Bazai, H.; Kargar, E.; Mehrabi, M.

Using an encoder-decoder convolutional neural network to predict the solid holdup patterns in a pseudo-2d fluidized bed

Published in:
Chemical Engineering Science

DOI:
10.1016/j.ces.2021.116886

Published: 31/12/2021

Document Version
Peer-reviewed accepted author manuscript, also known as Final accepted manuscript or Post-print

Using an encoder-decoder convolutional neural network to predict the solid holdup patterns in a pseudo-2d fluidized bed

H Bazai[1], E Kargar[2], M Mehrabi[1*]

[1]*Department of Mechanical and Aeronautical Engineering, University of Pretoria, Pretoria, South Africa*

[2]*School of Electrical Engineering, Aalto University, Espoo, Finland*

***Corresponding author**

E-mail addresses: mehdi.mehrabi@up.ac.za

Abstract:

In this paper, the capability of combined use of computational fluid dynamics (CFD) and data-based deep learning to predict fluidized beds' complex behavior without solving transport equations is being examined. A convolutional neural network (CNN) is trained to anticipate fluidized bed volume fraction contours based on the numerical simulations' results and data-based machine learning. The trained CNN receives the first ten frames from the CFD as input and predicts the next frame. This process continues until all the required frames are obtained. The results show the superior spatial learning capability of CNN and how its combination with CFD can reduce the required computational power without compromising the accuracy. This paper also indicates a pathway to CFD simulations' future if deep learning is fully utilized and integrated.

Keywords: CFD, Deep Learning, Fluidized bed, Convolutional Neural Networks

1. Introduction

Gas–solid fluidized beds are widely used in industry, particularly in the petroleum and chemical sectors. Understanding fluidization is of great interest, and a considerable number of studies have been conducted to study fluidization phenomena [1], investigate bubble formation mechanism [2], and flow patterns of fluidized beds [3]. Computational fluid dynamics (CFD) has been employed regularly as an effective technique to study the complex nature of gas-solid fluidized beds [4 - 7]. Two common approaches of CFD modeling of fluidized beds are Eulerian-Lagrangian and Eulerian-Eulerian. In the Eulerian-Lagrangian approach, the gas phase is considered a continuous

phase, while the solid phase is the discrete one. In this approach, Newton's second law of motion is applied to each particle, and the effects of particle interactions and forces of gas-phase are taken into consideration [8].

On the other hand, in the Eulerian-Eulerian approach, gas and solid phases are considered interpenetrating continua. The integral form of continuity, momentum, and thermal energy equations is utilized for both phases and jump conditions of the interface. In this approach, the solid phase is assumed to behave like the fluid phase [9]. Askaripour and Dehkordi [10] modeled a 3D gas-solid fluidized bed with various drag models to compare the bed's expansion ratio and solids' velocity. A wide range of particle size, the bed's static height, and fluidization velocity were investigated in their study. The results showed that the Wen-Yu drag model [11] provides the best prediction of the expansion ratio and solids velocity. Askarishahi ae al. [12] carried out a numerical study on the flow pattern of solid in gas-solid fluidized beds to investigate the effects of particle size and time averaging by using the two-fluid model (TFM) closed by the kinetic theory of granular flows (KTGF). Their results showed that the particle size does not have a significant impact on the solid circulation and its flow pattern. It was also concluded that there is an optimum value for time-sampling, which is far smaller than the characteristic time for solid distribution.

Recently, a handful number of researchers employed machine and deep learning techniques in fluid mechanics, particularly in multi-phase flows [13 - 16]. CFDNet [17] predicts the fluid's primary physical properties, including velocity, pressure, and eddy viscosity, using a single convolutional neural network at its core. Thuerey et al. [18] use U-Net to investigate deep learning models' accuracy for Reynolds-Averaged Navier-Stokes solutions' inference. Tompson et al. [19] proposed a convolutional neural network that leverages the approximation power of deep learning with the precision of standard solvers to obtain fast and highly realistic simulations to solve the incompressible Euler equations using the standard operator splitting method, in which a large sparse linear system with many free parameters must be solved. Zhu et al. [20] used machine learning methods for turbulence modeling in subsonic flows around airfoils and reconstructed a mapping function between the turbulent eddy viscosity and the mean flow variables by neural networks. Ling et al. [21] presented a deep learning method to learn a model for the Reynolds stress anisotropy tensor from high-fidelity

simulation data. Although many researchers employed artificial neural networks (ANN) and deep learning techniques in complex fluid mechanics, no research has been conducted on the frame generation in fluidized beds by coupling CFD and deep learning models. In this paper, the authors tried to close this gap by introducing a combination of CFD and deep learning techniques for frame generation in fluidized beds. In this hybrid system, the CFD generates the first ten time-step particle volume fraction contours. The deep learning model was then applied to generate the next time-steps contour without solving any transport equations. The CFD simulation was done using AnsysFluent, and 20000 frames were generated during the numerical simulation, then an encoder-decoder neural network was employed to train the network with 18000 frames. After the training process concludes, the encoder network gets sequential stacked frames as input and converts them into a latent vector. The decoder network gets the latent vector to predict the next frame. Then the predicted frame and all the previous frames create the new stack of frames and feed it again into the encoder, and this process continues until all the frames are generated.

