Stress Field Prediction in Cantilevered Structures Using Convolutional Neural Networks

Zhenguo Nie
Department of Mechanical Engineering,
Carnegie Mellon University,
Pittsburgh, PA 15213
e-mail: zhenguon@andrew.cmu.edu

Haoliang Jiang
Department of Mechanical Engineering,
Carnegie Mellon University,
Pittsburgh, PA 15213
e-mail: haolianj@andrew.cmu.edu

Levent Burak Kara[1]
Department of Mechanical Engineering,
Carnegie Mellon University,
Pittsburgh, PA 15213
e-mail: lkara@cmu.edu

The demand for fast and accurate structural analysis is becoming increasingly more prevalent with the advance of generative design and topology optimization technologies. As one step toward accelerating structural analysis, this work explores a deep learning-based approach for predicting the stress fields in 2D linear elastic cantilevered structures subjected to external static loads at its free end using convolutional neural networks (CNNs). Two different architectures are implemented that take as input the structure geometry, external loads, and displacement boundary conditions, and output the predicted von Mises stress field. The first is a single input channel network called SCSNet as the baseline architecture, and the second is the multichannel input network called StressNet. Accuracy analysis shows that StressNet results in significantly lower prediction errors than SCSNet on three loss functions, with a mean relative error of 2.04% for testing. These results suggest that deep learning models may offer a promising alternative to classical methods in structural design and topology optimization. Code and dataset are available.[2]
[DOI: 10.1115/1.4044097]

Keywords: deep learning, stress fields, CNN, StressNet

1 Introduction

While computational stress analysis is fundamentally critical in design and engineering, advances in automatic generative design systems impose higher demands on analysis speed without compromising accuracy. In this work, we explore the potential of data-driven stress analysis, where conventional run-time analysis is replaced by a machine learning system that can generate solutions instantaneously. Our approach is inspired by the great and increasing success of data-driven approaches that model physical phenomena and use the acquired knowledge to make predictions for unseen problems. Recent advances include data-driven approaches demonstrated in fluid dynamics [1–4], design and topology optimization [5–8], molecular dynamics simulation [9–12], and others [13–17].

In computational solid mechanics, there have been early attempts to use neural regression for finite element analysis (FEA) model updating [18,19]. More recently, deep learning has been gaining interest in the solution of traditional mechanics problems. Javadi and Tan [20] use a simple neural network in FEA as an alternative to the conventional constitutive material model. However, the geometry is abstracted and simplified into a feature vector, making the approach difficult to generalize to complex cases. Deep learning has also been integrated into FEA to optimize the numerical quadrature in the element stiffness matrix on a per-element basis [21]. Compared with the conventional numerical approach, this approach accelerates the calculation of the element stiffness matrix.

The use of convolutional networks, which use learned filters to understand image-based representations, has caught the attention of the academic community these years. Spruegel et al. [22] use a convolutional network as a classifier to check the plausibility of finite element (FE) simulations. In an inspiring work, Liang et al. [23] develop a three-module convolutional network for aortic wall stress prediction to accelerate the patient-specific FEA. The network takes as input the tube-shaped geometry and outputs the stress field. The description of geometry, boundary conditions, and loads is tailored for aorta models. We build upon these advances to demonstrate the potential of data-driven techniques for stress field predictions in a moderately more general case.

In this work, we present an end-to-end deep learning-based approach to predict the stress field in 2D linear elastic deformations. Two different architectures are explored that take as input the geometry, loads, and boundary conditions, and output the predicted von Mises stress field. The first architecture is a single-channel stress prediction neural network (SCSNet) where the loads are augmented with the feature representation (FR). The second is a multichannel stress prediction neural network (StressNet) with five separate channels of input including the geometry, load, and boundary conditions represented by images of identical sizes. In principle, StressNet can be extended to a generalized model for any arbitrary 2D conditions once given the corresponding training data, due to its on-limits multiple input channels.

