# Deep Learning for Stress Field Prediction Using Convolutional Neural Networks

Zhenguo Nie, Haoliang Jiang, Levent Burak Kara*

Department of Mechanical Engineering, Carnegie Mellon University, Pittsburgh, PA 15213, USA

*Corresponding author: Levent Burak Kara

*E-mail addresses*:

zhenguon@cmu.edu;zhenguonie@gmail.com(Z. Nie)

haolianj@andrew.cmu.edu (H. Jiang)

lkara@cmu.edu (L. B. Kara)

## Abstract

This work presents a deep learning based approach for predicting stress fields in the solid material elastic deformation using convolutional neural networks (CNN). Two different architectures are proposed to solve the problem. One is Feature Representation embedded Convolutional Neural Network (FR-CNN) with a single input channel, and the other is Squeeze-and-Excitation Residual network modules embedded Fully Convolutional Neural network (SE-Res-FCN) with multiple input channels. Both architectures are stable and converged reliably in training and test on GPUs. Accuracy analysis shows that SE-Res-FCN has a significantly smaller mean squared error (MSE) and mean absolute error (MAE) than FR-CNN. Mean relative error (MRE) of the SE-Res-FCN model is about 0.32% with respect to the average ground truth. The validation results indicate that the SE-Res-FCN model can accurately predict the stress. As fully trained deep learning models have higher computational efficiency over conventional FEM during runtime, they offer a promising alternative to classical methods in structural design and topology optimization.

*Keywords*: Deep learning; Stress field; CNN; FR-CNN; SE-Res-FCN.

## 1. Introduction

This work presents a deep learning based approach for predicting stress fields of the elastic deformation using deep convolutional neural networks. Deep learning is one of machine learning methods with multiple levels of representation, obtained by composing simple but non-linear modules that each transform the representation at one level into a representation at a higher, slightly more abstract level[1]. Deep learning can deal with much more intricate mapping relationship among many parameters than conventional rigid rules. It used to mainly target at images and audio for classification, style transferring and generation. Due to the development of new learning architectures and the accessibility of computational power, recent advances show that machine learning, deep or not, is getting more and more commonly used in computational engineering fields. For example, fluid simulation [2-5], design and topology optimization[6-9], nonlinear dynamics analysis[10], autonomous vehicles[11-13], molecular dynamics simulation[14-17], quantum learning[18, 19], and so on.

Fluid dynamics simulation was studied a lot in recent years. Long short-term memory (LSTM), as a type of recurrent neural networks, shows attractive potential in modeling temporal dynamics of turbulence [3]. Besides, generative neural network with convolutional layers could construct dynamic Eulerian fluid simulation and obtain the velocity fields, ranging from turbulent smoke to gooey liquids[5]. Generative adversarial networks (GANs), introduced by Ian Goodfellow et al. in 2014[20], can also be widely used in fluid dynamics simulation. Given arbitrary boundary conditions and domain, the conditional GANs can generate the solutions of steady-state heat conduction and incompressible fluid flow[4], and even the phase segregation in fluid flow[21].

In addition, neural networks were widely applied to the computational mechanics [22]. As the strong classification and regression capability, deep learning was embedded into FEM to optimize the numerical quadrature for calculating the element stiffness matrix in the element-by-element basis[23]. There is, however, little research on solid material computational mechanics using pure deep learning methods to obtain stress fields directly, just like FEM, from the imported geometry, loads, boundary conditions, and material properties.

This work presents a deep learning based approach to predict stress fields of the elastic deformation using deep convolutional neural networks. Two different architectures are proposed to solve the problem. One is Feature Representation embedded Convolutional Neural Network (FR-CNN) with a single input channel, and the other is Squeeze-and-Excitation Residual network modules embedded Fully Convolutional Neural network (SE-Res-FCN) with multiple input channels. The validation results show that SE-Res-FCN model has significantly high accuracy in stress field prediction with a mean relative error of 0.25%.

## 2. Architectures of deep neural networks

The presented work is devoted to predicting the stress field of solid mechanical structures with the linear isotropic elastic material. The input data contains geometry, loads, and boundary conditions; output data is the stress field. Based on the input channel condition, two architectures of deep neural networks are proposed respectively: a) FR-CNN architecture with a single input channel, and b) SE-Res-FCN architecture with multiple input channels.

### 2.1 Definition of the physical problem

Consider the cantilever beam in Fig. 1 composed of a homogeneous and isotropic linear elastic material. The left end of the beam is affixed to the wall, and the right end bears loads. The evenly distributed external force is applied in both horizontal and vertical directions. The magnitudes of the two components as shown by $q_x$ and $q_y$ vary separately within a prescribed window of magnitude. The input domain is not limited to a rectangle, but other shapes such as the trapezoid, the trapezoid with curved sides, and all such structures with holes, are utilized. To simplify the model, the material parameters are kept unchanged for all cases.

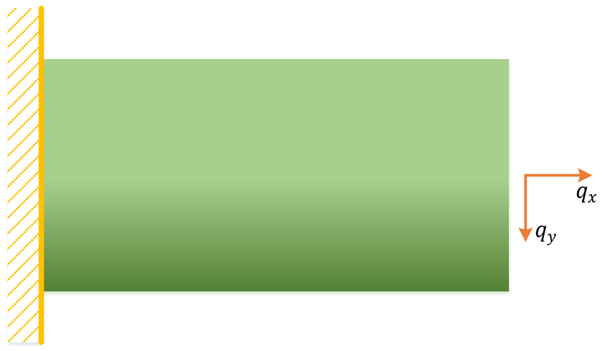


Fig. 1. The schematic diagram of a two-dimensional cantilever with the linear isotropic material.

This two-dimensional cantilever elastic deformation, in essence, is a plane strain problem. The governing equations consist of the strain-displacement equation (1), compatibility equation (2), and equilibrium equation (3), and generalized Hooke's law in Equation (4).

$$\varepsilon_{ij} = \frac{1}{2}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right) \tag{1}$$

$$\frac{1}{2}\left(\frac{\partial^2 \varepsilon_{11}}{\partial {x_2}^2} + \frac{\partial^2 \varepsilon_{22}}{\partial {x_1}^2}\right) = \frac{\partial^2 \varepsilon_{12}}{\partial x_2 \partial x_1} \tag{2}$$

$$\frac{\partial \sigma_{ii}}{\partial x_i} + \frac{\partial \sigma_{ij}}{\partial x_j} + f_i = 0 \tag{3}$$

$$\begin{bmatrix} \sigma_{11} \\ \sigma_{22} \\ \sigma_{12} \end{bmatrix} = \frac{E}{(1+\nu)(1-2\nu)} \begin{bmatrix} 1-\nu & \nu & 0 \\ \nu & 1-\nu & 0 \\ 0 & 0 & 1-2\nu \end{bmatrix} \begin{bmatrix} \varepsilon_{11} \\ \varepsilon_{22} \\ \varepsilon_{12} \end{bmatrix} \tag{4}$$

where $i$ and $j$ are subscripts with values of 1 or 2, $x_1$ represents the x-axis and $x_2$ represents the y-axis, $u_i$ is the displacement in $x_i$ direction, $\varepsilon_{ij}$ is strain on surface $x_i$ in $x_j$ direction, $\sigma_{ij}$ is stress on surface $x_i$ in $x_j$ direction, $f_i$ is the body force component in $x_i$ direction , $E$ is Young's modulus, and $\nu$ is Poisson's ratio.