The CFD simulation was done on high performance computer (HPC) with 240 cores of Intel Xeon with the CPU clock of 2.6 GHz at South African CSIR/DST Centre for High Performance Computing (CHPC) centre. The simulation took 48 hours. For the CNN section, two NVIDIA V100 16 GB with 24 cores at CHPC were used for training. The training time was 10 hours. After training the CNN, the time required to predict each frame (interface time) was 0.10348170042037964 seconds.

2. Numerical Modelling

2.1 Governing equations

In this study, the Eulerian-Eulerian approach is employed to model the fluidized bed. As mentioned before, in this approach, it is assumed that both gas and solid phases are interpenetrating continua. The Navier-Stokes equations representing the particle-particle interactions and the gas-solid interactions were developed for the solid phase. The continuity equations can be written for the gas and particle phases as follows:

$$\frac{\partial}{\partial t}\left(\varepsilon_g \rho_g\right) + \nabla.\left(\varepsilon_g \rho_g \vec{v}\right) = 0 \tag{1}$$

$$\frac{\partial}{\partial t}(\varepsilon_s \rho_s) + \nabla.(\varepsilon_s \rho_s \vec{v}) = 0 \tag{2}$$

and momentum equations for both phases are:

$$\frac{\partial}{\partial t}(\varepsilon_g \rho_g \vec{v}_g) + \nabla.(\varepsilon_g \rho_g \vec{v}_g \vec{v}_g) = -\varepsilon_g \nabla P_g + \nabla.\tau_g + \varepsilon_g \rho_g \vec{g} - F_{gs} \tag{3}$$

$$\frac{\partial}{\partial t}(\varepsilon_s \rho_s \vec{v}_s) + \nabla.(\varepsilon_s \rho_s \vec{v}_s \vec{v}_s) = -\varepsilon_s \nabla P_s + \nabla.\tau_s + \varepsilon_s \rho_s \vec{g} + F_{gs} \tag{4}$$

Where ε, ρ, and $\vec{v}_s$ the volume fraction, density, and velocity, respectively. P denotes pressure, τ denotes stress tensor, and F_{gs} represents the exchange of momentum between phases. F_{gs} can be calculated as $F_{gs} = \beta_{gs}(v_g - v_s)$. The parameter of β_{gs} known as a drag function, plays a crucial role in predicting solid-gas behavior, and Wen & Yu's model [11] was used to calculate this parameter.

The solid phase (particle) is treated as a fluid with adequate transport characteristics. The stress tensor τ_s is defined as follows:

$$\tau_s = (-P_s + \eta \mu_b \nabla.\vec{v}_s)\bar{\bar{I}} + 2\mu_s S_S \tag{5}$$

$$S_S = \frac{1}{2}(\nabla \to v_s + (\nabla \to v_s)^T) - \frac{1}{3}\nabla.\to v_s \bar{\bar{I}} \tag{6}$$

$$\eta = \frac{1 + e_{pp}}{2}, \mu_b = \frac{256}{5\pi}\mu \varepsilon_s^2 g_{0,ss} \tag{7}$$

Where μ_s, μ_b and e_{pp} denote the solid shear viscosity, bulk viscosity, and restitution coefficient, respectively. The solid shear viscosity μ_s model of Agrawal et al. [22] was used to estimate the tangential force. Due to the fluctuations of particle velocity, the equation of motion was coupled with an equation that predicts the pseudo-thermal energy. Finally, to examination, the frictional viscosity, the proposed model of Schaeffer [23] was chosen since it can satisfy the Coulomb yield status at a too high concentration of solids.

2.2 Initial and boundary conditions

The boundary conditions used for the modeling of the fluidized bed are as follows: uniform velocity for gas at the bottom boundary, no-slip boundary for all wall sides with particle-wall restitution of 0.9 and the specularity coefficient of 0.9, and the atmospheric

pressure was selected to allow particles for leaving the bed. The initial conditions of the bed are given in TABLE 1.

2.3 Grid generation

In the numerical simulation, grids size has a significant impact on the results. Therefore, choosing the right size of grids is one of the critical parts of a numerical study. Most studies have been performed on numerical problems that use grid independence to determine the optimum grid size. In the Eulerian-Eulerian approach for modeling fluidized beds, the grids' size depends on the particle size [24]. Therefore, in the present study, to generate the mesh, we selected $\Delta x = \Delta y = 5$ mm and $\Delta z = 3$ mm for each element for the particle size of 250μm. Consequently, the number of nodes for height, width, and depth of the bed are 300, 60, and 5, respectively, and the total number of cells is 90000.

2.4 Numerical simulation method

In the present study, the conservation equations of energy, momentum, mass, turbulence for both phases of gas and solid, and the equation of granular temperature for the solid phase have been solved simultaneously. The other equations explained in the previous section are used to predict the bed's particles' behavior. The finite volume method (FVM) has been adopted to solve the governing equations, and the SIMPLE Coupled algorithm has been manipulated for the pressure-velocity coupling. The first-order implicit method and the second-order upwind method have been implemented for transient and spatial discretization. TABLE 2 provides all the parameters and properties which are required for simulation. The simulation was performed with a time step of 0.0005 seconds, and a fixed CFL number of 1.0 and 40 iterations for each time step were considered to reach the required convergence. The simulation was performed for 15 seconds, and the first 10 seconds were ignored to eliminate the transient conditions, and the rest of the time was considered the time-averaging. Numerical simulation was performed by AnsysFluent software on a high performance computer (HPC) with 240 cores of Intel Xeon with the CPU clock of 2.6 GHz at South African CSIR/DST Centre for High Performance Computing (CHPC) centre. The numerical simulation was done in 48 hours.