This image-based problem representation allows the encoding of arbitrary 2D structures (within the prescribed domain resolution), boundary conditions, and external forces. As one step toward assessing the feasibility of such an approach, we focus on cantilevered structures with loads applied to the free end of the structure. A dataset involving 120,960 problems with variations in geometry and loads is generated to train and evaluate the networks. In our tests, StressNet with a deep architecture has a significantly higher accuracy over SCSNet, with a mean relative stress error of 2.04%. An in between network that combines elements of SCSNet and StressNet is also studied to assess the impact of single versus multichannel input representation.

2 Background and Related Work

Finite element analysis for stress computation. Stress analysis of a given structure requires the solution of related partial differential equations. FEA is the conventional approach to solve this problem. It simplifies the structure by breaking it down into a large number of FEs and computes the coupled mechanical deformations and stresses based on the boundary and load conditions by building up an algebraic equation:

$$Ku = F \quad (1)$$

---

[1]Corresponding author.

where K is the global stiffness matrix, F is the vector describing the applied external nodal forces, and u denotes the nodal displacement vector. To compute displacements, the global stiffness matrix has to be first assembled.

We assume linear isotropic materials and small deformations in a 2D field. The elemental stiffness matrix $\boldsymbol{K}_e$ can be computed as follows:

$$\boldsymbol{K}_e = A_e \boldsymbol{B}_e^T \boldsymbol{C}_e \boldsymbol{B}_e \qquad (2)$$

where A_e is the area of the element, $\boldsymbol{B}_e$ is the strain–displacement matrix that depends only on the element's rest shape, and $\boldsymbol{C}_e$ is the elasticity tensor constructed using Young's modulus and Poisson's ratio of the base material. Given a mesh $\mathbb{V}$ with m elements, one can assemble the global stiffness matrix $\boldsymbol{K}$ in order to determine the displacements $\boldsymbol{u}$ from Eq. (1) through a proper application of the zero displacement boundary conditions and the external nodal forces. Then, the stress–displacement relationship can be written as

$$\boldsymbol{\sigma} = \boldsymbol{C}_g \boldsymbol{B} \boldsymbol{u} \qquad (3)$$

where $\boldsymbol{\sigma} \in \mathbb{R}^{4m}$ captures the unique four elements of the elemental stress tensor for planar stress calculations and $\boldsymbol{B}$ is the assembly of $\boldsymbol{B}_e$ matrices. Block-diagonal matrix $\boldsymbol{C}_g \in \mathbb{R}^{4m \times 4m}$ is constructed with elemental elasticity tensors $\boldsymbol{C}$ on the diagonal. For each element, $\boldsymbol{C}_e$ can be computed analogous to $\boldsymbol{K}_e$. While applicable to different element types, we use linear quad elements making $\boldsymbol{K} \in \mathbb{R}^{2n \times 2n}$, $\boldsymbol{u} \in \mathbb{R}^{2n}$, $\boldsymbol{f} \in \mathbb{R}^{2n}$, and $\boldsymbol{B} \in \mathbb{R}^{4m \times 2n}$ for a planar mesh with quad elements having n nodes.

Then, the von Mises Stress of each element is computed using 2D von Mises Stress form:

$$\sigma_{vm} = \sqrt{\sigma x^2 + \sigma_y^2 - \sigma_x \sigma_y + 3\tau_{xy}^2} \qquad (4)$$

where σ_{vm} is von Mises stress, σ_x and σ_y are the stress components in the x and y directions, respectively, and τ_{xy} is the shear stress.

The final stress distribution could be obtained after the stresses of all elements are computed. Several factors impact FEA's complexity including the number of elements and the degree of the elements, as they give rise to large stiffness matrices that need to be assembled from elemental stiffness matrices [23]. This leaves an opportunity for researchers to seek faster methods for inner loop simulations for structure design and optimization. This work explores the use of deep learning toward this goal.

