manifolds using deep convolutional autoencoders, 2018, arXiv preprint arXiv:1812.08373.
[6] T. Murata, K. Fukami, K. Fukagata, Nonlinear mode decomposition for fluid dynamics, 2019, arXiv preprint arXiv:1906.04029.
[7] S.L. Brunton, B.R. Noack, P. Koumoutsakos, Machine learning for fluid mechanics, Annu. Rev. Fluid Mech. 52 (2020) 477–508.
[8] N. Geneva, N. Zabaras, Quantifying model form uncertainty in Reynolds-averaged turbulence models with Bayesian deep neural networks, J. Comput. Phys. 383 (2019) 125–147.
[9] J. Ling, A. Kurzawski, J. Templeton, Reynolds averaged turbulence modelling using deep neural networks with embedded invariance, J. Fluid Mech. 807 (2016) 155–166.
[10] J.N. Kutz, Deep learning in fluid dynamics, J. Fluid Mech. 814 (2017) 1–4.
[11] F.J. Gonzalez, M. Balajewicz, Deep convolutional recurrent autoencoders for learning low-dimensional feature dynamics of fluid systems, 2018, arXiv preprint arXiv:1808.01346.
[12] Y. Xie, E. Franz, M. Chu, N. Thuerey, Tempogan: A temporally coherent, volumetric gan for super-resolution fluid flow, ACM Trans. Graph. 37 (4) (2018) 95.
[13] B. Kim, V.C. Azevedo, N. Thuerey, T. Kim, M. Gross, B. Solenthaler, Deep fluids: A generative network for parameterized fluid simulations, in: Computer Graphics Forum, vol. 38, Wiley Online Library, 2019, pp. 59–70.
[14] M. Reichstein, G. Camps-Valls, B. Stevens, M. Jung, J. Denzler, N. Carvalhais, et al., Deep learning and process understanding for data-driven Earth system science, Nature 566 (7743) (2019) 195.
[15] A.B. Farimani, J. Gomes, V.S. Pande, Deep learning the physics of transport phenomena, 2017, arXiv preprint arXiv:1709.02432.
[16] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative adversarial nets, in: Advances in Neural Information Processing Systems, 2014, pp. 2672–2680.
[17] M. Cheng, F. Fang, C.C. Pain, I.M. Navon, Data-driven modelling of nonlinear spatio-temporal fluid flows using a deep convolutional generative adversarial network, Comput. Methods Appl. Mech. Engrg. 365 (2020) 113000.
[18] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, H. Lee, Generative adversarial text to image synthesis, 2016, arXiv preprint arXiv:1605.05396.
[19] C. Li, M. Wand, Precomputed real-time texture synthesis with Markovian generative adversarial networks, in: European Conference on Computer Vision, Springer, 2016, pp. 702–716.
[20] P. Isola, J.-Y. Zhu, T. Zhou, A.A. Efros, Image-to-image translation with conditional adversarial networks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 1125–1134.
[21] D.P. Kingma, M. Welling, Auto-encoding variational bayes, 2013, arXiv preprint arXiv:1312.6114.

[22] S. Semeniuta, A. Severyn, E. Barth, A hybrid convolutional variational autoencoder for text generation, 2017, arXiv preprint arXiv:1702.02390.
[23] X. Hou, L. Shen, K. Sun, G. Qiu, Deep feature consistent variational autoencoder, in: 2017 IEEE Winter Conference on Applications of Computer Vision, WACV, IEEE, 2017, pp. 1133–1141.
[24] J. Walker, C. Doersch, A. Gupta, M. Hebert, An uncertain future: Forecasting from static images using variational autoencoders, in: European Conference on Computer Vision, Springer, 2016, pp. 835–851.
[25] Y. Pu, Z. Gan, R. Henao, X. Yuan, C. Li, A. Stevens, L. Carin, Variational autoencoder for deep learning of images, labels and captions, in: Advances in Neural Information Processing Systems, 2016, pp. 2352–2360.
[26] M. Simonovsky, N. Komodakis, Graphvae: Towards generation of small graphs using variational autoencoders, in: International Conference on Artificial Neural Networks, Springer, 2018, pp. 412–422.
[27] A. Roberts, J. Engel, D. Eck, Hierarchical variational autoencoders for music, in: NIPS Workshop on Machine Learning for Creativity and Design, 2017.