2.5 Validation of CFD model:

The bed's particle volume fraction's contour is compared to available X-Ray images of Askaripour and Dehkordi [10] to validate the numerical simulation. FIGURE 1 shows a visual comparison between experimental and numerical generated contours of particle volume fraction with two air inlet velocities of 0.264 (m/s) and 0.936 (m/s). This figure shows a good agreement between the numerically generated contours and the experimental X-ray pictures. Bed expansion ratio (H/H0) is also used to validate numerical simulation results compared to the experimental results. The bed's expansion ratio with the air inlet velocity of 0.264 (m/s) was reported to be 1.38, while it is numerically calculated to be 1.36. The bed's expansion ratio with the air inlet velocity of 0.936 (m/s) was reported to be 2.32, while the numerical simulations reported 2.3. In both cases, the numerical results' expansion ratio is in excellent agreement with the experimental results.

3. Convolutional Neural Network (CNN) and Network Architecture

Convolutional neural networks (CNNs) are currently specialized neural networks for processing data with a known grid-like topology. Examples include time-series data, which can be thought of as a 1-D grid taking samples at regular time intervals, and image data, which can be considered a 2-D grid of pixels. Convolutional networks have been tremendously successful in practical applications. The name "convolutional neural network" indicates that the network employs a mathematical operation called convolution. Convolution is a specialized kind of linear operation. Convolutional networks are simply neural networks that use convolution in place of general matrix multiplication in at least one of their layers [25]. These neural networks are widely used for research related to computer vision and image classification. In this type of neural network, images can be fed directly into the network as input, and the feature extraction process takes place with the use of convolutional layers. These convolutional layers slide a set of randomly initialized independent filters over the images, which transform the images based on the filters' values. These filters that are used to extract patterns from the images can be learned using Gradient Descent. During the training process, filters are adjusted to extract features and generate meaningful information. Hence, the values of the filters are the weights that need to be optimized during the training. By

stacking multiple convolutional layers on top of each other, each layer will earn different input features. The first layers learn to extract simple features like edges, and the later layers extract more abstract and in-depth information. However, the use of multiple convolutional layers leads to an exponential increase in parameters that need to be trained [26].

The model architecture used for future state prediction in this paper is explained here. This problem is approached as a future frame prediction using several sequential frames (states of the system) to predict the next frame (state). Then the predicted state was used as an input to predict the next frame again and so on. The proposed network's schematic is shown in FIGURE 2 and is based on Autoencoder [27] with one encoder and one decoder. The encoder network $f_\theta(z \vee x_{t-T:t})$ receives a sequence of T frames as input $x_{t-T:t}$and encodes them into a low dimensional latent vector z. The decoder network $g_\phi(\hat{x}_{t+1} \vee z)$gets the encoded latent vector z and gives back the next frame $\hat{x}_{t+1}$. Here θ and φ denote the parameters of the encoder and decoder networks, respectively. To find the optimum number of sequential stacked frames, the network was trained for five different input sequence lengths of 2, 3, 5, 10, and 15. TABLE 3 shows the amount of total loss for each case. Even though the result shows that the input sequence length does not significantly impact the total loss, the input sequence length of 10 has the lowest total loss and is used for feeding the encoder. The encoder network gets ten sequential stacked frames as input and converts them into a latent vector, and the decoder network gets the latent vector to predict the next frame. Then the predicted frame and nine previous frames create the new stack of ten frames and feed it again into the encoder, and this process continues until all the frames are generated. To train this encoder-decoder structure and obtain the θ and φ parameters, the negative log-likelihood loss function is used to minimize the error between the next ground truth frame and the predicted frame:

$$L = -E_{f\theta(z|x_{t-T:t})} log\left(g_\phi(\hat{x}_{t+1}|z)\right) \tag{8}$$

The input images are ten sequential frames with a size of 64×64×3, which are concatenated into one tensor of size 64×64×30, and the output is one frame with the size of 64×64×3. The encoder network has five Conv-layers of 4×4 kernel size, stride 2 for all layers except the first one, which is 1, padding 1 for all layers, with 16, 32, 64,

128, and 256 channels. Each Conv-layer is followed by a Batch Normalization layer except the first Conv-layer and Leaky ReLU activation function. The feature map is flattened and followed by a linear layer with Batch Norm and Leaky ReLU activation function to create the latent vector with a dimension of 100. All the Leaky ReLU activation functions have a negative slope of 0.2. The decoder network, the predictor's head, has one fully connected layer followed by Batch Normalization-layer and ReLU activation function and a reshape function to generate the feature-map 256×4×4, and then four Transposed Convolution-layers with stride 2, padding 1, and kernel size 4×4. The Transposed Convolution-layers have 64, 32, 16, and 3 channels followed by Batch Normalization-layer and ReLU activation function except the last one, followed by a Sigmoid activation function. The detailed structure of the generator (both encoder and decoder) used in this paper has shown in FIGURE 3. The Autoencoder network is trained using Adam [28] optimizer with the learning rate of 0.001 and the batch size of 1024 for 500 epochs. The model is implemented in PyTorch [29]. A data set of 20000 frames is collected, and 90 percent of it is used for training and the remaining for testing. Two NVIDIA V100 16 GB with 24 cores at CHPC were used for training the network. The training was done in 10 hours, and the time required to predict each frame (interface time) after the training was 0.10348170042037964 seconds.