Convolutional neural networks and ResNet. Convolutional neural networks (CNNs) are designed to process data that comes in the form of multiple arrays, for example, a color image composed of three 2D arrays containing pixel intensities in the three-color channels [24]. CNN consists primarily of two core structures: a convolutional layer and a pooling layer. In a convolutional layer, a filter bank with a set of weights slides over the input image to produce a feature map. After a convolutional layer, the result of the weighted sum is usually passed through a nonlinearity such as a ReLU function: $f(x) = \max(0, x)$, and then passed on to the pooling layer. The role of the pooling layer is to merge semantically similar features into one.

Inspired by the philosophy of VGG net [25], He et al. [26] propose a 152-layer Residual Network (ResNet) that won the first place in ILSVRC-2015 with an error rate of 3.6%. As shown in Fig. 1, the defining feature of ResNet is the shortcut connection added to each pair of 3×3 filters in the residual version. The shortcut connection simply performs identity mapping, and its output is added to the output of the stacked layers. To the extreme, if an identity mapping was optimal, it would be easier to push the residual to 0 than to fit an identity mapping by the stacked layers. It means ResNet can dynamically select the layer depth for the desired underlying mapping.

Deep learning in computational mechanics. The artificial neural networks with the multilayer perceptron (MLP) has been applied in computational mechanics for several years, for structural analysis
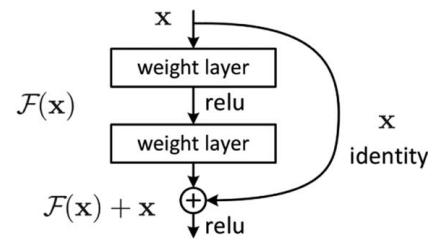

Fig. 1   ResNet: a building block with a shortcut connection [26]

[27–29], materials failure and damage [30–33], regression of the material constitutive properties [30,34–37], computational mechanics enhancement [21,20], topological design, and optimization [38,39]. However, due to the intrinsic limitation of MLP, the geometry is usually abstracted and simplified into a feature vector that loses the spatial position relationship, making the approach difficult to generalize to complex cases.

Spruegel et al. [22] use a convolutional network as a classifier to check the plausibility FE simulations. Similarly, Sosnovik and Oseledets [40] propose a convolutional encoder-decoder architecture to accelerate topology optimization methods. Khadilkar et al. [41] use CNN to predict the stress field for the bottom-up SLA 3D printing process. In an inspiring work, Liang et al. [23] develop a three-module convolutional network for aortic wall stress prediction to accelerate the patient-specific FEA. The network takes as input the tube-shaped geometry and outputs the stress field. However, as the description of the problem is tailored for aorta models, the network cannot be extended to arbitrary geometry, boundary conditions, and loads. We build upon these advances to demonstrate the potential of data-driven techniques for stress field predictions in a moderately more general case.

3   Method

This work presents a data-driven approach to stress field prediction in 2D cantilevered structures with a linear isotropic elastic material subjected to external loads at the structure's free end. The approach takes as input the structure geometry, external loads, and displacement boundary conditions, and outputs the predicted stress field. Based on the input channel configuration, two deep neural network architectures are proposed: (a) SCSNet with a single input channel and (b) StressNet with multiple input channels.

3.1   Problem Description and Dataset. Consider the cantilevered structure in Fig. 2 composed of a homogeneous and isotropic linear elastic material. The left end of the structure is affixed to the wall and the right end bears the external static loads. The evenly

Fig. 2   The schematic diagram of a two-dimensional cantilevered structure with linear isotropic material

distributed external loads are applied in both the horizontal (q_x) and vertical (q_y) directions. For each sample, the loads (q_x, q_y) are constant due to the static problem, and determined by the resultant force q and its direction θ : $q_x = q \cos \theta$ and $q_y = q \sin \theta$. The geometry is not limited to a rectangle, but other shapes such as the trapezoid, the trapezoid with curved sides, and all the above structures with holes are utilized. In total, there are 28 geometries within such six categories with the change of geometry contour, hole shape, size, and location. Theoretically speaking, the eligible cantilevered structures are infinite, so that we cannot enumerate them all. What we select are common structures in mechanical engineering. In addition, the material properties keep unchanged and isotropic for all samples.

