Computer Modeling in Engineering & Sciences

Tech Science Press

DOI: 10.32604/cmes.2022.022088

ARTICLE

Image Representations of Numerical Simulations for Training Neural Networks

Yiming Zhang[1,*], Zhiran Gao[1], Xueya Wang[1] and Qi Liu[2]

[1]School of Civil and Transportation Engineering, Hebei University of Technology, Tianjin, 300401, China

[2]School of Computer and Software, Nanjing University of Information Science & Technology, Nanjing, 210044, China

*Corresponding Author: Yiming Zhang. Email: yiming.zhang@hebut.edu.cn

Received: 20 February 2022 Accepted: 15 April 2022

ABSTRACT

A large amount of data can partly assure good fitting quality for the trained neural networks. When the quantity of experimental or on-site monitoring data is commonly insufficient and the quality is difficult to control in engineering practice, numerical simulations can provide a large amount of controlled high quality data. Once the neural networks are trained by such data, they can be used for predicting the properties/responses of the engineering objects instantly, saving the further computing efforts of simulation tools. Correspondingly, a strategy for efficiently transferring the input and output data used and obtained in numerical simulations to neural networks is desirable for engineers and programmers. In this work, we proposed a simple image representation strategy of numerical simulations, where the input and output data are all represented by images. The temporal and spatial information is kept and the data are greatly compressed. In addition, the results are readable for not only computers but also human resources. Some examples are given, indicating the effectiveness of the proposed strategy.

KEYWORDS

Numerical simulations; neural network; pre-/post-processing; data compression

1 Introduction

With the recent developments of machine learning algorithms, frameworks and systems, numerous Artificial Neural Networks (ANNs) have been proposed, built and adopted rapidly and widely in engineering applications. Neural networks can be driven by mechanisms or data. The first type can be represented by the Physical-Informed Neural Network (PINN) [1,2], which uses control equations (commonly in the form of partial differential equations) for building objective and loss functions [3–5] and then finds the optimal solution in the approximation space [6]. They are powerful tools for solving problems that are numerically unstable and time consuming for conventional methods such as finite element and mesh-free methods [7–9]. The second type on the contrary are the neural networks driven by labeled data. The data can be obtained by on-site sensors, experiments and numerical simulations. In many cases, the knowledge behind the phenomena described by these data is unclear for the networks. On the one hand, the interpretability of such networks is unsatisfactory. On the other hand, these networks may help to reveal new patterns, rules, and knowledge by "learning" [10]. Except for tools learning new patterns, data driven neural network can also be considered as a surrogate tool or a

This work is licensed under a Creative Commons Attribution 4.0 International License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

hierarchy model [11], as illustrated in Fig. 1, taking the engineering design process as an example. In Fig. 1, the blue arrows belong to the conventional design process and the red arrows belong to the design process augmented by data driven machine learning models. The green arrows belong to both. Once the machine learning models are trained, the large input-output database from parameter studies is unnecessary when the procedures in the frame can work independently and efficiently.

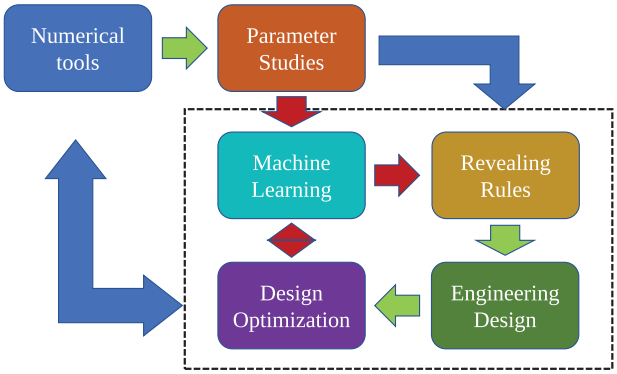

Figure 1: Date driven machine learning models in the engineering design process

Furthermore, engineering structures can be relatively large across space and time and the amount of data from on-site sensors and experiments, especially spatial data, is generally insufficient. In addition, these data can deviate considerably because of errors from monitoring or testing. In contrast, the quantity and quality of data from numerical simulations can be assured. Hence, first validating the numerical model by comparing the results to the experimental and monitored results and then training the neural network with numerically obtained data can be an advantageous procedure.

Basic data driven neural networks are sequential learning models. There are input and output datasets, between which the structures of the neurons can be assembled and built in the platforms associated with TensorFlow [12] and PyTorch. The lower bound of the number of datasets is problem dependent. Except for the design of neural networks, methods for efficiently transferring the datasets from numerical simulations to neural networks are necessary. In this work, considering the advantages of modern neural networks on graphic processing, we propose an image representation method of numerical simulations for training neural networks. The main features of the proposed method include:

- The method is easy to follow and can be implemented into the pre- and pro-processing parts of numerical tools such as those built in the finite element method (FEM) framework.
- The input images naturally take into account the spatial information of the cases, which can be understood by not only neural networks but also human resources.
- The sizes of the images can be further compressed/decompressed by other models such as autoencoders.

Some examples will be provided to show the flexibility and effectiveness of the method. Moreover, we want to emphasize here that some procedures we proposed in this work could be very basic and natural for researchers working in computational mechanics, who follow similar rules for pre- and pro-processing during programming and computing for a long time. Nevertheless, we believe the method can be inspiring and helpful for researchers and engineers working in other fields such as computer

science, and civil and mechanical engineering, which is the main motivation of this work. In the next section, we will provide basic rules and examples together with which the procedures are clarified.