4. Results and Discussions:

The deep learning generated frame takes the first ten time-steps of the CFD simulated frames and generates the eleventh frame without solving transport equations. In our proposed model, CFD and deep learning are not used to generate an identical time-step frame. Still, in this section, to show the capability of our proposed hybrid model, both CFD and deep learning were used to generate the contours of particle volume fraction at $t = t_0$, $t = t_0 + 83\Delta t$, $t = t_0 + 95\Delta t$, and $t = t_0 + 170\Delta t$ (time-steps are 0.001 seconds). These frames were compared in FIGURE 3 to FIGURE 6. These figures also help understand how many independent time-steps, the deep learning generated frames, and CFD results are compatible with contours of particle volume fraction in a fluidized bed. FIGURES 4a1, 4a2, and 4a3 illustrate the CFD and the deep learning generated contours of particle volume fraction and the average volume fraction of the particles in different positions in the direction of the x-axis at $t = t_0$, respectively. The results show that the flow patterns are in good agreement with each other, and in both cases, most particles

are amassed on the right side of the column and in the middle part of the bed. The result also shows that for both CFD and deep learning generated frames, the gas phase concentration is higher than that of the solid phase. FIGURES 4b1-4b3 show the numerical simulation results and deep learning developed particle volume fraction frames after 83 time-steps at $t_0 + 83\Delta t$. It is essential to mention that between $t = t_0$, and $t = t_0 + 83\Delta t$, no transport equation has been solved. FIGURE 4b2 is in good agreement with the CFD-generated contour. The bed's expansion height and the particles' accumulation location are approximately similar in both FIGURES of 4b1 and 4b2 even after 83 time-steps. The proposed deep learning model and CFD after 95 time-steps are shown in FIGURES 4c1 to 4c3.

For 95 straight time-steps from $t = t_0$ and $t = t_0 + 95\Delta t$, the deep learning proposed model generated the particle volume fraction contour independent of CFD and without solving any transport equation. Even though FIGURES 4c1 and 4c2 visually show an acceptable agreement, the careful comparison reveals a few deviations in the middle of the bed. FIGURE 4c3 also indicates a divergence in the middle of the bed. It is something that was expected after 95 time-steps of independent frame generation. Finally, the proposed deep learning model and CFD after 170 time-steps are shown in FIGURES 4d1 to 4d3. These figures show that after 170 time-steps, the proposed deep learning model cannot generate correct contours of particle volume fraction in a fluidized bed compared to the result of CFD, especially at a nondimensional length of 0.5. It is easily noticeable that at the nondimensional length of 0.4 to 0.6, an area at the bottom of the bed consists of the gas phase in the CFD-generated contour.

In contrast, there is none in the deep learning generated ones. The result demonstrates that the proposed deep learning model can predict fluidized beds' behavior and generate future frames independent of CFD without solving any transport equation. It is observed that the error gradually increases with the increase in the number of time-steps. It is also shown that there is an optimal time-step that the proposed deep learning model cannot generate the frame independently and with acceptable accuracy after that time-step. That is why generating frames using the proposed deep learning model stops, and the model automatically uses CFD for the successive frames. When CFD generates enough frames, the deep learning model will start again to generate frames, and this process continues until all required frames are generated.

5. Conclusion

This study illustrates the high capability of a deep learning-based model to predict the complex behavior of fluidized bed and independent frame generation of particle volume fraction quicker than CFD with less computational load due to not solving transport equations. Our proposed deep learning model is first trained using the CFD generated frames from $t = 0$ to $t = t_0$. After completing the training process, the proposed deep learning model becomes independent of CFD and can generate the next time-step frame. The proposed CNN can predict the next time-step frame at $t = t_0 + \Delta t$ quicker than CFD. It is because the proposed CNN requires less computational power after being trained. It is also related to the fact that the proposed CNN generates the particle volume fraction contour without solving any transport equation and using computer vision.

Acknowledgments

South African CSIR/DST Centre for High Performance Computing (CHPC) support and resources are gratefully acknowledged.

References:

[1] Oke O, Lettieri P, Mazzei L. An investigation on the mechanics of homogeneous expansion in gas-fluidized beds. Chemical Engineering Science 2015; 127:95–105. https://doi.org/10.1016/j.ces.2015.01.020.

[2] Olowson PA, Almstedt AE. Hydrodynamics of a bubbling fluidized bed: influence of pressure and fluidization velocity in terms of drag force. Chemical Engineering Science 1992; 47:357–66. https://doi.org/10.1016/0009-2509(92)80026-9.

[3] Samuelsberg A, Hjertager BH. An experimental and numerical study of flow patterns in a circulating fluidized bed reactor. International Journal of Multiphase Flow 1996; 22:575–91. https://doi.org/10.1016/0301-9322(95)00080-1.