This two-dimensional elastic deformation is a plane strain problem. The governing equations consist of the strain–displacement equation (5), compatibility equation (6), equilibrium equation (7), and the generalized Hooke's law in Eq. (8):

$$\varepsilon_{ij} = \frac{1}{2}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right) \tag{5}$$

$$\frac{1}{2}\left(\frac{\partial^2 \varepsilon_{11}}{\partial x_2^2} + \frac{\partial^2 \varepsilon_{22}}{\partial x_1^2}\right) = \frac{\partial^2 \varepsilon_{12}}{\partial x_2 \partial x_1} \tag{6}$$

$$\frac{\partial \sigma_{ii}}{\partial x_i} + \frac{\partial \sigma_{ij}}{\partial x_j} + f_i = 0 \tag{7}$$

$$\begin{bmatrix} a_{11} \\ a_{21} \\ a_{31} \end{bmatrix} = \frac{E}{(1-\nu)(1-2\nu)} \begin{bmatrix} 1-\nu & 0 & 0 \\ 0 & 1-\nu & 0 \\ 0 & 0 & 1-2\nu \end{bmatrix} \begin{bmatrix} \varepsilon_{11} \\ \varepsilon_{22} \\ \varepsilon_{12} \end{bmatrix} \tag{8}$$

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 the x_i direction, ε_{ij} is strain on surface x_i in the x_j direction, σ_{ij} is stress on surface x_i in the x_j direction, f_i is the body force component in the x_i direction, E is Young's modulus, and ν is Poisson's ratio.

A 2D finite element method (FEM) software SolidsPy [42] is used to generate the training and testing data. The full domain contains 32×24 elements. Any element in the domain is a 4-node quadrilateral with a size of 1×1 (mm). Randomly selected samples from a total of 120,960 samples are shown in Fig. 3. Each case is represented as a 32×24 image. Images in the left column are the input for the single channel and the images on the right are the corresponding von Mises ground truth stress fields. For input images, the solid domain traverses the whole domain from left to right, and the external loads are applied on the right-hand edge. In dataset generation, the resultant load q ranges from 0 to 100 N in an interval of 20 N, and the load direction θ ranges from 0 to 2π in an interval of $\pi/12$. This information delineated with distinct (q_x, q_y) in the input data. For the ground truth fields, the von Mises stress within the solid domain is plotted as the stress field where the stress is absolute zero within the void domain. Among all computed FEM samples, the von Mises stress varies from 0 to 2475 MPa with an average of 67.80 MPa.