2 Method and Examples

2.1 Basics

We focus on 2D images and 2D simulations (planer or 1D transient cases) in this work, but the ideas can be extended to higher dimensional cases by using a series of continuous images/animations. Considering RGB images, every pixel has channels of three colors: red, green and blue $RGB = [R_{value}, G_{value}, B_{value}]$. The value of each channel is between 0 and 255. A neural network was used for recognizing different compositions of heterogeneous materials represented by RGB images in [13], indicating that the information of RGB images including the RGB values as well as the pixel position can be properly transferred to neural networks. Herein, we take the numerical simulations conducted in the finite element framework as an example. The discretized domain is composed of elements, and each element has its own material and geometric properties. To ensure the performance of the neural network, only testable parameters can be considered as input parameters while the internal variables should not.

2.2 Mechanical Responses of Matrix-Inclusion Material

The mechanical responses of matrix-inclusion materials are basic numerical simulations for composites, such as concrete, rocks and polymers. The model and mesh are shown in Fig. 2. The model will be loaded considering different boundary conditions including compression and shearing. As mentioned before, after large number of simulation results are obtained and transferred to neural network for training, the trained neural network can play the role of a database, which can provide mechanical responses of similar composites subjected to similar loading conditions.

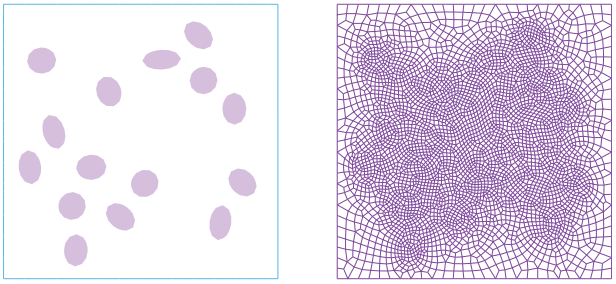

Figure 2: The model and mesh of the matrix-inclusion material

When the matrix and inclusion are isotropic and linear elastic, basic material properties include elastic modulus and Poisson's ratio, represented by red and green channels as Eq. (1)

$$R_{value} = 255 \left(\frac{E - E_{lw}}{E_{up} - E_{lw}} \right),$$
$$G_{value} = 255 \frac{\nu}{0.5}, \tag{1}$$

where $(\cdot)_{lw}$ and $(\cdot)_{up}$ are the lower and upper bounds of the corresponding parameters, respectively. The displacements along the x and y directions regarding isotropic and linear elastic conditions can

be used for setting loading conditions, represented by red and green channels as Eq. (2)

$$R_{value} = 255\left(\frac{u_x - u_{x,lw}}{u_{x,up} - u_{x,lw}}\right),$$
$$G_{value} = 255\left(\frac{u_y - u_{y,lw}}{u_{y,up} - u_{y,lw}}\right). \tag{2}$$

Meanwhile, the stress tensor $\sigma = \left[\sigma_x, \sigma_y, \tau_{xy}\right]^T$ is the output parameter, occupying only three channels as Eq. (3)

$$R_{value} = 255\left(\frac{\sigma_x - \sigma_{x,lw}}{\sigma_{x,up} - \sigma_{x,lw}}\right),$$
$$G_{value} = 255\left(\frac{\sigma_y - \sigma_{y,lw}}{\sigma_{y,up} - \sigma_{y,lw}}\right),$$
$$B_{value} = 255\left(\frac{\tau_{xy} - \tau_{xy,lw}}{\tau_{xy,up} - \tau_{xy,lw}}\right). \tag{3}$$

It can be found that transforming the input/output parameters into RGB figures is a normalization step, which shall be done anyway for neural networks.

Considering the elastic modulus of the matrix and inclusion as 20 and 80 GPa respectively and the Poisson's ratios of the matrix and inclusion as 0.3 and 0.1, respectively, the input and output images are shown in Fig. 3, which includes compression and shearing conditions along the x and y directions.

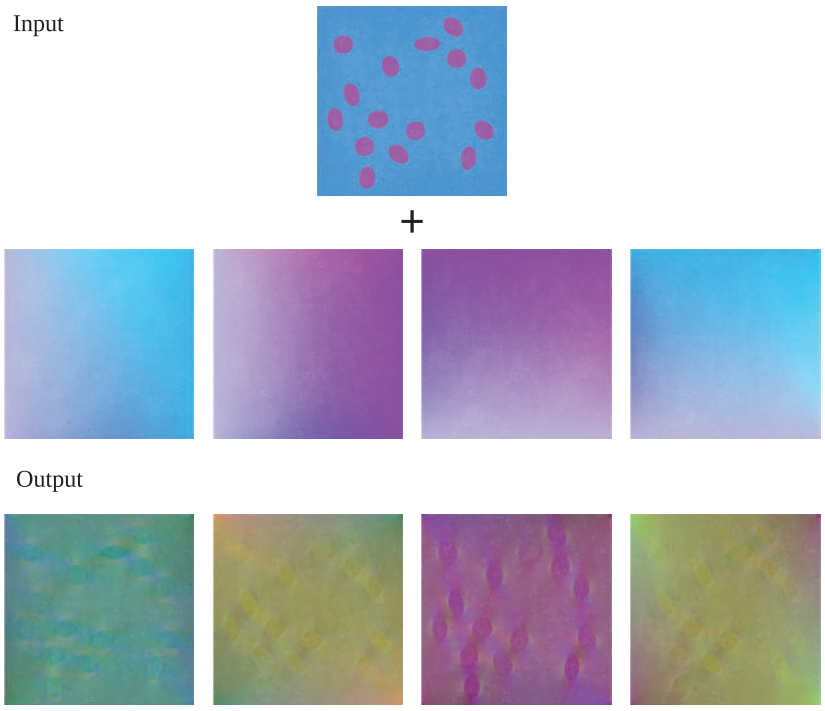

Figure 3: The input/output image representations of numerical simulations of mechanical responses of matrix-inclusion material with $E_{lw} = 10$ GPa, $E_{up} = 100$ GPa, $u_{x,lw} = u_{y,lw} = -0.05$ mm, $u_{x,up} = u_{y,up} = 0.02$ mm, $\sigma_{x,lw} = \sigma_{y,lw} = -2.5$ MPa, $\sigma_{x,up} = \sigma_{y,up} = 2$ MPa, $\tau_{xy,lw} = -1$ MPa, and $\tau_{xy,up} = 1$ MPa