[4] Yue Y, Zhang C, Shen Y. CFD-DEM model study of gas-solid flow in a spout fluidized bed with an umbrella-like baffle. Chemical Engineering Science 2021; 230:116234. https://doi.org/10.1016/j.ces.2020.116234.

[5] Namdarkedenji R, Hashemnia K, Emdad H. Effect of flow pulsation on fluidization degree of gas-solid fluidized beds by using coupled CFD-DEM. Advanced Powder Technology 2018;29:3527–41. https://doi.org/10.1016/j.apt.2018.09.033.

[6] Bellan S, Matsubara K, Cho HS, Gokon N, Kodama T. A CFD-DEM study of hydrodynamics with heat transfer in a gas-solid fluidized bed reactor for solar thermal applications. International Journal of Heat and Mass Transfer 2018; 116:377–92. https://doi.org/10.1016/j.ijheatmasstransfer.2017.09.015.

[7] Amiri Z, Movahedirad S. Bubble-induced particle mixing in a 2-D gas-solid fluidized bed with different bed aspect ratios: A CFD-DPM study. Powder Technology 2017; 320:637–45. https://doi.org/10.1016/j.powtec.2017.07.097.

[8] Patankar NA, Joseph DD. Modeling and numerical simulation of particulate flows by the Eulerian-Lagrangian approach. International Journal of Multiphase Flow 2001; 27:1659–84. https://doi.org/10.1016/s0301-9322(01)00021-0.

[9] Meier HF, Mori M. Gas-solid flow in cyclones: The Eulerian-Eulerian approach. Computers & Chemical Engineering 1998; 22: S641–4. https://doi.org/10.1016/s0098-1354(98)00114-8.

[10] Askaripour H, Molaei Dehkordi A. Simulation of 3D freely bubbling gas–solid fluidized beds using various drag models: TFM approach. Chemical Engineering Research and Design 2015; 100:377–90. https://doi.org/10.1016/j.cherd.2015.05.041.

[11] Wen CY, Yu YH. A generalized method for predicting the minimum fluidization velocity. AIChE J 1966; 12:610–2. https://doi.org/10.1002/aic.690120343.

[12] Askarishahi M, Salehi M-S, Molaei Dehkordi A. Numerical investigation on the solid flow pattern in bubble in gas–solid fluidized beds: Effects of particle size and time averaging. Powder Technology 2014; 264:466–76. https://doi.org/10.1016/j.powtec.2014.05.061.

[13] Poletaev I, Tokarev MP, Pervunin KS. Bubble patterns recognition using neural networks: Application to the analysis of a two-phase bubbly jet. International Journal of Multiphase Flow 2020; 126:103194. https://doi.org/10.1016/j.ijmultiphaseflow.2019.103194.

[14] Cerqueira RFL, Paladino EE. Development of a deep learning-based image processing technique for bubble pattern recognition and shape reconstruction in dense bubbly flows. Chemical Engineering Science 2021; 230:116163. https://doi.org/10.1016/j.ces.2020.116163.

[15] Bao H, Feng J, Dinh N, Zhang H. Computationally efficient CFD prediction of bubbly flow using physics-guided deep learning. International Journal of Multiphase Flow 2020; 131:103378. https://doi.org/10.1016/j.ijmultiphaseflow.2020.103378.

[16] Zhang Y, Jiang M, Chen X, Yu Y, Zhou Q. Modeling of the filtered drag force in gas–solid flows via a deep learning approach. Chemical Engineering Science 2020; 225:115835. https://doi.org/10.1016/j.ces.2020.115835.

[17] Obiols-Sales O, Vishnu A, Malaya N, Chandramowlishwaran A, CFDNet: a deep learning-based accelerator for fluid simulations, arXiv: 2005.04485.

[18] Thuerey N, Weissenow K, Prantl L, Hu X, Deep Learning Methods for Reynolds-Averaged Navier-Stokes Simulations of Airfoil Flows, arXiv:1810.08217v3.

[19] Tompson J, Schlachter K, Sprechmann P, Perlin K, Accelerating Eulerian Fluid Simulation with Convolutional Networks, arXiv:1607.03597v6.

[20] Zhu L, Zhang W, Kou J, Liu Y. Machine learning methods for turbulence modeling in subsonic flows around airfoils. Physics of Fluids 2019; 31:015105. https://doi.org/10.1063/1.5061693.

[21] Ling J, Kurzawski A, Templeton J. Reynolds averaged turbulence modelling using deep neural networks with embedded invariance. J Fluid Mech 2016; 807:155–66. https://doi.org/10.1017/jfm.2016.615.

[22] Agrawal K, Loezos PN, Syamlal M, Sundaresan S. The role of meso-scale structures in rapid gas–solid flows. J Fluid Mech 2001; 445:151–85. https://doi.org/10.1017/s0022112001005663.

[23] Schaeffer DG. Instability in the evolution equations describing incompressible granular flow. Journal of Differential Equations 1987; 66:19–50. https://doi.org/10.1016/0022-0396(87)90038-6.

[24] Cloete S, Zaabout A, Johansen ST, van Sint Annaland M, Gallucci F, Amini S. The generality of the standard 2D TFM approach in predicting bubbling fluidized bed hydrodynamics. Powder Technology 2013; 235:735–46. https://doi.org/10.1016/j.powtec.2012.11.041.