Based on the computation of FEM, the multichannel dataset for StressNet can be transformed from the single-channel dataset for SCSNet. As shown in Fig. 4, all the datasets are encoded as matrices and displayed as colorful images. Firstly, the geometry channel is encoded as a 32×24 matrix containing just zeros and ones and shown as a binary-color image. As a single channel geometry, the solid (1 for solid) and void (0 for void) parts of the domain are distinguished. Secondly, for the two load channels, the component magnitude of the force is located in the matrix at the force applied location (force magnitudes on the right-hand edge and 0 on other areas). Thirdly, for the two displacement boundary condition channels, all displacement-constrained points are delineated in their respective binary matrices (−1 for the
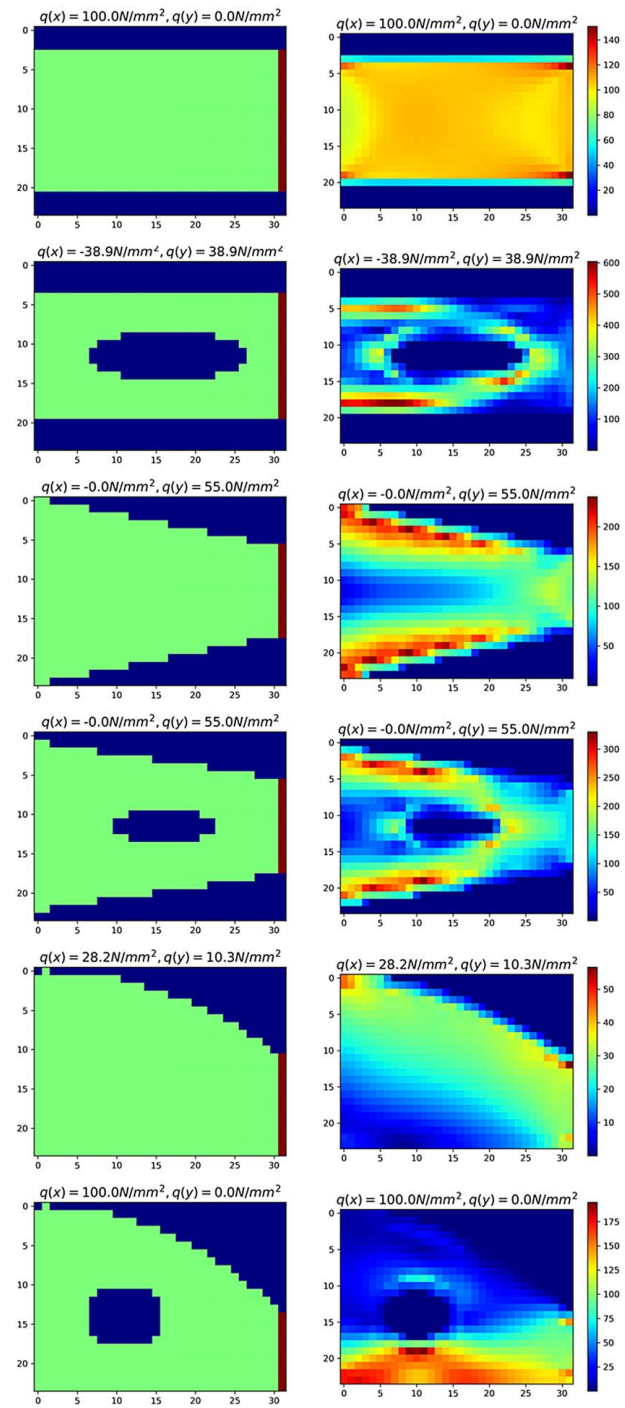

Fig. 3 Samples of the dataset computed by FEM. Each row is a FEM sample. Images in the left column are input channels including geometry and load position; Right images are stress fields (units: mm-MPa-N).

constrained boundary and 0 for the free boundary). Finally, the von Mises stress field is also a 32×24 matrix and displayed as an image.

3.2 SCSNet. As illustrated in Fig. 5, a single-channel is proposed to perform the prediction of the von Mises stress field. The input of the model is a matrix including the information of the geometry and shown in one image. The output of the model is the computed stress field, where each pixel represents the von Mises stress.
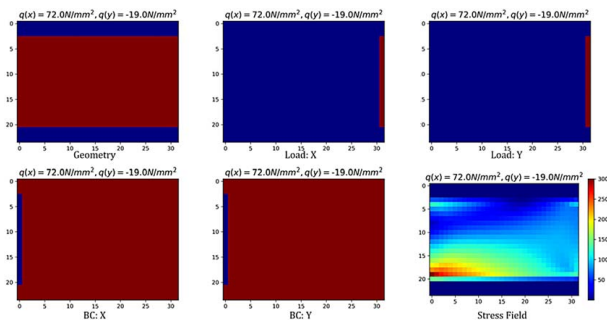
Fig. 4 An example of five-channel data representation and the output stress field
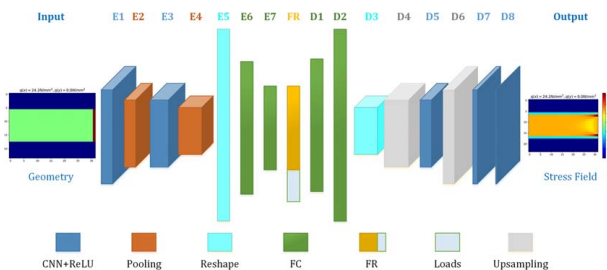