2.3 Slope Stability

The second example refers to limit analysis of slope, which provides a factor of safety of a slope for assessing its stability and safety. Considering upper bound limit analysis, the necessary material parameters are cohesion c [kPa], friction angle ϕ [-] and weight γ [kPa/m]. When slopes have very different sizes and these parameters are length scale dependent, normalizing the size of slopes before creating input images will be more convenient. We use the width of a slope l as the characteristic length and scale the slope into a width equal to 1. The material parameters become $(c\ l)$ [kN], ϕ [-], and $\left(\gamma\ l^2\right)$ [kN], represented by three channels as Eq. (4)

$$R_{value} = 255\left(\frac{(c\ l)-(c\ l)_{lw}}{(c\ l)_{up}-(c\ l)_{lw}}\right),$$
$$G_{value} = 255\left(\frac{\phi-\phi_{lw}}{\phi_{up}-\phi_{lw}}\right),$$
$$B_{value} = 255\left(\frac{\left(\gamma\ l^2\right)-\left(\gamma\ l^2\right)_{lw}}{\left(\gamma\ l^2\right)_{up}-\left(\gamma\ l^2\right)_{lw}}\right). \tag{4}$$

The output results are represented by slip lines or so called failure pattern images, which are obtained by discontinuity layout optimization [14–19] in this example. Other methods such as the strength reduction method or other finite element limit analysis methods are also applicable [20–22]. For illustration, the input and output images are shown in Fig. 4.

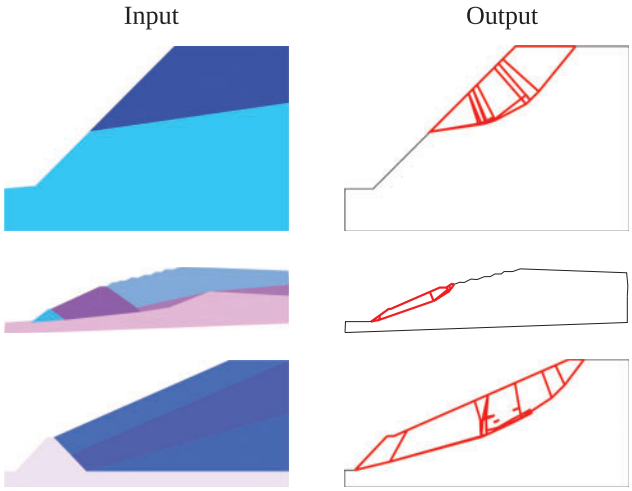

Figure 4: The input/output image representations of numerical simulations of slope stability with $(c\ l)_{lw} = 0$ kN, $(c\ l)_{up}$ =10,000 kN, $\phi_{lw} = 0$, $\phi_{up} = \pi/4$, $\left(\gamma\ l^2\right)_{lw}$ =5,000 kN, $\left(\gamma\ l^2\right)_{up}$ = 1,500,000 kN

2.4 Coupled Thermo-Hydro-Chemical Analysis of Heated Concrete

When concrete members are subjected to fire loadings, explosive spalling may occur, which is the violent fracturing and splitting of concrete pieces from the heated structures. Spalling greatly jeopardizes the integrity and duratbility of structures [23], such as tunnel linings under fire accidents. Spalling is caused partly by the pore-pressure built up inside concrete, referring to the phase change,

permeation and diffusion of liquid water and vapour [24–26] as a strongly coupled thermo-hydro-chemical (THC) process.

The control equations of the THC model of heated concrete are composed of three strongly coupled heat equations, which need to be solved concurrently, as a computing exhausting numerical procedure. Meanwhile, the fire loadings and concrete properties can be complex. Engineers and designers would always like to quickly assess the spalling risk of specific structures considering different conditions, which is a strong motive for developing data driven neural network models.

In [27], the authors summarized fifteen parameters referring concrete properties, fire loadings and environmental moisture as input parameters. In this work, we use grayscale images to represent these parameters. The output parameters are still represented by RGB images, where the saturation degree S_w, the pore pressure p^g, and temperature T occupy three channels as Eq. (5)

$$R_{value} = 255\ S_w,$$
$$G_{value} = 255\ \frac{p^g}{2.5\ \text{MPa}},$$
$$B_{value} = 255\ \frac{T}{1500^{\circ}\text{C}}. \tag{5}$$

Considering the 1D case, the horizontal direction of the output image is taken for space distributions and the vertical direction is taken for time evolutions of S_w, p^g, and T, see Fig. 5. The input and output images are shown in Fig. 6.