[25] Goodfellow I, Bengio Y, Courville A. Deep Learning. The MIT Press, 2016.

[26] Zaccone G, Karim R. Deep Learning with TensorFlow: Explore neural networks and build intelligent systems with Python- Second Edition. Packt publication, 2018.

[27] Baldi P, Autoencoders, Unsupervised Learning, and Deep Architectures, Proceedings of ICML Workshop on Unsupervised and Transfer Learning, JMLR Workshop and Conference Proceedings 27:37-49, 2012.

[28] Kingma DP, Ba J. Adam: A Method for Stochastic Optimization, arXiv: 1412.6980v9.

[29] Paszke A, Gross S, Massa F, Lerer A, Bradbury J, Chanan G, Killeen T, Lin Z, Gimelshein N, Antiga L, Desmaison A, Kopf A, Yang E, DeVito Z, Raison M, Tejani A,

Chilamkurthy S, Steiner B, Fang L, Bai J, Chintala S, Pytorch: An imperative style, high-performance deep learning library, arXiv:1912.01703.

TABLE1. Initial conditions of the bed

Freeboard section	Minimum fluidization condition
$\varepsilon_g = 1$	$\varepsilon_g = \varepsilon_{mf}$
$v_{s,x} = 0, v_{s,y} = 0, v_{s,z} = 0$	$v_{s,x} = 0, v_{s,y} = 0, v_{s,z} = 0$
$v_{g,x} = 0, v_{g,y} = v_{mf}, v_{g,z=0}$	$v_{g,x} = 0, v_{g,y} = v_{mf}/\varepsilon_{mf}, v_{g,z=0}$

TABLE 2. Required parameters and properties for CFD modeling

Parameter	Value
Particle density (kg.m^{-3})	2500
Particle sizes (μm)	250
Particle-wall restitution coefficient	0.9
particle-particle restitution coefficient	0.9
Gas density (kg.m^{-3})	1.225
Gas viscosity (Pa.s)	1.789×10^{-5}
Column height (m)	1.5
Column width (m)	0.3
Column depth (m)	0.015
Specularity coefficient	0.5
The angle of internal friction (◦)	30
Initial bed voidage fraction	0.4
Maximum packing limit	0.63

TABLE 3. Total loss for different input sequncence lengths

Input sequence length	2	3	5	10	15
Total Loss	0.2513	0.2506	0.2489	0.2406	0.2473

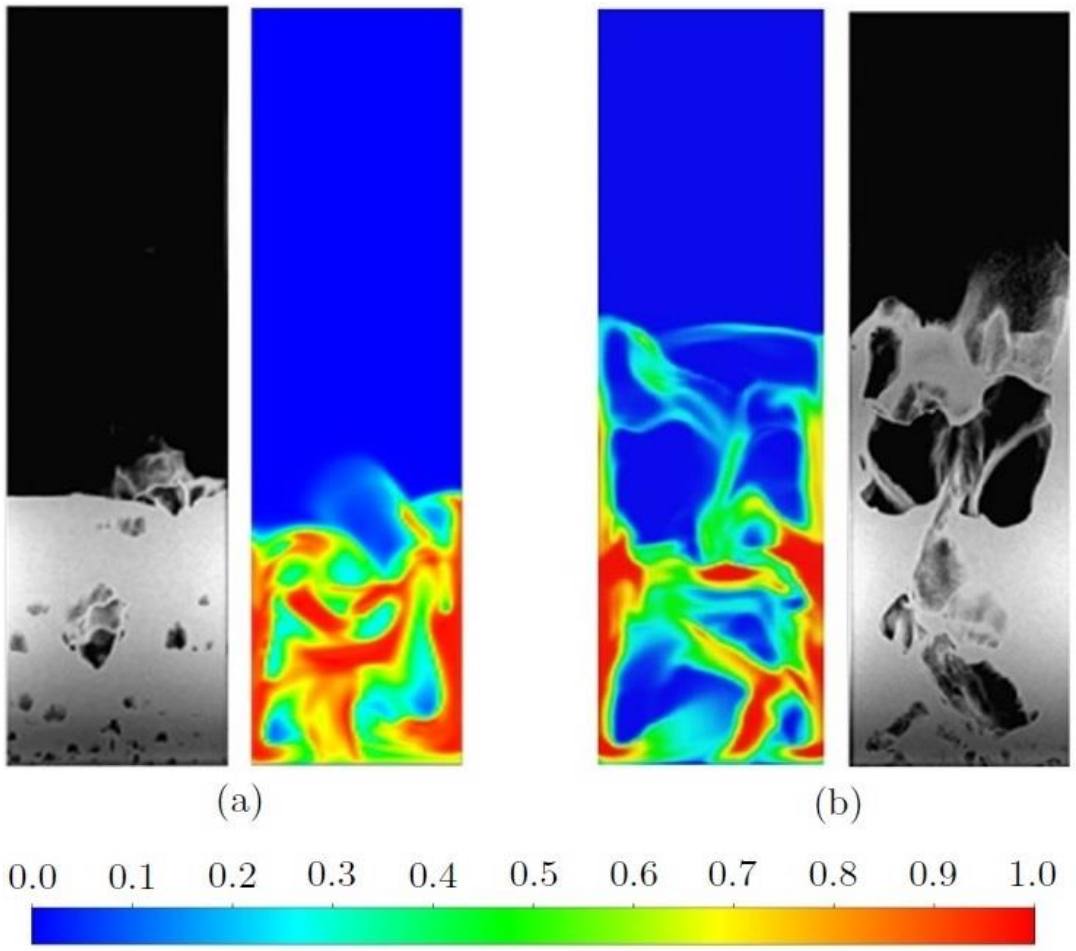

Figure 1. Comparison between CFD and experimental results for two different inlet velocities (a) 0.264 m/s, and (b) 0.936 m/s