[28] S. Latif, R. Rana, J. Qadir, J. Epps, Variational autoencoders for learning latent representations of speech emotion: A preliminary study, 2017, arXiv preprint arXiv:1712.08708.
[29] M. Rosca, B. Lakshminarayanan, D. Warde-Farley, S. Mohamed, Variational approaches for auto-encoding generative adversarial networks, 2017, arXiv preprint arXiv:1706.04987.
[30] Y. Wu, Y. Burda, R. Salakhutdinov, R. Grosse, On the quantitative analysis of decoder-based generative models, 2016, arXiv preprint arXiv:1611.04273.
[31] L. Mescheder, S. Nowozin, A. Geiger, Adversarial variational bayes: Unifying variational autoencoders and generative adversarial networks, in: Proceedings of the 34th International Conference on Machine Learning-Volume 70, JMLR. org, 2017, pp. 2391–2400.
[32] F. Fang, C.C. Pain, I.M. Navon, D. Xiao, An efficient goal-based reduced order model approach for targeted adaptive observations, Internat. J. Numer. Methods Fluids 83 (3) (2017) 263–275.
[33] D. Xiao, C. Heaney, F. Fang, L. Mottet, R. Hu, D. Bistrian, E. Aristodemou, I. Navon, C. Pain, A domain decomposition non-intrusive reduced order model for turbulent flows, Comput. & Fluids 182 (2019) 15–27.
[34] J.-C. Loiseau, S.L. Brunton, Constrained sparse Galerkin regression, J. Fluid Mech. 838 (2018) 42–67.
[35] J.-C. Loiseau, B.R. Noack, S.L. Brunton, Sparse reduced-order modelling: sensor-based dynamics to full-state estimation, J. Fluid Mech. 844 (2018) 459–490.
[36] E. Kaiser, B.R. Noack, L. Cordier, A. Spohn, M. Segond, M. Abel, G. Daviller, J. Östh, S. Krajnović, R.K. Niven, Cluster-based reduced-order modelling of a mixing layer, J. Fluid Mech. 754 (2014) 365–414.
[37] E. Kaiser, M. Morzyński, G. Daviller, J.N. Kutz, B.W. Brunton, S.L. Brunton, Sparsity enabled cluster reduced-order models for control, J. Comput. Phys. 352 (2018) 388–409.
[38] J.N. Kutz, S.L. Brunton, B.W. Brunton, J.L. Proctor, Dynamic Mode Decomposition: Data-Driven Modeling of Complex Systems, SIAM, 2016.
[39] I. Goodfellow, NIPS 2016 tutorial: Generative adversarial networks, 2016, arXiv preprint arXiv:1701.00160.
[40] D.P. Kingma, J. Ba, Adam: A method for stochastic optimization, 2014, arXiv preprint arXiv:1412.6980.
[41] L. Mosser, O. Dubrule, M.J. Blunt, Reconstruction of three-dimensional porous media using generative adversarial neural networks, Phys. Rev. E 96 (4) (2017) 043309.
[42] S. Lee, D. You, Data-driven prediction of unsteady flow over a circular cylinder using deep learning, Bull. Am. Phys. Soc. (2018).
[43] A. Makhzani, J. Shlens, N. Jaitly, I. Goodfellow, B. Frey, Adversarial autoencoders, 2015, arXiv preprint arXiv:1511.05644.
[44] C.C. Pain, A.P. Umpleby, C.R.E. De Oliveira, A.J.H. Goddard, Tetrahedral mesh optimisation and adaptivity for steady-state and transient finite element calculations, Comput. Methods Appl. Mech. Engrg. 190 (29–30) (2001) 3771–3796.
[45] O. Seyedashraf, A. Rezaei, A.A. Akhtari, Dam break flow solution using artificial neural network, Ocean Eng. 142 (2017) 125–132.
[46] Z.Q. Zhou, J.O. De Kat, B. Buchner, A nonlinear 3D approach to simulate green water dynamics on deck, in: Proceedings of the Seventh International Conference on Numerical Ship Hydrodynamics, Nantes, France, 1999, pp. 1–15.
[47] I.R. Park, K.S. Kim, J. Kim, S.H. Van, A volume-of-fluid method for incompressible free surface flows, Internat. J. Numer. Methods Fluids 61 (12) (2009) 1331–1362.