## 2.2 Feature representation embedded convolutional neural networks (FR-CNN) with a single input channel

As illustrated in Fig. 2, a simple fully convolutional neural network is proposed to perform the prediction of the Mises stress field. The input of the model is an image including the information of the geometry and load position. The output of the model is a color-code stress field, which contains a Mises stress value for each pixel. Of course, it is necessary to point out that the output result is just a two-dimensional matrix but not a three-channel RGB image although it is displayed in a colorful image.

The model has an encoder-decoder structure. The encoder network consists of two convolutional layers **E1** and **E3** and two pooling layers (**E2** and **E4**). Each convolutional layer has a filter with a kernel of $3 \times 3$ and a stride of $1 \times 1$. The padding scheme is zero padding to keep the output image the same size as the input. After a reshape layer **E5**, and a fully connected (FC) layer **E6**, we can get the feature representation (**FR**) layer at the bottleneck (BN). The load vector $(q_x, q_y)$ is jointed to the FR vector in the end. The decoder network copies the architecture of the encoder and reverses it. Upsampling layers take the place of pooling layers for increasing the image resolution. The whole model contains totally five convolutional layers.

The convolutional layer applies a convolutional operation to the input channels and passes the result to the next layer. CNN can extract distinguishing features from the input images through the scanning and convolutional operation by filter banks[24, 25].

The height, width, and channel of the input image vary through the model: Input image is $24 \times 32 \times 1$; **E1** is $24 \times 32 \times 32$; **E2** is $12 \times 16 \times 32$; **E3** is $13 \times 16 \times 64$; **E4** is $6 \times 8 \times 64$; **E5** is $3074 \times 1 \times 1$; **E6** is $1024 \times 1 \times 1$; **E7** is $30 \times 1 \times 1$; **FR** is $32 \times 1 \times 1$; **D1** is $1024 \times 1 \times 1$; **D2** is $3074 \times 1 \times 1$; **D3** is $6 \times 8 \times 64$; **D4** is $12 \times 16 \times 64$; **D5** is $12 \times 16 \times 32$; **D6** is $24 \times 31 \times 32$; **D7** is $24 \times 32 \times 16$; **D8**-Output image is $24 \times 32 \times 1$.

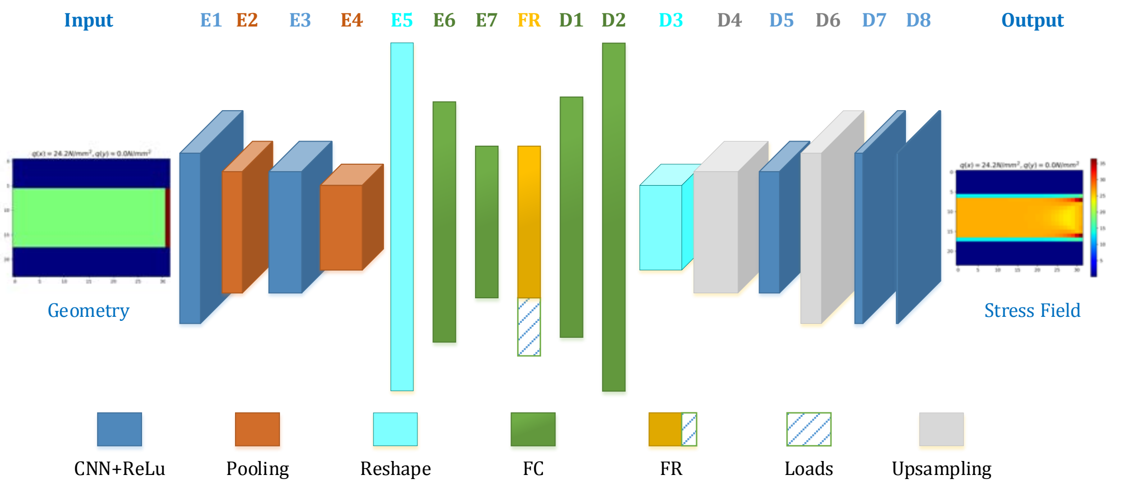


Fig. 2. The architecture of FR-CNN with a single input channel.

## 2.3 Squeeze-and-Excitation Residual Network modules embedded fully convolutional neural network (SE-Res-FCN) with multiple input channels

The FR-CNN model can only solve simple FEM problems, in which the loads distribute uniformly and boundary conditions are the same in X and Y directions. To increase its versatility and decrease prediction error, we propose a SE-Res-FCN architecture with multiple input channels as shown in Fig. 3. A downsampling-and-upsampling structure is employed, and five Squeeze-and-Excitation residual network (SE-ResNet) modules are used between the downsampling and upsampling structures. All convolutional layers are followed by batch normalization (BN) and ReLU layers.

The input data contains five channels: 1) geometry channel, 2) X-component of the load, 3) Y-component of the load, 4) X-component of the boundary condition, 5) Y-component of the boundary condition. For each channel, the image is a two-dimensional matrix. All of these five channels are stacked together. As with the FR-CNN architecture, the output data of SE-Res-FCN is also one channel stress field.

Downsampling comprises three classic convolutional layers (**C1**, **C2**, and **C3**), and upsampling comprises three reverse convolutional layers (**C4**, **C5**, and **C6**). Referring to the structure of image transformation networks [26], we use $9 \times 9$ kernels in the first and last layers (**C1** and **C6**), and $3 \times 3$ kernels in all the other convolutional layers.

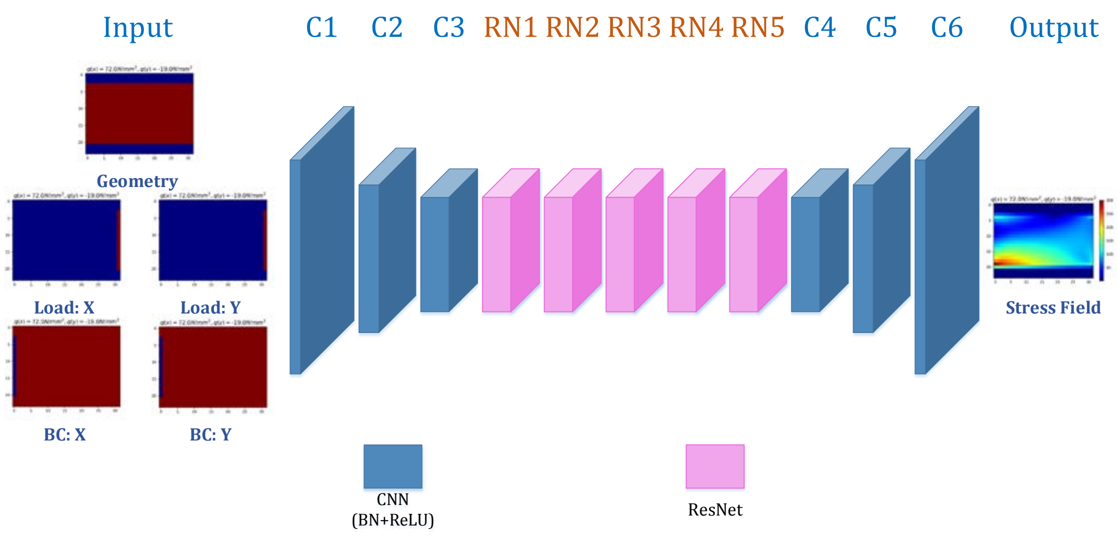


Fig. 3. The architecture of SE-Res-FCN with multiple input channels.