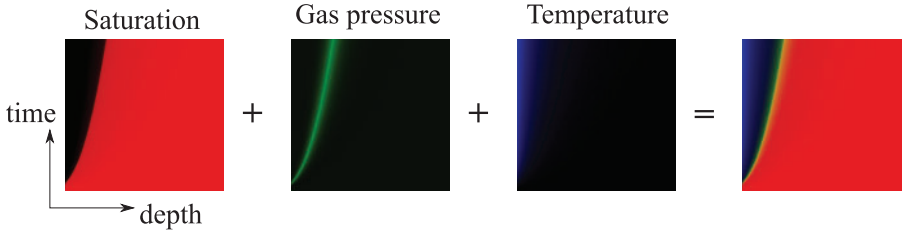

Figure 5: Using an RGB image for representing the time-space variations of S_w, p^g, and T

Figure 6: (Continued)

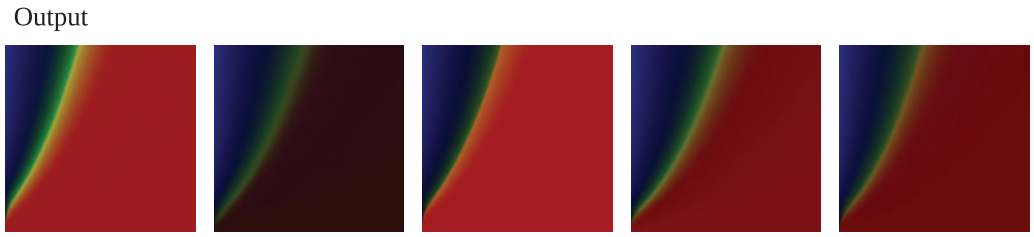

Figure 6: The input/output image representations of numerical simulations of the coupled thermo-hydrochemical analysis of heated concrete

3 Numerical Example for Training a Neural Network by Images

3.1 The Structure of the Neural Network

A hybrid neural network composed of an autoencoder (AE) and a fully connected neural network (FNN) was built by the authors for learning the coupled THC example in [27]. In this work, we simplify the structure and use a network composed of a convolutional neural network (CNN) and a fully connected neural network (FNN). Three designs are considered, see Figs. 7 to 9, which are used for learning the data provided in Subsection 2.4 as examples. CNN can process images efficiently which is composed of convolution and pooling layers, where the convolution layers can extract features and the pooling layers can compress the data. Design 1 has four convolution layers and no pooling layer. Design 2 has three convolution layers and two pooling layers. Design 3 has four convolution layers and three pooling layers. In our example, the CNN is expected to extract and compress features from the input images containing concrete material parameters, environmental humidity and fire load. The FNN is used to build the mapping relation from the features to the output images containing the pore pressure, temperature and saturation information. For both CNN and FNN, the layer plays the role of building the mapping relation from the input vector $\mathbf{X}$ to the output vector $\mathbf{Y}$ as Eq. (6)

$$\mathbf{Y} = f\left(\mathbf{W}\mathbf{X} + \mathbf{b}\right), \tag{6}$$

where $\mathbf{W}$ and $\mathbf{b}$ are the weights and biases respectively used in this layer. $f(\cdot)$ is the activation function. For the FNN, the input vector and output vector are fully connected. In other words, each element of $\mathbf{X}$ influences each element of $\mathbf{Y}$. In contrast, CNN uses a filter in the mapping process that slides at a defined step and outputs the sum of the product [28].

Taking the structure shown in Fig. 9 as an example, the CNN-FNN hybrid neural network is similar to an autoencoder (AE). The CNN plays the role of an encoder when the FNN plays the role of a decoder. There are 6720 sets of input and output images in the example, 90% of which were used as training sets and 10% as test sets. The number of convolution kernels of the first three convolution layers is 16, 32, 64, and the size of the convolution kernels is 3×3. A pooling layer is added after each of the first three convolution layers. The size of the feature map after pooling is 128, 64, 32. The fourth convolution layer contains a convolution kernel with size 1×1, further reducing the dimension of the feature vector. The compression rate of the CNN is 0.13%. The FNN has three hidden layers to amplify the feature vector to the output images, which is similar to the work we presented in [27]. The optimizer of the network is the Adam optimizer, and the stochastic gradient descent method is used. The MSE with regularization term is chosen as the loss function as Eq. (7)

$$\mathbf{L} = \frac{1}{2q}\sum_{i=1}^{q}\|\mathbf{P}_i - \mathbf{S}_i\|^2 + \frac{\lambda}{2q}\|\mathbf{W}\|^2, \tag{7}$$

λ is the regularization hyper-parameter and $\lambda = 10^{-4}$ used in this work. $\mathbf{P}_i$ is the real image vector and $\mathbf{S}_i$ is the predicted image vector. q is the number of output images.