Fig. 5 The architecture of SCSNet with a single input channel

SCSNet is a baseline architecture that uses multiple CNN layers in an encoder-decoder structure (convolutional autoencoder). This convolutional autoencoder learns to encode the input in a set of simple signals and then tries to reconstruct the input from them [43–45]. The encoder network consists of two convolutional layers (E1 and E3) and two max-pooling layers (E2 and E4). Each convolutional layer has a filter with a kernel of 3×3 and a stride of 1×1. 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 obtain the latent FR of geometry at the bottleneck. The load vector (q_x, q_y) is concatenated with FR before the new FR is fed into the decoder network. The decoder network is the reverse of the encoder. Upsampling layers take the place of pooling layers for increasing the field resolution. The entire model contains a total of 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 [46,47].

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 $24 \times 32 \times 1$.

3.3 StressNet. SCSNet has constant CNN layers and can only model simple FEM problems, in which the loads are distributed uniformly and displacement boundary conditions are the same in the X and Y directions. To decrease the prediction error and increase its versatility, we additionally propose a StressNet architecture with multiple input channels as shown in Fig. 6. A downsampling-and-upsampling structure is employed, and five squeeze-and-excitation (SE) ResNet modules are used between the downsampling and upsampling structures. As mentioned before, ResNet can dynamically select the layer depth by shortcut connections. All convolutional layers are followed by batch normalization and ReLU layers.

The input data contains five channels: (1) geometry, (2) X-component of the load, (3) Y-component of the load, (4) X-component
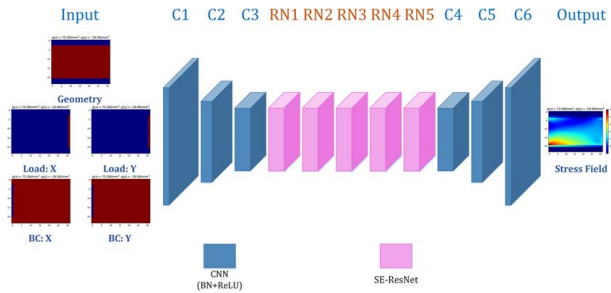

Fig. 6 The architecture of StressNet with multiple input channels

of the displacement boundary condition, and (5) Y-component of the displacement boundary condition. For each channel, the information is encoded in a two-dimensional 32×24 matrix. As with SCSNet, the output data of StressNet are also a single-channel von Mises stress field that is encoded in a 32×24 matrix and displayed as a colorful image.

Downsampling comprises three convolutional layers (C1, C2, and C3) and upsampling comprises three deconvolutional layers (C4, C5, and C6). Referring to the structure of image transformation networks [48], we use 9×9 kernels in the first and last layers (C1 and C6) and 3×3 kernels in all the other convolutional layers.

Residual blocks are used to mimic identical layers to combat the vanishing gradient problem [26]. As shown in Fig. 7, an SE-ResNet module is composed of two convolutional layers with 3×3 kernels and one SE network block. The output of the SE-ResNet module, as shown in Eq. (9), can be conducted by feedforward neural networks with shortcut connections.