ResNet modules are used to weaken the degradation problem [27]. As shown in Fig. 4, a SE-ResNet module comprises two convolutional layers with $3 \times 3$ kernels and one Squeeze-and-Excitation (SE) network block. The output of the SE-ResNet module, as shown in Equation (5), can be conducted by feedforward neural networks with “shortcut connection”.

SE blocks are used inside the SE-ResNet modules to improve the representational capacity of the network by enabling it to perform dynamic channel-wise feature recalibration [28]. The input data $u \in \mathcal{R}^{H \times W \times C}$ is shrunk into $S(u) \in \mathcal{R}^{C}$ through the global average-pooling layer. Then two fully connected layers are employed to downsample (FC+ReLU) and upsample (FC+Sigmoid) the linear array respectively. A reshape operation is conducted to obtain the excitation output data $E(u)$ that has the same dimension and size as the initial input data $u$. The final output of the block is obtained by a rescaling operation that is the element-wise matrix multiplication, as shown in Equation (6).

$$z = F(x, \{w_i\}) + x \tag{5}$$

$$v = E(u) \times u \tag{6}$$

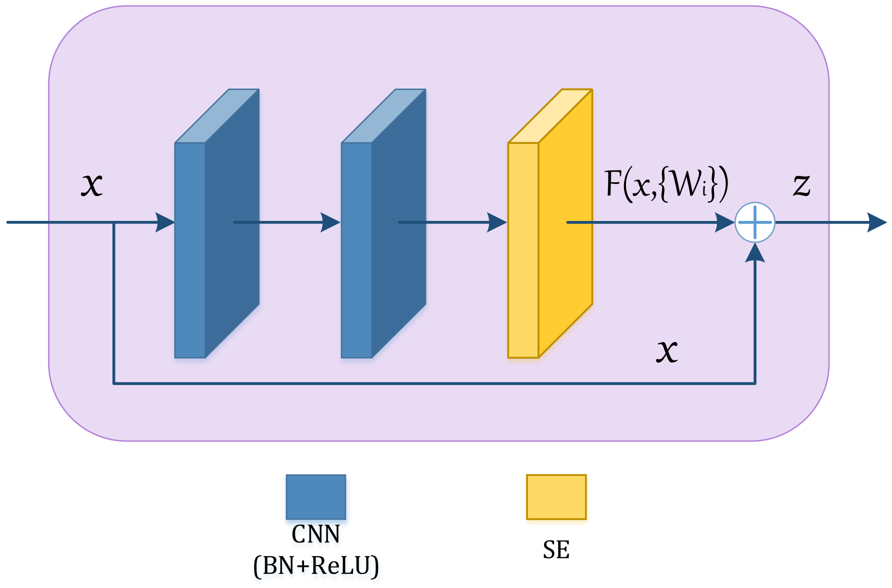


Fig. 4. The ResNet module with a SE block.

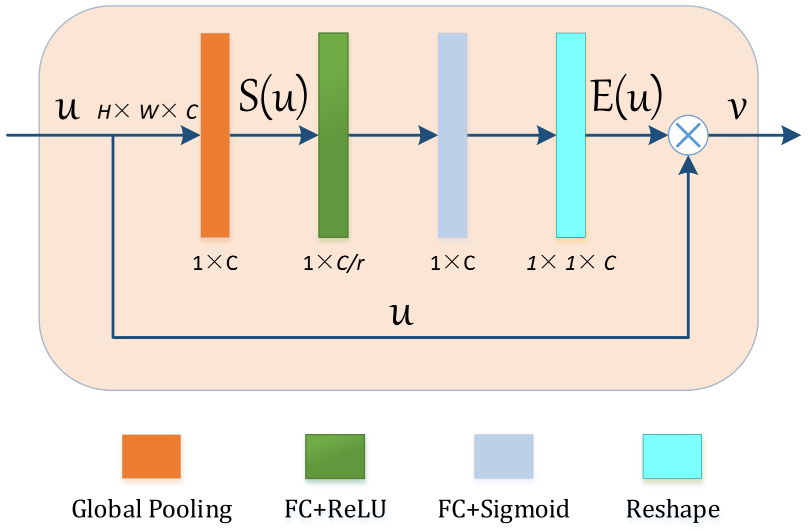


Fig. 5. The SE network block.

## 3. Methodology of the deep learning

In this section, we illustrate the format of datasets and the methodology of deep neural networks. The activation function, batch normalization, loss function, metric, learning rate and batch size are all discussed.

## 3.1 Datasets

A 2D finite element method (FEM) software SolidsPy is used to generate the training and test data [29]. Some of the results are selected randomly from a total of 120,960 cases as shown in Fig. 6. The resolution of each image is $32 \times 24$. Images in the left column are input data for the single channel, and right images are the ground truths. For input images, the blue color represents the void domain with no material; the light green color represents the solid domain with the isotropic material; the brown color on the right side represents the position of the loads. All the three colors are numbered in the input-matrix as zero, one, and two. For the ground truths, the Mises stress within the solid domain is plotted as stress field, and the stress value is zero within the void domain. Among all computed FEM cases, the stress magnitude varies from 0 to 2,475 MPa with an average of 67.8 MPa.

Based on FEM results, the dataset of multi-channel for SE-Res-FCN can be transformed from the dataset of the single- channel for FR-CNN, as shown in Fig. 7. Firstly, for the geometry channel matrix, as with the single channel geometry matrix, zeros represent the void domain in blue, and ones represent the solid domain in brick red. Secondly, for the two load-channels, the load component magnitude is located on the pixel point of the force position, which is in brick red; and zeros are laid on the other points, which are in blue. Thirdly, for two boundary condition channel matrices, all constrained points are marked as one (blue), and free points are marked as zero (brick red).