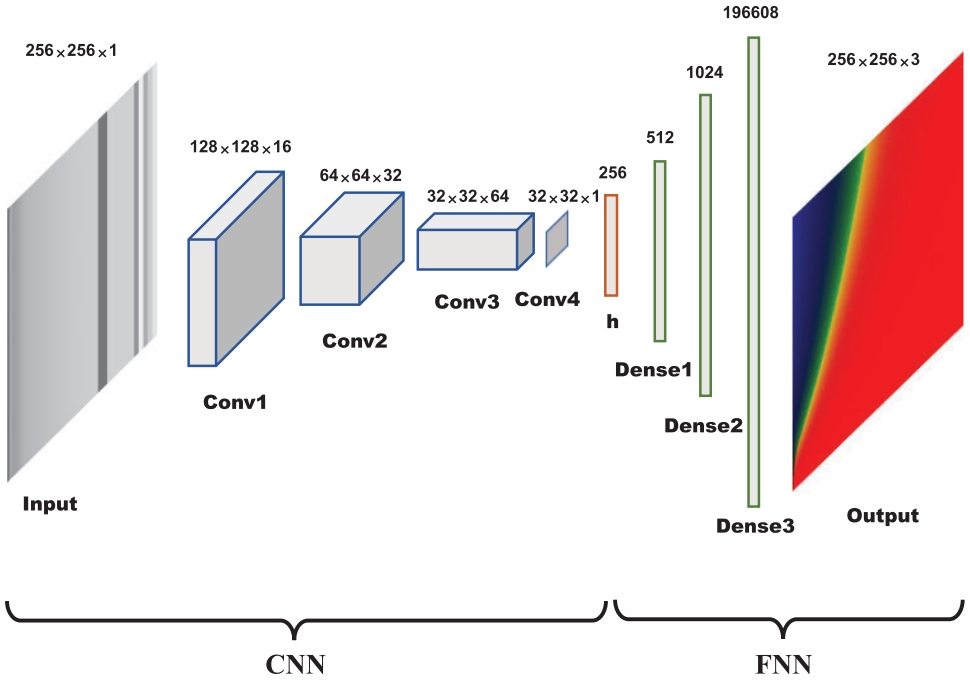

Figure 7: The structure of neural network 1

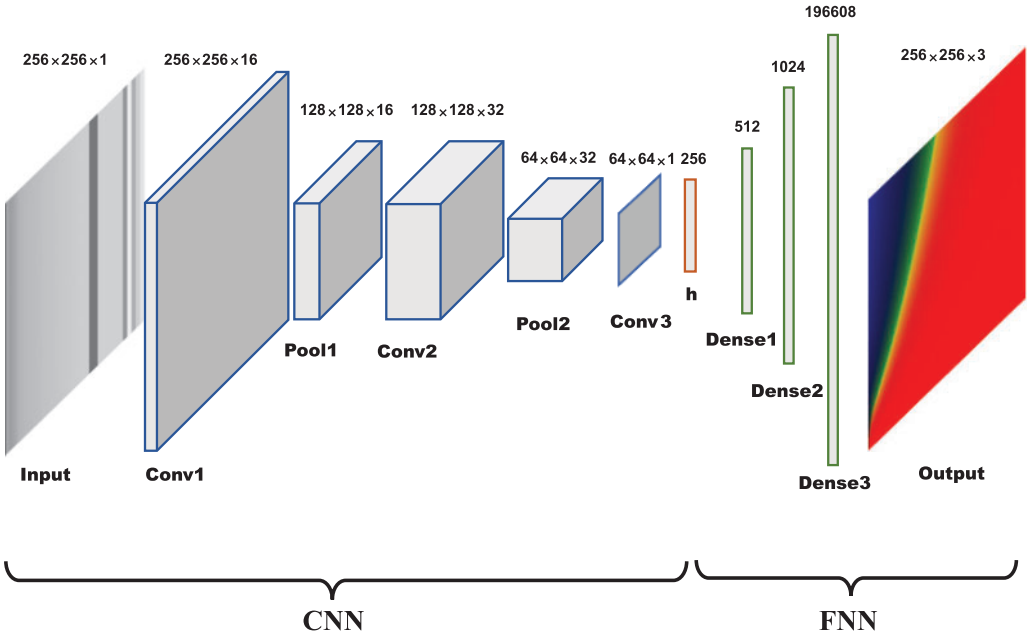

Figure 8: The structure of neural network 2

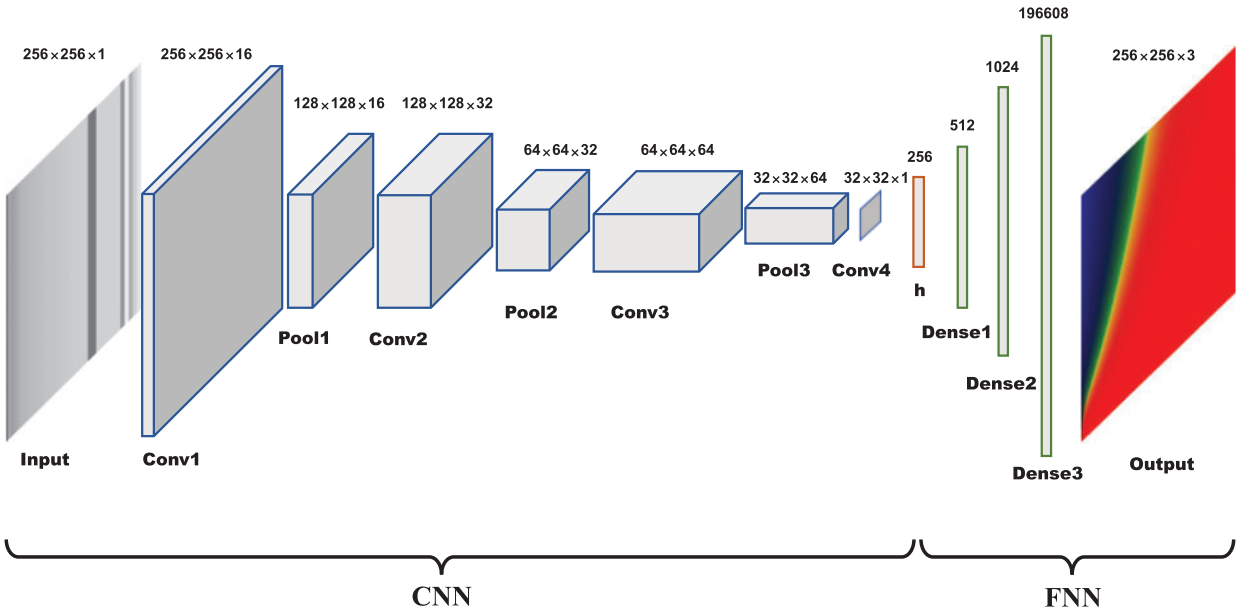

Figure 9: The structure of neural network 3

To avoid over-fitting, the K-fold cross-validation method is used to verify the generalization capability of the model. The 6048 groups of input and output images of the training set were divided into 10 groups of disjoint subsets and trained 10 times. Each time, one group was selected as the verification set and the other 9 groups were selected as the training set. Ten groups of data were trained and evaluated. The average loss and standard deviation obtained from ten-fold cross validation were 0.002295 (±0.000171). Table 1 summarizes the hyperparameters used in the network. It is worth mentioning that numerous designs can be considered. We are still working on improving the designs, which will be presented later.