SE blocks, illustrated in Fig. 8, 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 [49]. The input data $u \in R^{H \times W \times C}$ are shrunk into $\mathbb{S}(u) \in R^C$ 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 $\mathbb{E}(u)$ 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 Eq. (10)

$$z = \mathbb{F}(x, \{w_i\}) + x \tag{9}$$

$$v = \mathbb{E}(u) \times u \tag{10}$$

3.4 Loss Function and Metrics. We use the mean squared error (MSE) and mean absolute error (MAE) to evaluate the
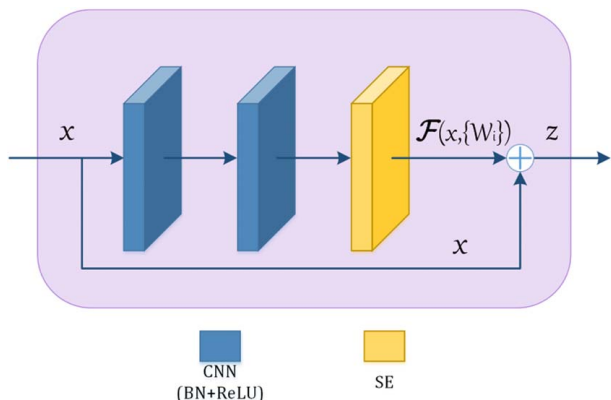

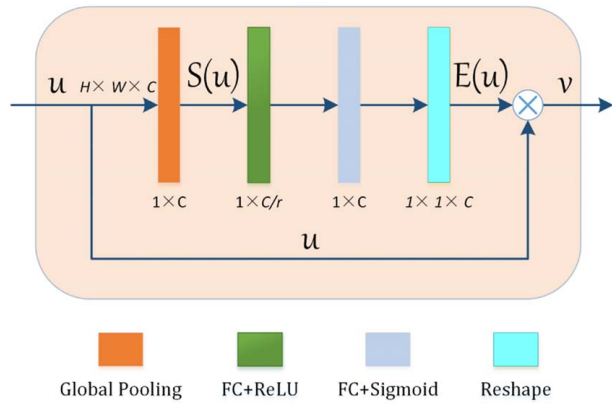**Fig. 7 The residual block with an SE block**
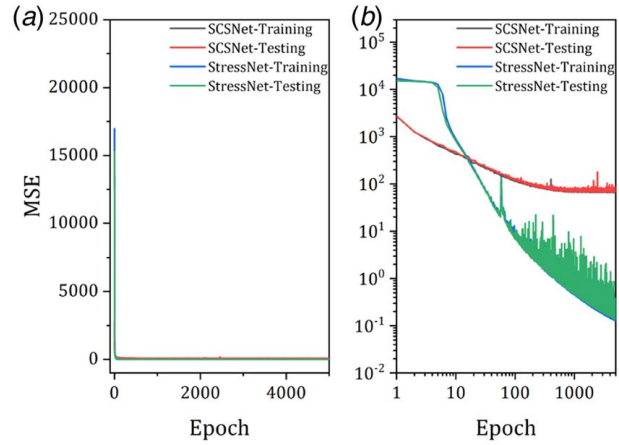
Fig. 8 The SE block

Fig. 9 MSE curves on training and testing data of two architectures: (a) is shown in arithmetic coordinates and (b) is shown in logarithmic coordinates
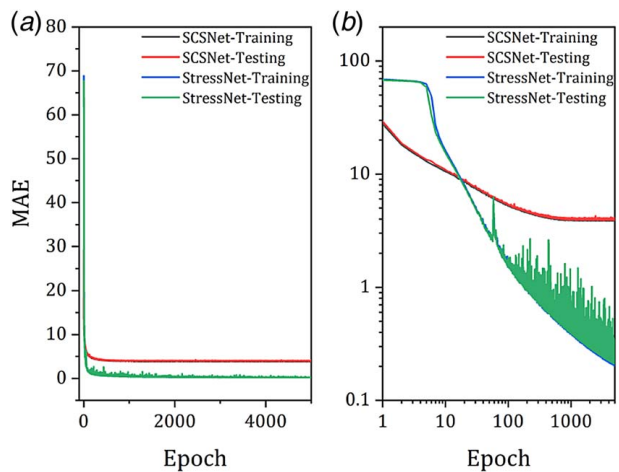
Fig. 10 MAE curves on training and testing data of two architectures: (a) is shown in arithmetic coordinates and (b) is shown in logarithmic coordinates