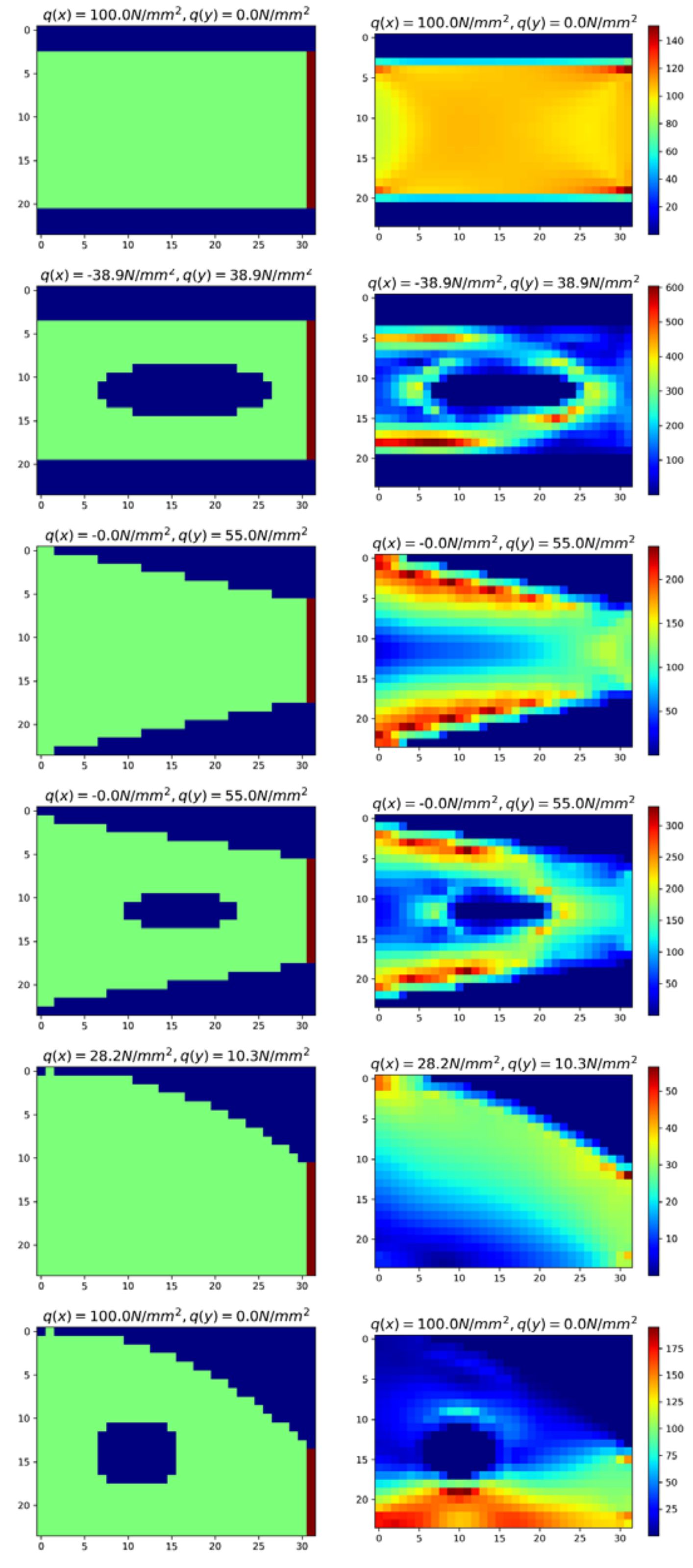


Fig. 6. Samples of the dataset computed by FEM. Images in the left column are input channels including geometry and load position; Right images are stress fields (Units: mm-MPa-N).

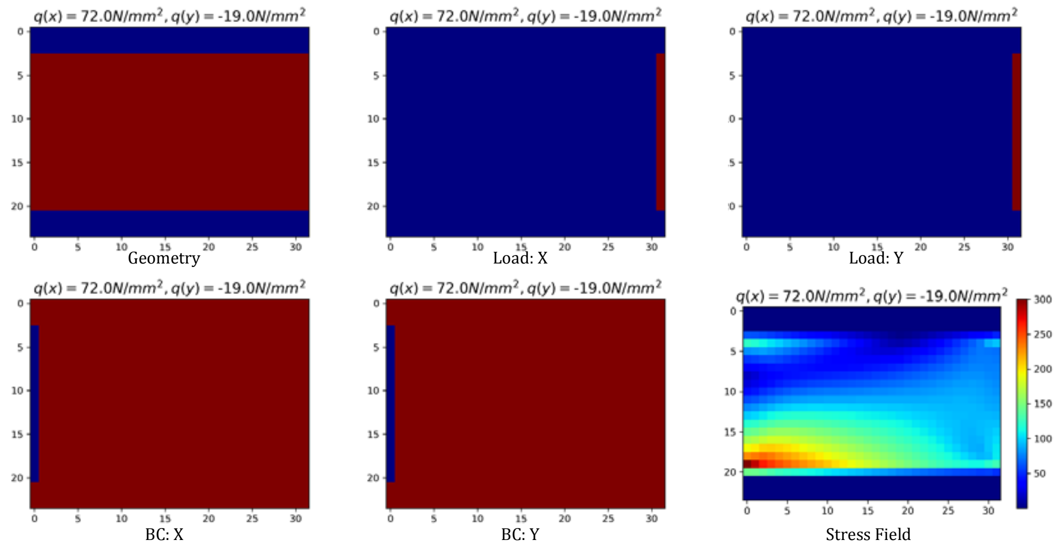


Fig. 7. A data example of multiple channels for the SE-Res-FCN architecture with five input channels and one output stress field.

## 3.2 Activation function and batch normalization

CNNs are designed to process data that comes in the form of multiple arrays, for example, a color-code image composed of three 2D arrays containing pixel intensities in the three-color channels[1]. Unless stated, a rectified linear unit (ReLU) is deployed as the default activation function for both the two proposed CNN architectures. For FR-CNN model, Softplus functions are used in layer **E7** and **D1**, which are just on both sides to the feature representation layer. Both ReLU and Softplus are similar except near zero where the Softplus is differentiable. ReLU is much faster than any other activation functions in deep learning.

Batch normalization (BN) is used in each convolutional layer except the last one for both the two deep learning architectures. BN normalizes the input layer by adjusting and scaling the activations, and can avoid the internal covariate shift [30].

## 3.3 Loss function and metric

MSE (mean squared error) and MAE (mean absolute error) are two of the most common metrics used to evaluate the model accuracy for continuous variables. The prediction $\hat{y}$ and ground truth $y$ in our current model are both two-dimensional matrices with a size of $32 \times 24$. Before computing MSE and MAE, these two 2D-matrices are reshaped into 1D-arrays with a length of 768. The reshaped prediction $\hat{y}$ can be expressed as $\hat{y} = (\hat{y}_1, \hat{y}_1, \cdots, \hat{y}_n)$, while the reshaped ground truth $y = (y_1, y_1, \cdots, y_n)$. MSE and MAE can be expressed as,

$$\mathrm{MSE} = \frac{1}{n}\sum_{j=1}^{n}\left(y_j - \hat{y}_j\right)^2 \tag{7}$$

$$\mathrm{MAE} = \frac{1}{n}\sum_{j=1}^{n}\left|y_j - \hat{y}_j\right| \tag{8}$$

as $n = 768$.

Based on the Cauchy-Schwarz inequality, it has $\mathrm{MSE} \geq \mathrm{MAE}^2$. MSE has a tendency to be increasingly larger than $\mathrm{MAE}^2$ with an increase of test sample, and is more sensitive on data variance. In our training process, MSE is employed as the loss function for training, while both error measures are used as metrics to evaluate convergence.

## 3.4 Learning rate and batch size

Adaptive Moment Estimation (Adam) is used to train our architectures. Adam, which is different to classical stochastic gradient descent (SGD), is a method for efficient stochastic optimization that only requires first-order gradients with little memory requirement [31]. The Adam optimization algorithm is a combination of AdaGrad [32] and RMSprop [33] algorithms. In most cases, the appropriate learning rate usually becomes smaller and smaller along with the training. Adam calculates an exponential moving average of the gradient and the squared gradient, and then control the decay rates of these moving averages.

With the increase of training data, it becomes more and more difficult to input the whole dataset for deep neural networks. Therefore, a typical implementation is to divide the training dataset into a series of batches of some fixed size [34]. By monitoring the used GPU memory, a suitable batchsize can be obtained by manual adjustment.

## 4. Results and Discussions

We trained and validated the two models on NVidia GeForce GTX 1080 Ti GPGPU. The programs were written in TensorFlow. Both architectures are stable and converged reliably. After training, to compute the stress fields in current research, it takes only 1.56 seconds for FR-CNN to process 120,000 FEM cases, and 10.4 seconds for SE-Res-FCN. FEM software, in contrast, takes nearly ten hours (on Intel i7-6500U CPU) to accomplish the same amount of FEM computation.