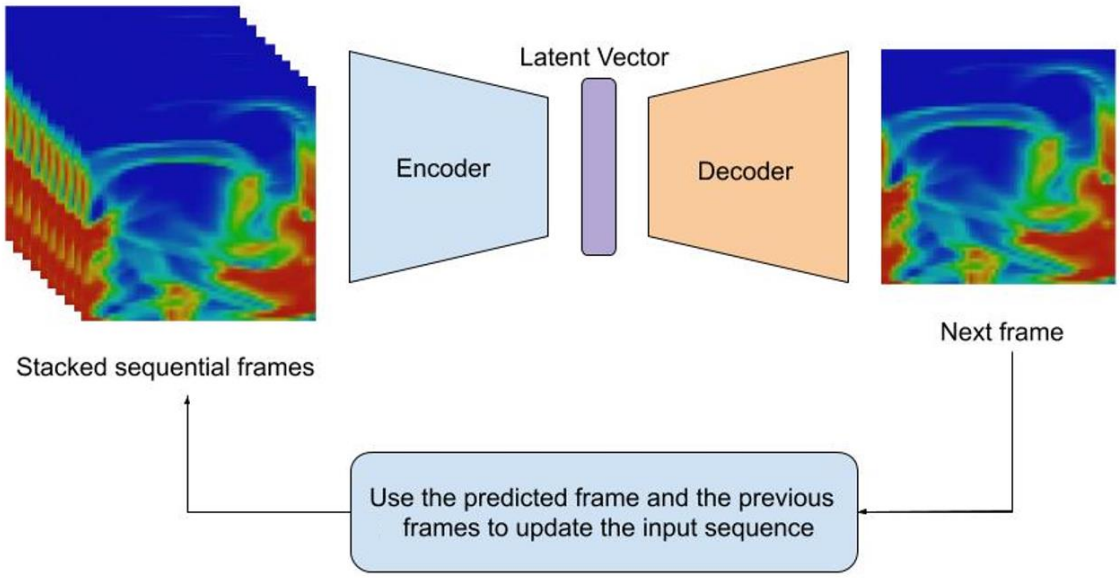

FIGURE 2. Convolutional autoencoder architecture used in current study

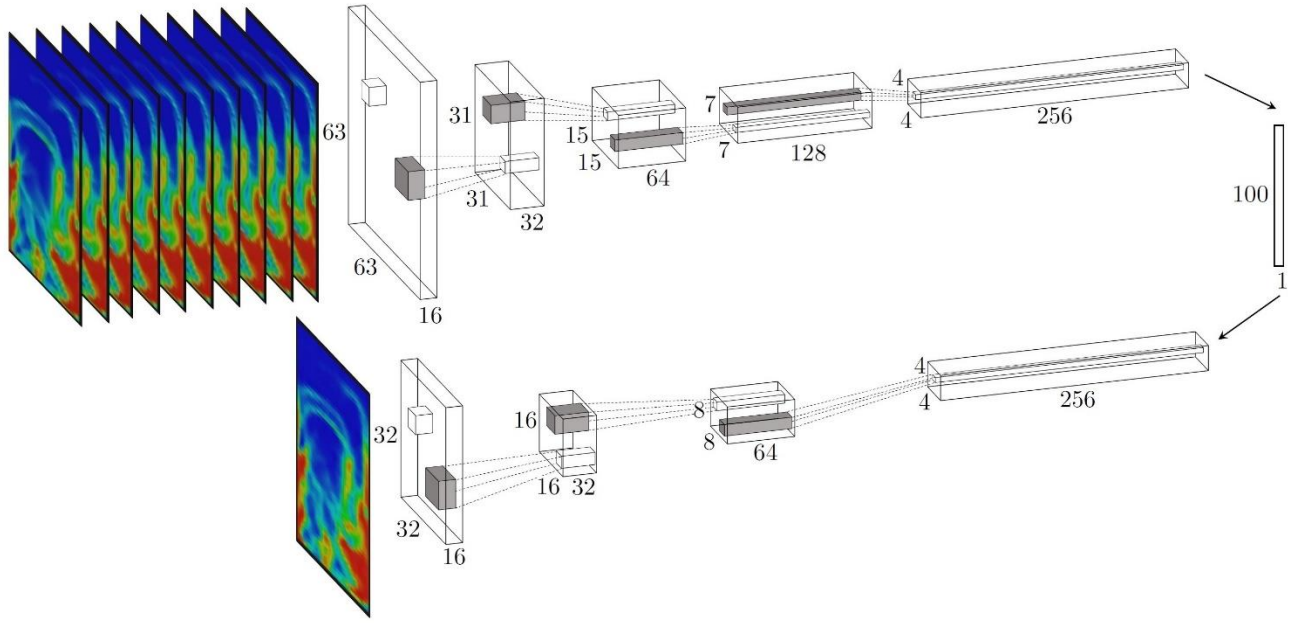

FIGURE 3. Detailed structures of the proposed CNN based encoder-decoder

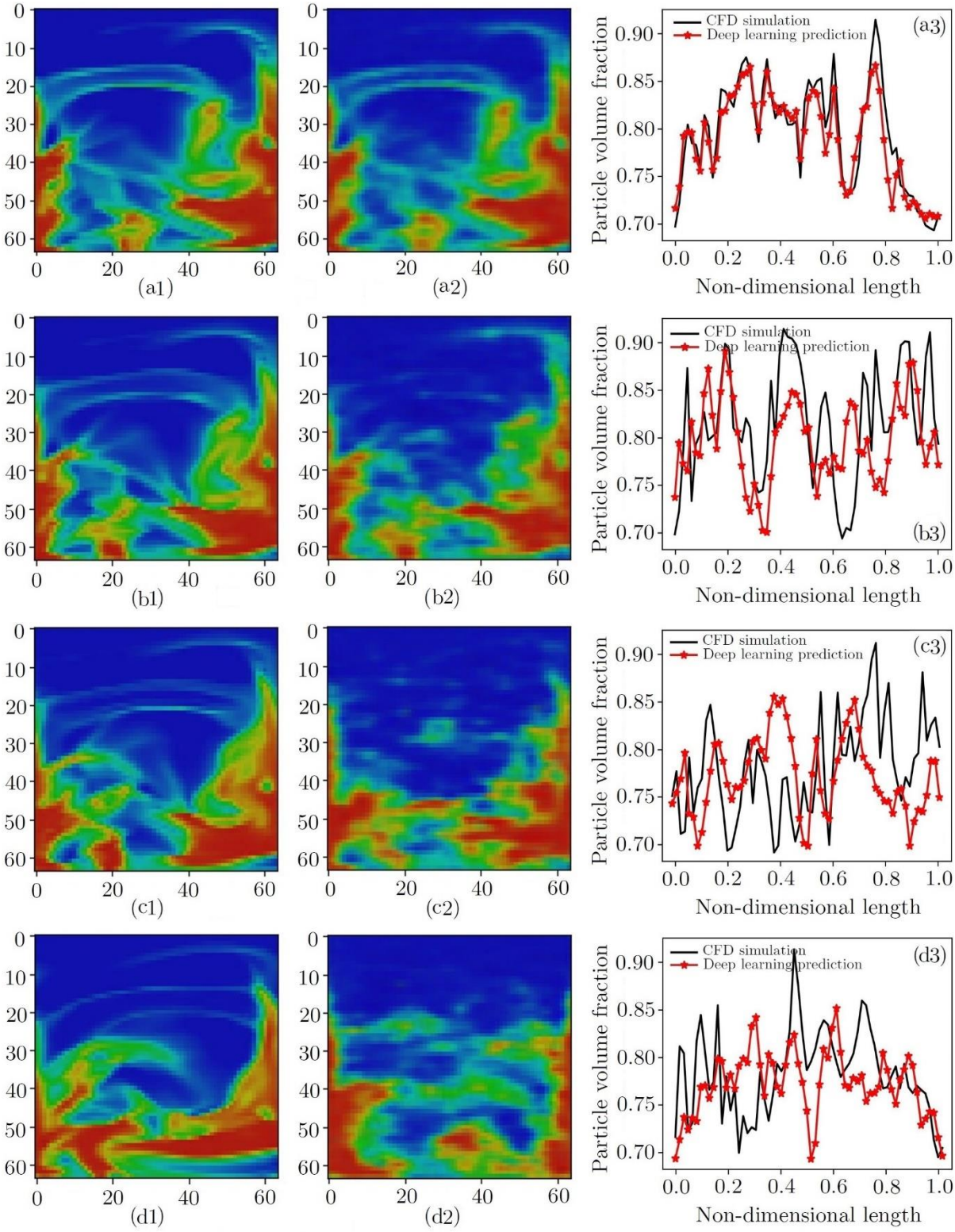

FIGURE 4. (a1) CFD generated frame at t = t_0 (a2) deep learning generated frame at t = t_0 (a3) the average volume fraction of the particles in different positions in the direction of the x-axis at t = t_0 (b1) CFD generated frame at t = t_0 + 83Δt (b2) deep learning generated frame at t = t_0 + 83Δt (b3) the average volume fraction of the particles in different positions in the direction of the x-axis at t = t_0 + 83Δt (c1) CFD generated frame

at $t = t_0 + 95\Delta t$ (c2) deep learning generated frame at $t = t_0 + 95\Delta t$ (c3) the average volume fraction of the particles in different positions in the direction of the x-axis at $t = t_0 + 95\Delta t$ (d1) CFD generated frame at $t = t_0 + 150\Delta t$ (d2) deep learning generated frame at $t = t_0 + 150\Delta t$ (d3) the average volume fraction of the particles in different positions in the direction of the x-axis at $t = t_0 + 150\Delta t$.