Table 1: List of hyperparameter of the hybrid autoencoder neural network

Hyper parameters	Index
Learning rate	10^{-4}
Epochs	200
Batch size	64
Activation function	ReLU for CNN and Sigmoid for FNN
Loss function	MSE
Sample size	6720

3.2 Results

The evolution of the MSE loss with the evolution time (epoch) of all designs considering training and testing are shown in Fig. 10. For all designs, the MSE loss drops very fast in training as well as testing. The original, and predicted results and their errors are shown in Figs. 11 to 13, where the original images do not belong to the training or testing data. The error images are generated with

corresponding pixel values of $255 - |\mathbf{P} - \mathbf{S}|$. The results indicate that the shapes are captured and that the colors are very similar. However, some details do not agree very well, especially in regions with high gas pressure. The errors can be further reduced by increasing the amount of data. Generally, design 3 provides the best results. We would like to emphasize that although the results are generally satisfying for the coupled THC example, the design of the network is mostly case dependent. When learning new data sets, the procedure to design, compare, test, and improve the network shall be conducted once more. Some recently proposed networks indicate that it is possible to build some widely applicable networks [29], which uses a similar design for a large number of scenarios. The researchers only need to adjust the hyperparameters. Some research is still ongoing. Except for the design of the network, increasing the number of data sets can effectively improve the prediction accuracy, which can be time consuming.

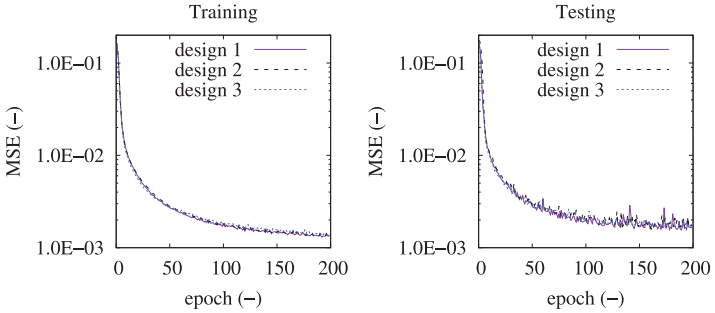

Figure 10: The learning and testing results considering different network designs