### 4.1 Accuracy and performance with maximum training data size

Model training is conducted with the maximum training data size. The training data size is 100,000, and the test data size is 20,000. Fig. 9 shows MSE curve as a function of epochs. Chart (a) is in arithmetic coordinates, and chart (b) is in logarithmic coordinates. Chart (a) shows that all the four MSE curves decline rapidly in the first fifty epochs, and then begin to flatten. Form chart (b), it can be seen that FR-CNN preserves a nearly constant order of magnitude after 1000 epochs. To the contrary, SE-Res-FCN continues to decrease with more epochs, even after 5000 epochs. SE-Res-FCN has a significantly smaller MSE than FR-CNN in the end. For either architecture, the training data and test data have roughly equal MSE trends and magnitudes. This means the model has minimal overfitting. Fig. 10 shows MAE curves on training data and test data of two architectures. The changing trend of MAE is almost the same as MSE. SE-Res-FCN has a much better accuracy than FR-CNN as it has a much deeper architecture.

Here we define the mean relative error (MRE) as the ratio of MAE and the mean ground truth ($\bar{y}$), as shown in Equation (11) and (12). MRE is an error rate in percentage to measure how close predictions are to the ground truths. When MRE is zero, it means the prediction is absolutely the same as the ground truth, and MRE = 100% means 100% error deviating from the ground truth. All the error indicator results are summarized in Table 1. We can see that MRE of SE-Res-FCN is only around 0.32%, which we deem is acceptable accuracy for predicting the ground truth.

$$\mathrm{MRE} = \frac{\mathrm{MAE}}{\bar{y}} \times 100\% \tag{9}$$

$$\bar{y} = \frac{1}{n}\sum_{j=1}^{n} y_j \tag{10}$$

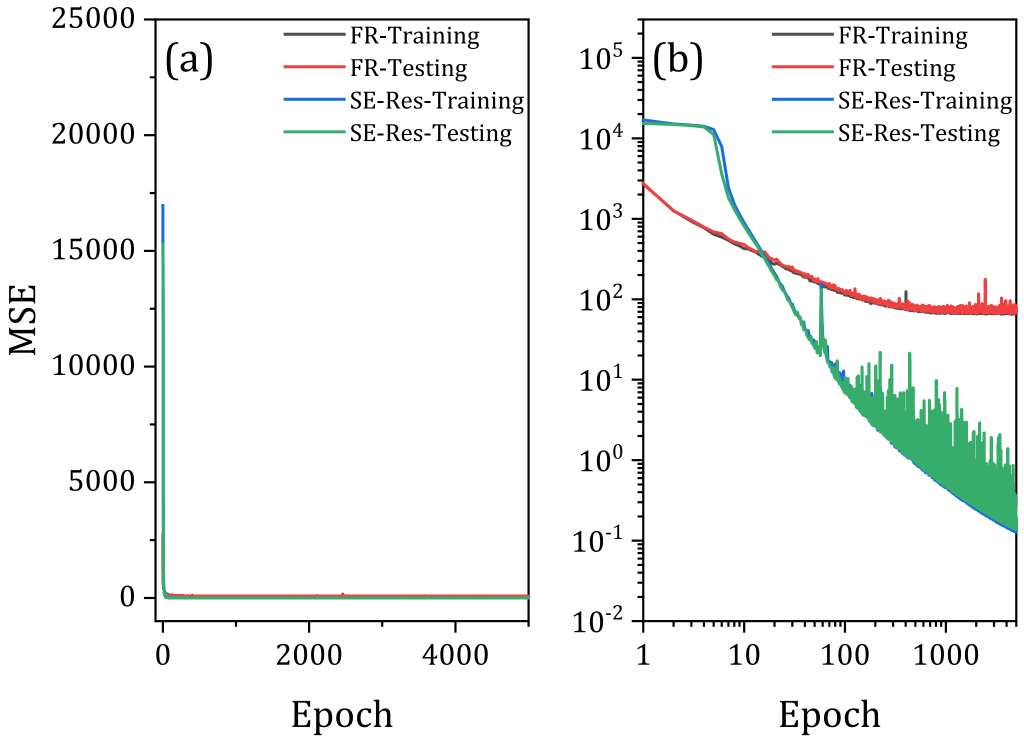


Fig. 8. MSE curves on training and test data of two architectures. (a) is shown in arithmetic coordinates; (b) is shown in logarithmic coordinates.

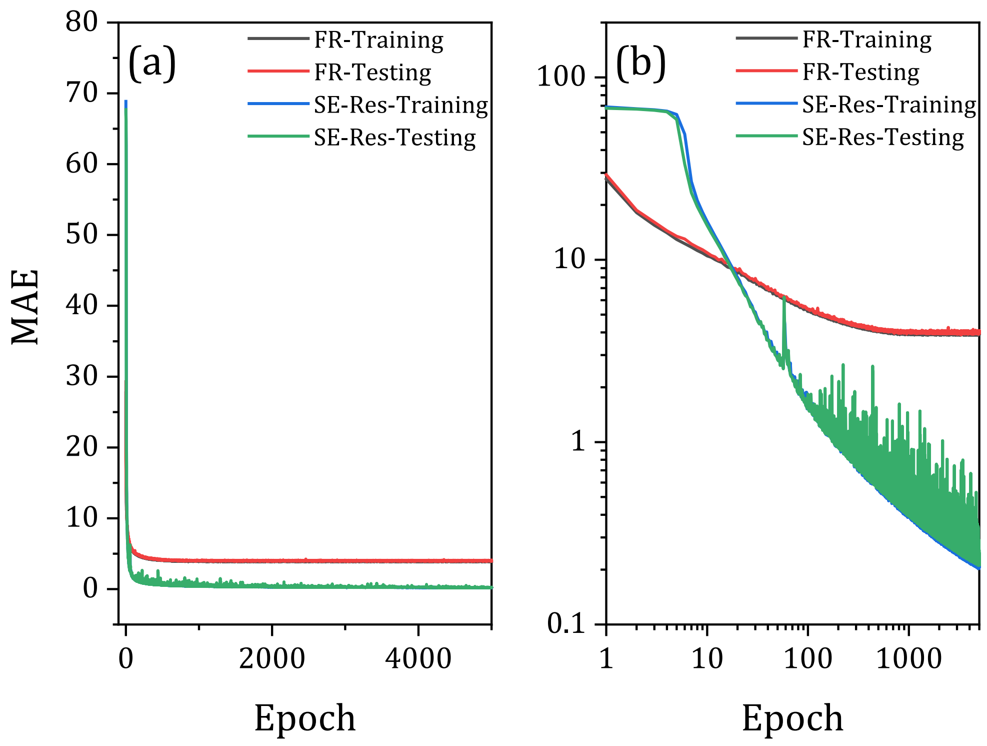


Fig. 9. MAE curves on training and tes data of two architectures. (a) is shown in an arithmetic coordinate; (b) is shown in a double logarithmic coordinate. The changing trend of MAE is almost the same as MSE.

Table 1. Error indicator results. Epoch = 5000, $\bar{y}$ = 67.8021. F means FR-CNN, and S means SE-Res-FCN.

| Data | F-MSE | S-MSE | F-MAE | S-MAE | F-MRE | S-MRE |
|---|---|---|---|---|---|---|
| Training | 69.6734 | 0.14494 | 3.90516 | 0.21529 | 5.760% | 0.31753% |
| Test | 73.5044 | 0.15146 | 4.01891 | 0.22212 | 5.927% | 0.32776% |

## 4.2 Validation and evaluation

We select several random results to demonstrate the performance of our models. Fig. 11 shows five stress fields computed by the SE-Res-FCN model. None of these test cases appeared in the training set. Each row is one case, and the first image is the ground truth. Images from the second to the last are predictions in different epochs. It can be seen that the prediction becomes closer and closer to the ground truth, and the last column is almost identical to the ground truth.

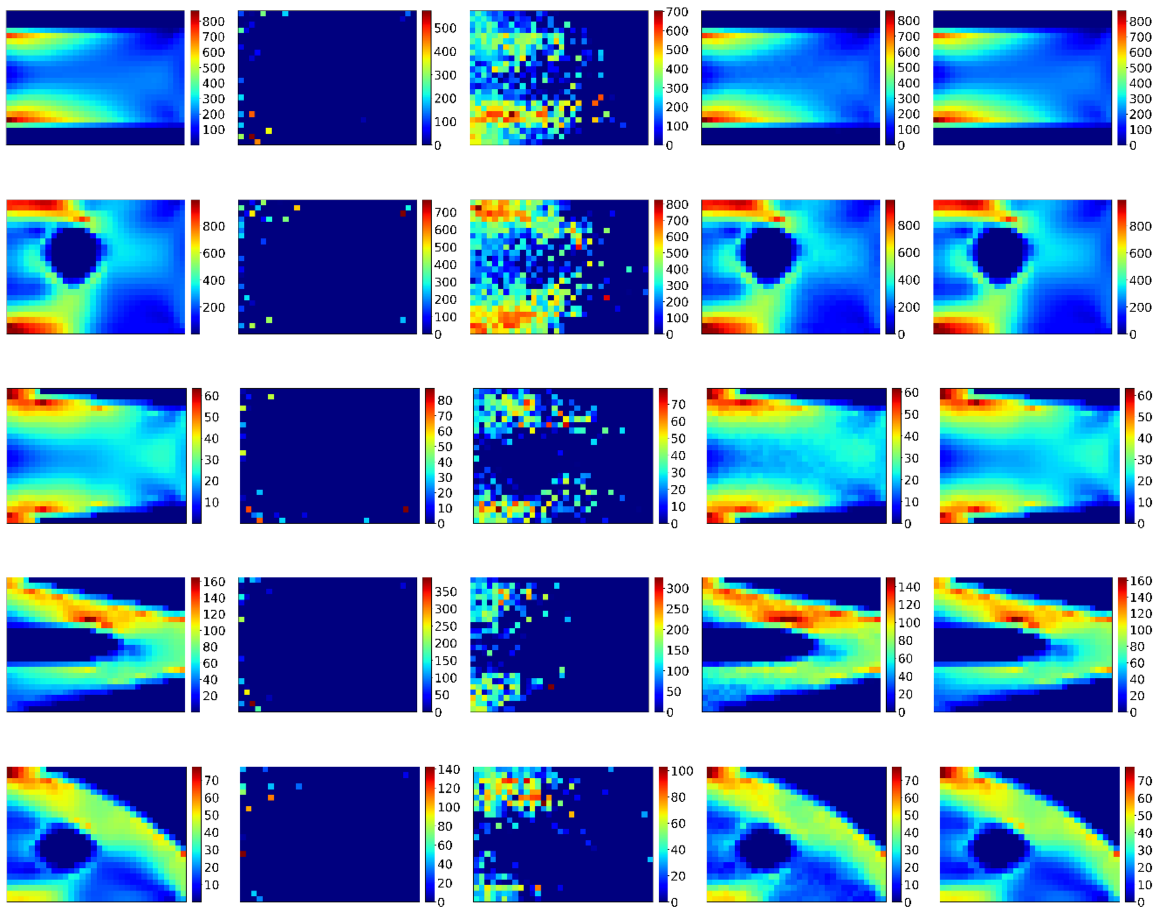


Fig. 10. Computed stress fields by SE-Res-FCN: ground truths and evolutive predictions. From left to right: 1) ground truths; 2) epoch = 1; 3) epoch = 10; 4) epoch = 100; 5) epoch = 5000. The ratio of training data size and test data size is 100,000/20,000.

## 4.3 Effect of the training data size on performance

Besides using all the data for training and test with the ratio of 100,000/20,000, we also reduce the training data size from 100,000 to 20,000 to demonstrate the effect of training size. Fig. 12 shows the change of MSE and MAE for both training and test with different training sizes. The variations of MSE and MAE follow similar trends. With the increase of the training data, both MSE and MAE decline gradually. The standard deviation (error bar) also decreases with training data. In addition, MSE and MAE of the test data are slightly larger than those of the training data. Similarly, we plot the computed stress fields with different training data size in Fig. 13. By using different sizes of the training data, trainings on the SE-Res-FCN model are finished at the five thousandth epoch. As shown, the computed stress fields look similar, as MAE values across different training sizes are relatively small with respect to the magnitudes of the stress fields.