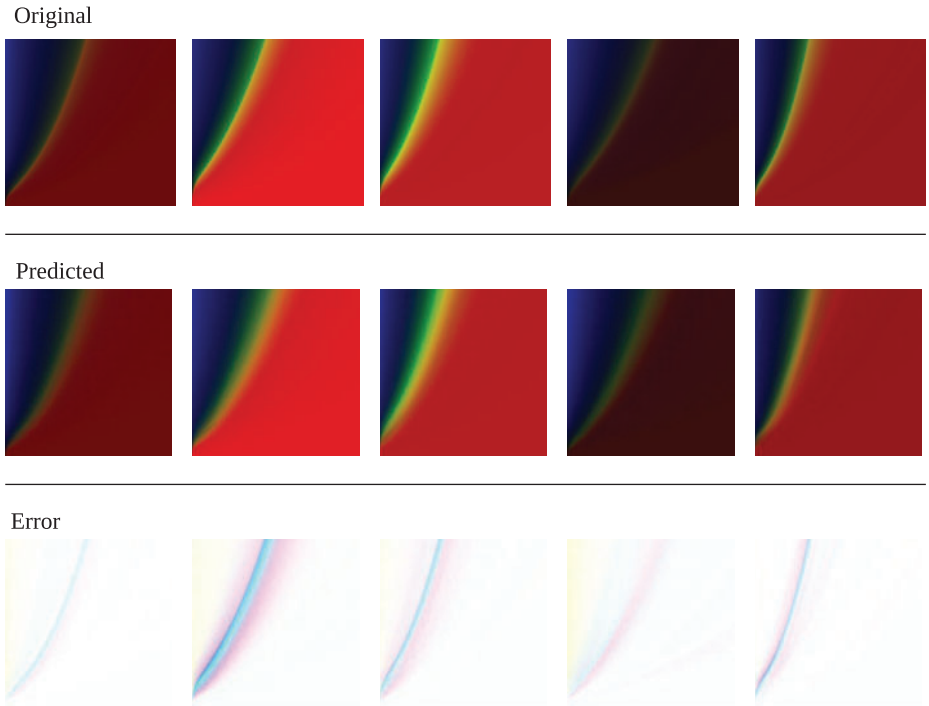

Figure 11: The original, predicted and error images of design 1

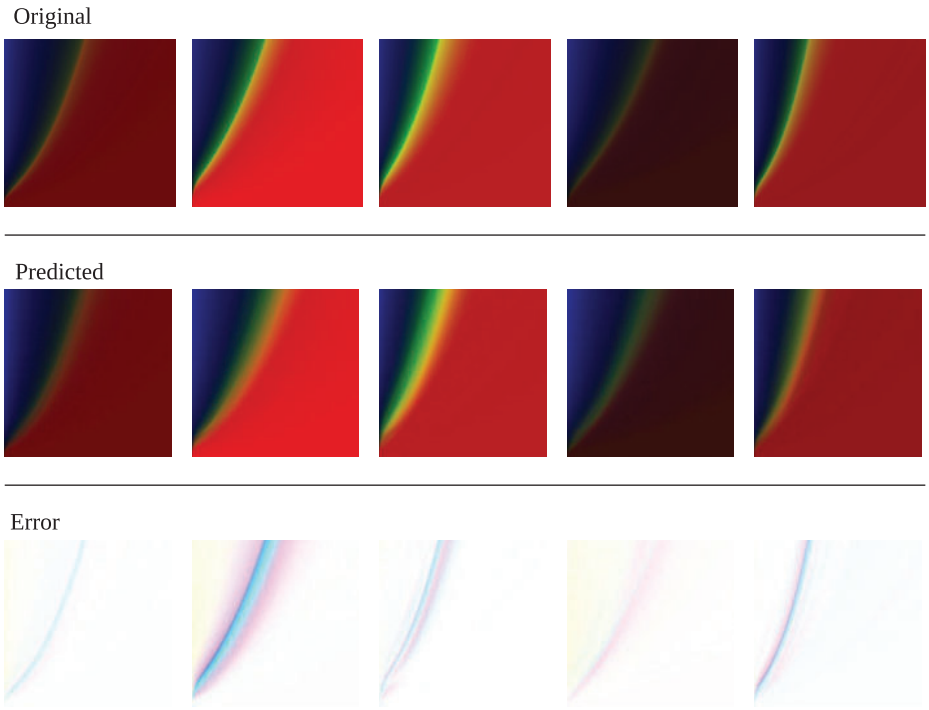

Figure 12: The original, predicted and error images of design 2

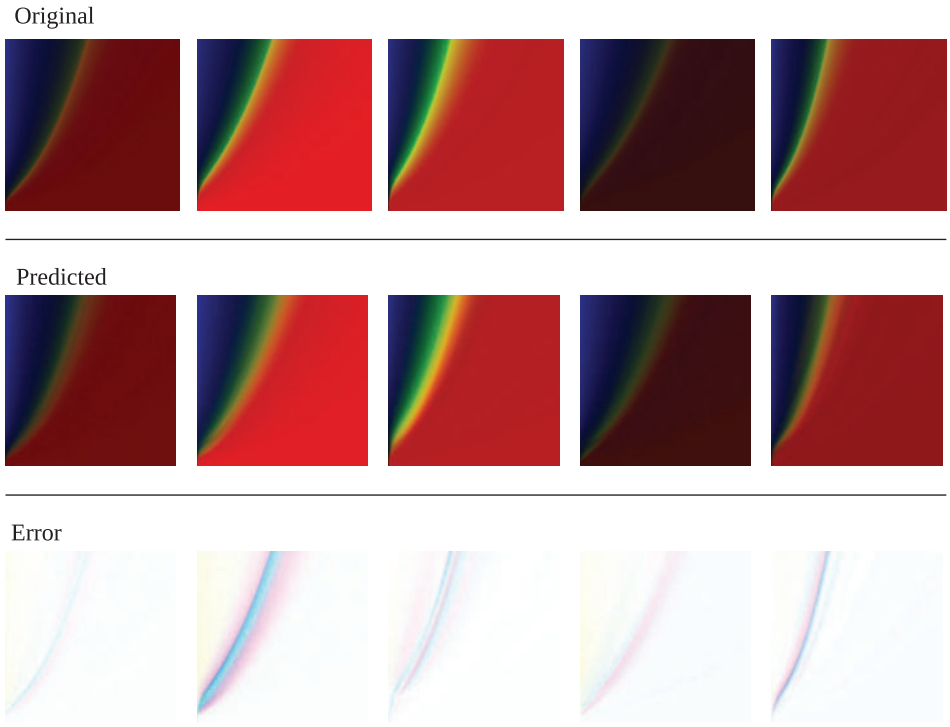

Figure 13: The original, predicted and error images of design 3

4 Conclusions

In this work, we present a strategy for representing the input parameters and output results of numerical simulations by images. With several examples we show that this strategy is simple and compatible with the pre/post-processing parts of popular numerical tools. The images account for the spatial and temporal information used and obtained in the numerical simulations. In addition, all images can be reprocessed by other algorithms. For one of the examples, we train a hybrid CNN-FNN neural network with the input/output images, indicating the effectiveness of the proposed strategy.

Funding Statement: The authors gratefully acknowledge the financial support from the National Natural Science Foundation of China (NSFC) (52178324).

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

1. Raissi, M., Perdikaris, P., Karniadakis, G. (2019). Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics, 378,* 686–707. DOI 10.1016/j.jcp.2018.10.045.
2. Guo, H., Zhuang, X., Rabczuk, T. (2019). A deep collocation method for the bending analysis of kirchhoff plate. *Computers, Materials & Continua, 59(2),* 433–456. DOI 10.32604/cmc.2019.06660.
3. Zhuang, X., Guo, H., Alajlan, N., Zhu, H., Rabczuk, T. (2021). Deep autoencoder based energy method for the bending, vibration, and buckling analysis of kirchhoff plates with transfer learning. *European Journal of Mechanics-A/Solids, 87,* 104225. DOI 10.1016/j.euromechsol.2021.104225.
4. W, E., Yu, B. (2018). The deep ritz method: A deep learning-based numerical algorithm for solving variational problems. *Communications in Mathematics and Statistics, 6(1),* 1–12. DOI 10.1007/s40304-018-0127-z.
5. Anitescu, C., Atroshchenko, E., Alajlan, N., Rabczuk, T. (2019). Artificial neural network methods for the solution of second order boundary value problems. *Computers, Materials & Continua, 59,* 345–359. DOI 10.32604/cmc.2019.06641.
6. Samaniego, E., Anitescu, C., Goswami, S., Nguyen-Thanh, V., Guo, H. et al. (2020). An energy approach to the solution of partial differential equations in computational mechanics via machine learning: Concepts, implementation and applications. *Computer Methods in Applied Mechanics and Engineering, 362,* 112790. DOI 10.1016/j.cma.2019.112790.
7. Nguyen-Thanh, V. M., Zhuang, X., Rabczuk, T. (2020). A deep energy method for finite deformation hyperelasticity. *European Journal of Mechanics-A/Solids, 80,* 103874. DOI 10.1016/j.euromechsol.2019.103874.
8. Goswami, S., Anitescu, C., Chakraborty, S., Rabczuk, T. (2020). Transfer learning enhanced physics informed neural network for phase-field modeling of fracture. *Theoretical and Applied Fracture Mechanics, 106,* 102447. DOI 10.1016/j.tafmec.2019.102447.
9. Hamdia, K. M., Ghasemi, H., Bazi, Y., AlHichri, H., Alajlan, N. et al. (2019). A novel deep learning based method for the computational material design of flexoelectric nanostructures with topology optimization. *Finite Elements in Analysis and Design, 165,* 21–30. DOI 10.1016/j.finel.2019.07.001.
10. Kirchdoerfer, T., Ortiz, M. (2016). Data-driven computational mechanics. *Computer Methods in Applied Mechanics and Engineering, 304,* 81–101. DOI 10.1016/j.cma.2016.02.001.
11. Baiges, J., Codina, R., Castañar, I., Castillo, E. (2020). A finite element reduced-order model based on adaptive mesh refinement and artificial neural networks. *International Journal for Numerical Methods in Engineering, 121(4),* 588–601. DOI 10.1002/nme.6235.

12. Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A. et al. (2016). Tensorflow: A system for large-scale machine learning. *Proceedings of the 12th USENIX Conference on Operating Systems Design and Implementation*, pp. 265–283. Savannah, GA, USA.
13. Li, X., Liu, Z., Cui, S., Luo, C., Li, C. et al. (2019). Predicting the effective mechanical property of heterogeneous materials by image based modeling and deep learning. *Computer Methods in Applied Mechanics and Engineering, 347,* 735–753. DOI 10.1016/j.cma.2019.01.005.
14. Smith, C. C., Gilbert, M. (2007). Application of discontinuity layout optimization to plane plasticity problems. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences, 463,* 2461–2484. DOI 10.1098/rspa.2006.1788.
15. Smith, C. C., Gilbert, M. (2013). Identification of rotational failure mechanisms in cohesive media using discontinuity layout optimisation. *Géotechnique, 63,* 1194–1208. DOI 10.1680/geot.12.P.082.
16. Gilbert, M., He, L., Smith, C. C., Le, C. V. (2014). Automatic yield-line analysis of slabs using discontinuity layout optimization. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences, 470(2168),* 20140071. DOI 10.1098/rspa.2014.0071.
17. Zhang, Y. (2017). Multi-slicing strategy for the three-dimensional discontinuity layout optimization (3D DLO). *International Journal for Numerical and Analytical Methods in Geomechanics, 41,* 488–507. DOI 10.1002/nag.2566.
18. Zhang, Y., Zhuang, X., Lackner, R. (2018). Stability analysis of shotcrete supported crown of NATM tunnels with discontinuity layout optimization. *International Journal for Numerical and Analytical Methods in Geomechanics, 42,* 1199–1216. DOI 10.1002/nag.2775.
19. Sun, Z., Zhang, Y., Yuan, Y., Mang, H. A. (2019). Stability analysis of a fire-loaded shallow tunnel by means of a thermo-hydro-chemo-mechanical model and discontinuity layout optimization. *International Journal for Numerical and Analytical Methods in Geomechanics, 43(16),* 2551–2564. DOI 10.1002/nag.2991.
20. Sloan, S. W., Kleeman, P. (1995). Upper bound limit analysis using discontinuous velocity fields. *Computer Methods in Applied Mechanics and Engineering, 127,* 293–314. DOI 10.1016/0045-7825(95)00868-1.
21. Sloan, S. W. (1989). Upper bound limit analysis using finite elements and linear programming. *International Journal for Numerical and Analytical Methods in Geomechanics, 13,* 263–282. DOI 10.1002/(ISSN)1096-9853.
22. Zienkiewicz, O., Humpheson, C., Lewis, R. W. (1975). Associated and non-associated visco-plasticity and plasticity in soil mechanics. *Géotechnique, 25,* 835–840. DOI 10.1680/geot.1975.25.4.671.
23. Zhang, Y., Zeiml, M., Maier, M., Yuan, Y., Lackner, R. (2017). Fast assessing spalling risk of tunnel linings under RABT fire: From a coupled thermo-hydro-chemo-mechanical model towards an estimation method. *Engineering Structures, 142,* 1–19. DOI 10.1016/j.engstruct.2017.03.068.
24. Gawin, D., Pesavento, F., Schrefler, B. (2006). Towards prediction of the thermal spalling risk through a multi-phase porous media model of concrete. *Computer Methods in Applied Mechanics and Engineering, 195,* 5707–5729. DOI 10.1016/j.cma.2005.10.021.
25. Bazănt, Z., Thonguthai, W. (1978). Pore pressure and drying of concrete at high temperature. *Journal of the Engineering Mechanics Division, 104,* 1059–1079. DOI 10.1061/JMCEA3.0002404.
26. Zhang, Y., Zeiml, M., Pichler, C., Lackner, R. (2014). Model-based risk assessment of concrete spalling in tunnel linings under fire loading. *Engineering Structures, 77,* 207–215. DOI 10.1016/j.engstruct.2014.02.033.
27. Zhang, Y., Gao, Z., Wang, X., Liu, Q. (2022). Predicting the pore-pressure and temperature of fire-loaded concrete by a hybrid neural network. *International Journal of Computational Methods,* 2142011. DOI 10.1142/S0219876221420111.
28. Sun, Y., Xue, B., Zhang, M., Yen, G. G. (2019). A particle swarm optimization-based flexible convolutional autoencoder for image classification. *IEEE Transactions on Neural Networks and Learning Systems, 30(8),* 2295–2309. DOI 10.1109/TNNLS.5962385.
29. Geneva, N., Zabaras, N. (2022). Transformers for modeling physical systems. *Neural Networks, 146,* 272–289. DOI 10.1016/j.neunet.2021.11.022.