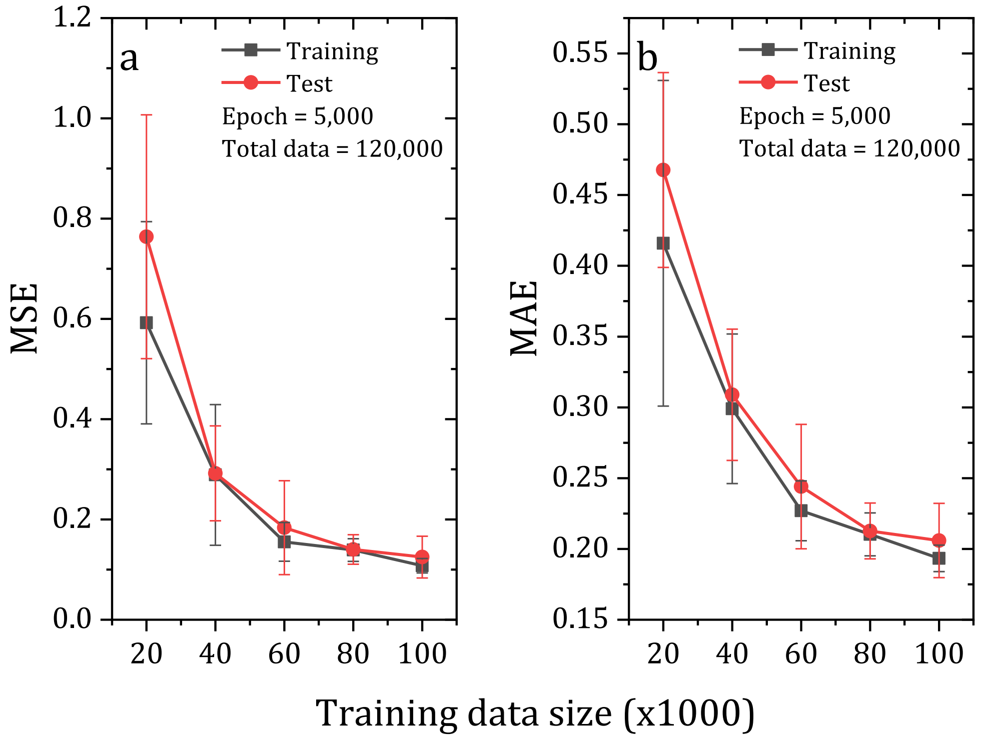
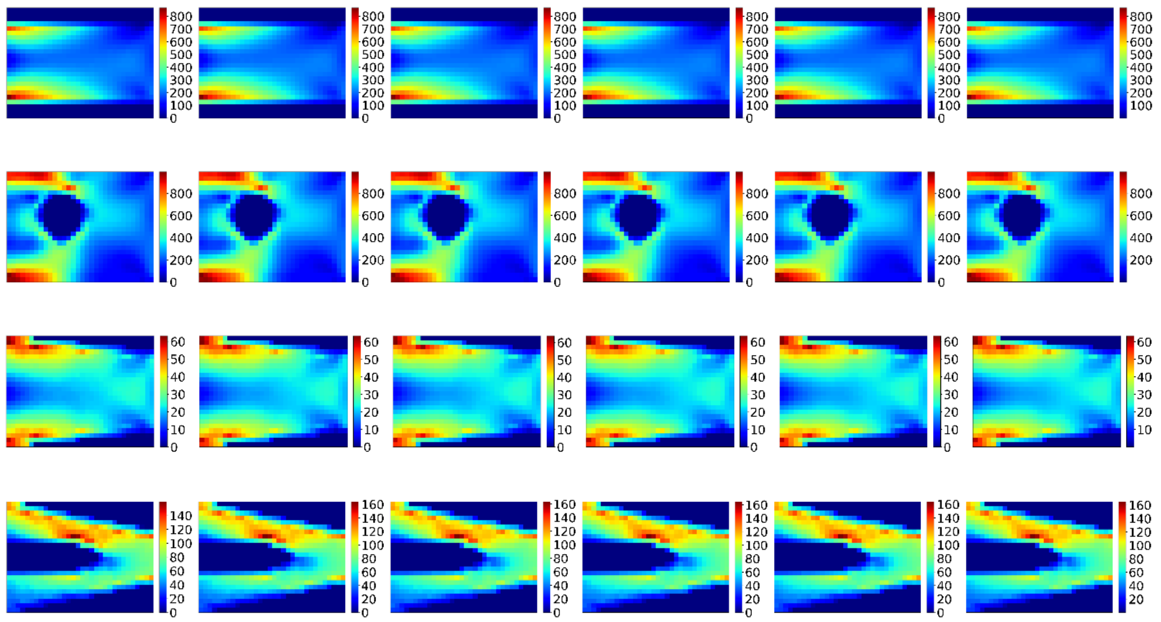


Fig. 11. Effect of training data size on performance of the SE-Res-FCN model.



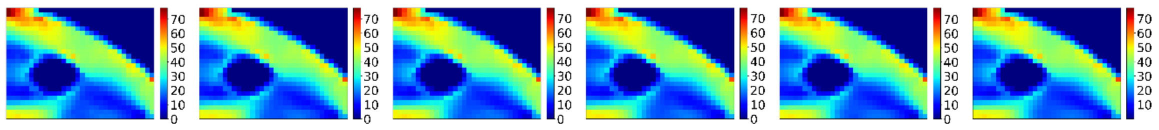


Fig. 12. Computed stress fields with different training data sizes by SE-Res-FCN. From left to right, the training data size is: 1) 20,000; 2) 40,000; 3) 60,000; 4) 80,000; 5) 100,000; 6) ground truths.

## 4.4 Effect of the hierarchical nature of deep learning on prediction accuracy

In addition to the two architectures described above, we also design another intermediate architecture in Fig. 14, which keeps the FR-CNN structure and inputs multiple channels instead of a single channel. Five channels are classified into three categories: geometry, loads and boundary conditions. All the three encoders are independent and parallel to each other. After the fully connected (FC) layers, three feature representations (FR) are combined in turn. Then the combined FR is decoded through reverse CNN layers. Such an intermediate model consists of eight convolutional layers totally, which is far less than SE-Res-FCN.

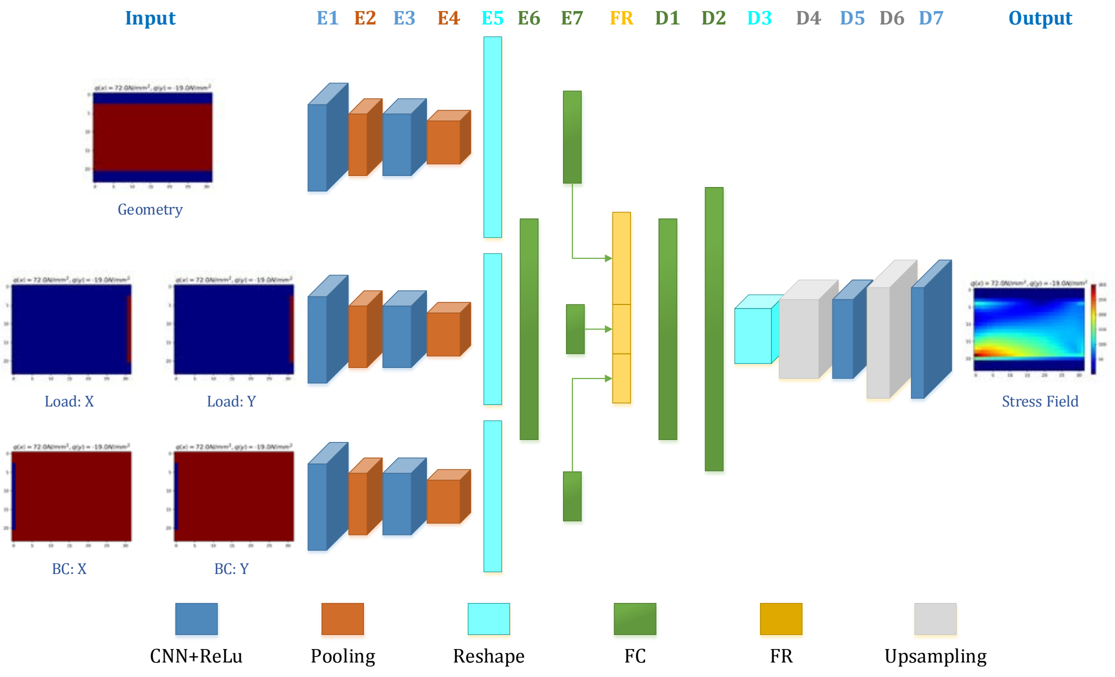


Fig. 13 An intermediate architecture using the FR-CNN architecture and multiple input channels

We trained the intermediate model on the same GPU, and the training result shows the intermediate model has no improvement on training accuracy with respect to FR-CNN. This supports the premise that as the hierarchical architecture becomes deeper within certain limits; its prediction becomes more accurate. Input channel number does not have a significant effect on the accuracy. For better prediction accuracy, SE-Res-FCN is a reasonable choice. ResNet modules are able to alleviate the degradation problem by using shortcut connections.

Comparatively speaking, FR-CNN can reduce the computational time, and reach an acceptable accuracy. For well-extracted feature representation, it is possible to implement a linear interpolation or extrapolation. It means that even for unseen cases, it is possible to search for answers in the existing representations. For SE-Res-FCN, it keeps more information of the input and thus enables more non-linear operations due to its deep architecture. In addition, due to its fully convolutional network architecture, it is able to account for large variations in size of the 2D FEM data. This makes SE-Res-FCN a great alternative to classical FEM once the training data is sufficient to train the network.

## 5. Conclusions

In this paper, we proposed two deep learning architectures for stress field prediction. One is Feature Representation embedded Convolutional Neural Network (FR-CNN) with a single input channel, and the other is Squeeze-and-Excitation Residual network modules embedded Fully Convolutional Neural network (SE-Res-FCN) with multiple input channels. A 2D FEM software SolidsPy was used to compute training and test data, which contains totally 120,960 FEM cases.

Both the two architectures are stable and converged reliably in training and test. MSE and MAE results show that SE-Res-FCN can obtain higher accuracy than FR-CNN. The mean relative error of the SE-Res-FCN model is only 0.32% with respect to the ground truth. This means our SE-Res-FCN model is accurate enough to predict the stress field. The effect of training data size on performance is studied. With the increase of the training data size, both MSE and MAE decline gradually. However, as the magnitude of MAE is relatively small with respect to the magnitudes of the stress fields, all the computed stress fields across different training data size in this work are desirably very similar.

The effect of the hierarchical nature of the deep network on prediction accuracy is studied. The results indicate that the input channel number does not have a significant effect on the prediction accuracy. As the hierarchical architecture becomes deeper within certain limits, its prediction becomes more accurate. For better prediction accuracy, SE-Res-FCN is a reasonable choice. It encodes and preserves more information of the input and enables a richer set of non-linear operations due to its architecture. Future work will be to improve the network to an increasingly more general method using generative deep learning to alleviate the need for extensive coverage of the boundary conditions, loads, and input geometry with the original data set.

## Reference

[1] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, Nature, 521 (2015) 436.
[2] A.G. Roy, S. Conjeti, S.P.K. Karri, D. Sheet, A. Katouzian, C. Wachinger, N. Navab, ReLayNet: retinal layer and fluid segmentation of macular optical coherence tomography using fully convolutional networks, Biomedical optics express, 8 (2017) 3627-3642.
[3] A.T. Mohan, D.V. Gaitonde, A Deep Learning based Approach to Reduced Order Modeling for Turbulent Flow Control using LSTM Neural Networks, arXiv preprint arXiv:1804.09269, (2018).
[4] A.B. Farimani, J. Gomes, V.S. Pande, Deep learning the physics of transport phenomena, arXiv preprint arXiv:1709.02432, (2017).
[5] B. Kim, V.C. Azevedo, N. Thuerey, T. Kim, M. Gross, B. Solenthaler, Deep Fluids: A Generative Network for Parameterized Fluid Simulations, arXiv preprint arXiv:1806.02071, (2018).
[6] N. Umetani, Exploring generative 3D shapes using autoencoder networks, in: Proc. SIGGRAPH Asia 2017 Technical Briefs, ACM Bangkok, Thailand, 2017, 1-4.
[7] Y. Yu, T. Hur, J. Jung, Deep learning for topology optimization design, arXiv preprint arXiv:1801.05463, (2018).
[8] W. Zhang, H. Jiang, Z. Yang, S. Yamakawa, K. Shimada, L.B.J.a.p.a. Kara, Data-driven Upsampling of Point Clouds, (2018).
[9] E. Ulu, R. Zhang, L.B. Kara, A data-driven investigation and estimation of optimal topologies under variable loading configurations, Computer Methods in Biomechanics and Biomedical Engineering: Imaging & Visualization, 4 (2016) 61-72.

[10] M. Raissi, P. Perdikaris, G.E. Karniadakis, Multistep Neural Networks for Data-driven Discovery of Nonlinear Dynamical Systems, arXiv preprint arXiv:1801.01236, (2018).
[11] A. Geiger, P. Lenz, R. Urtasun, Are we ready for autonomous driving? the kitti vision benchmark suite, in: Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on, IEEE, 2012, pp. 3354-3361.
[12] J. Levinson, J. Askeland, J. Becker, J. Dolson, D. Held, S. Kammel, J.Z. Kolter, D. Langer, O. Pink, V. Pratt, Towards fully autonomous driving: Systems and algorithms, in: Intelligent Vehicles Symposium (IV), 2011 IEEE, IEEE, 2011, pp. 163-168.
[13] H. Lipson, M. Kurman, Driverless: intelligent cars and the road ahead, Mit Press, 2016.
[14] G.B. Goh, N.O. Hodas, A. Vishnu, Deep learning for computational chemistry, Journal of computational chemistry, 38 (2017) 1291-1307.
[15] A. Mardt, L. Pasquali, H. Wu, F. Noé, VAMPnets for deep learning of molecular kinetics, Nature communications, 9 (2018) 5.
[16] G. Montavon, M. Rupp, V. Gobre, A. Vazquez-Mayagoitia, K. Hansen, A. Tkatchenko, K.-R. Müller, O.A. Von Lilienfeld, Machine learning of molecular electronic properties in chemical compound space, New Journal of Physics, 15 (2013) 095003.
[17] G.A. Tribello, M. Ceriotti, M. Parrinello, A self-learning algorithm for biased molecular dynamics, Proceedings of the National Academy of Sciences, 107 (2010) 17509-17514.
[18] P. Broecker, J. Carrasquilla, R.G. Melko, S. Trebst, Machine learning quantum phases of matter beyond the fermion sign problem, Scientific reports, 7 (2017) 8823.
[19] K.T. Schütt, F. Arbabzadah, S. Chmiela, K.R. Müller, A. Tkatchenko, Quantum-chemical insights from deep tensor neural networks, Nature communications, 8 (2017) 13890.
[20] 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.
[21] A.B. Farimani, J. Gomes, R. Sharma, F.L. Lee, V.S. Pande, Deep Learning Phase Segregation, arXiv preprint arXiv:1803.08993, (2018).
[22] G. Yagawa, H. Okuda, Neural networks in computational mechanics, Archives of Computational Methods in Engineering, 3 (1996) 435.
[23] A. Oishi, G. Yagawa, Computational mechanics enhanced by deep learning, Computer Methods in Applied Mechanics and Engineering, 327 (2017) 327-351.
[24] Y. LeCun, Y. Bengio, Convolutional networks for images, speech, and time series, 1995.
[25] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, in: Advances in neural information processing systems, 2012, pp. 1097-1105.
[26] J. Johnson, A. Alahi, L. Fei-Fei, Perceptual losses for real-time style transfer and super-resolution, in: European Conference on Computer Vision, Springer, 2016, pp. 694-711.

[27] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 770-778.
[28] J. Hu, L. Shen, G. Sun, Squeeze-and-excitation networks, arXiv preprint arXiv:1709.01507, 7 (2017).
[29] J. Gómez, N. Guarín-Zapata, SolidsPy: 2D-Finite Element Analysis with Python, in, 2018.
[30] S. Ioffe, C. Szegedy, Batch normalization: Accelerating deep network training by reducing internal covariate shift, arXiv preprint arXiv:1502.03167, (2015).
[31] D. Kingma, J.A. Ba, A method for stochastic optimization. arXiv 2014, arXiv preprint arXiv:1412.6980., (2014).
[32] J. Duchi, E. Hazan, Y. Singer, Adaptive subgradient methods for online learning and stochastic optimization, Journal of Machine Learning Research, 12 (2011) 2121-2159.
[33] T. Tieleman, G. Hinton, Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude, in: COURSERA: Neural networks for machine learning, 2012, pp. 26-31.
[34] I. Goodfellow, Y. Bengio, A. Courville, Y. Bengio, Deep learning, MIT press Cambridge, 2016.