model's prediction accuracy. The prediction $\hat{y}$ and ground truth y in our current model are both displayed as 32×24 resolution images. Before computing MSE and MAE, each predicted stress field is reshaped into 1D-arrays with a length of 768. The reshaped prediction $\hat{y}$ can be expressed as $\hat{y} = (\hat{y_1}, \hat{y_2}, \ldots, \hat{y_n})$, while the reshaped ground truth $y = (y_1, y_2, \ldots, y_n)$. MSE and MAE are, respectively, represented in Eqs. (11) and (12)

$$\text{MSE} = \frac{1}{n}\sum_{j=1}^{n}(y_j - \hat{y_j})^2 \qquad (11)$$

$$\text{MAE} = \frac{1}{n}\sum_{j=1}^{n}|y_j - \hat{y_j}| \qquad (12)$$

where $n = 768$ is the total number of elements.

Here, we also introduce the mean relative error (MRE) as shown in Eq. (13). MRE is a relative error rate in percentage to measure how close the predictions are to the ground truth.

$$\text{MRE} = \frac{1}{n}\sum_{j=1}^{n}\frac{|y_j - \hat{y_j}|}{\epsilon + \max(y_j, \hat{y_j})} \times 100\% \qquad (13)$$

where ϵ is a smoothing term that avoids division by zero (this article takes $\epsilon = 0.01$).

Based on the Cauchy–Schwartz inequality, $\text{MSE} \leq \text{MAE}^2$. MSE has a tendency to be increasingly larger than MAE^2 with an increase of testing sample and is more sensitive to data variance. In our approach, MSE is used for the training loss, while all three error measures are used to quantify the prediction performance.

4 Results and Discussions

All code is written in TensorFlow and run on an NVIDIA GeForce GTX 1080Ti GPU. The Adam optimization algorithm with an exponentially decaying learning rate is used for training, and the batch size is set to 256, which is the allowed maximum size according to the GPU memory. The training and testing results show that both the two architectures are stable and converged reliably. Under our experiment scenarios, it takes 1.56 s for SCSNet to render all the 120,960 FEM samples, and 10.4 s for StressNet. The FEM software takes approximately ten hours (on Intel i7-6500U CPU) to accomplish the FEM computation on the same number of 2D problems.

4.1 Accuracy and Performance. In this experiment, we train and evaluate our models using the whole dataset. The training data size is 100 000, and the separate testing data size is 20,960. Figure 9 shows MSE loss 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 50 epochs, and then begin to flatten. From chart (b), it can be seen that SCSNet preserves a nearly constant order of magnitude after 1000 epochs. By contrast, StressNet continues to decrease with more training, even after 5000 epochs. StressNet has a significantly smaller MSE than SCSNet in the end. Figure 10 shows the MAE loss on training data and testing data of the two architectures. The trend in the MAE loss is nearly identical to that of MSE. StressNet has much better accuracy than SCSNet. The other performance metrics are summarized in Table 1. It can be seen that MRE of StressNet is just 2.04% for testing, which we deem acceptable for stress fields prediction.

The evolutive predictions by StressNet with the increasing epoch during the training are plotted in Fig. 11. Each row denotes a randomly selected sample from the whole datasets. The first column is the ground truth. Images starting from the second column, in

Table 1 Error metrics. Epoch = 5000, $\bar{y} = 67.80$. SCS denotes SCSNet and SN denotes StressNet.

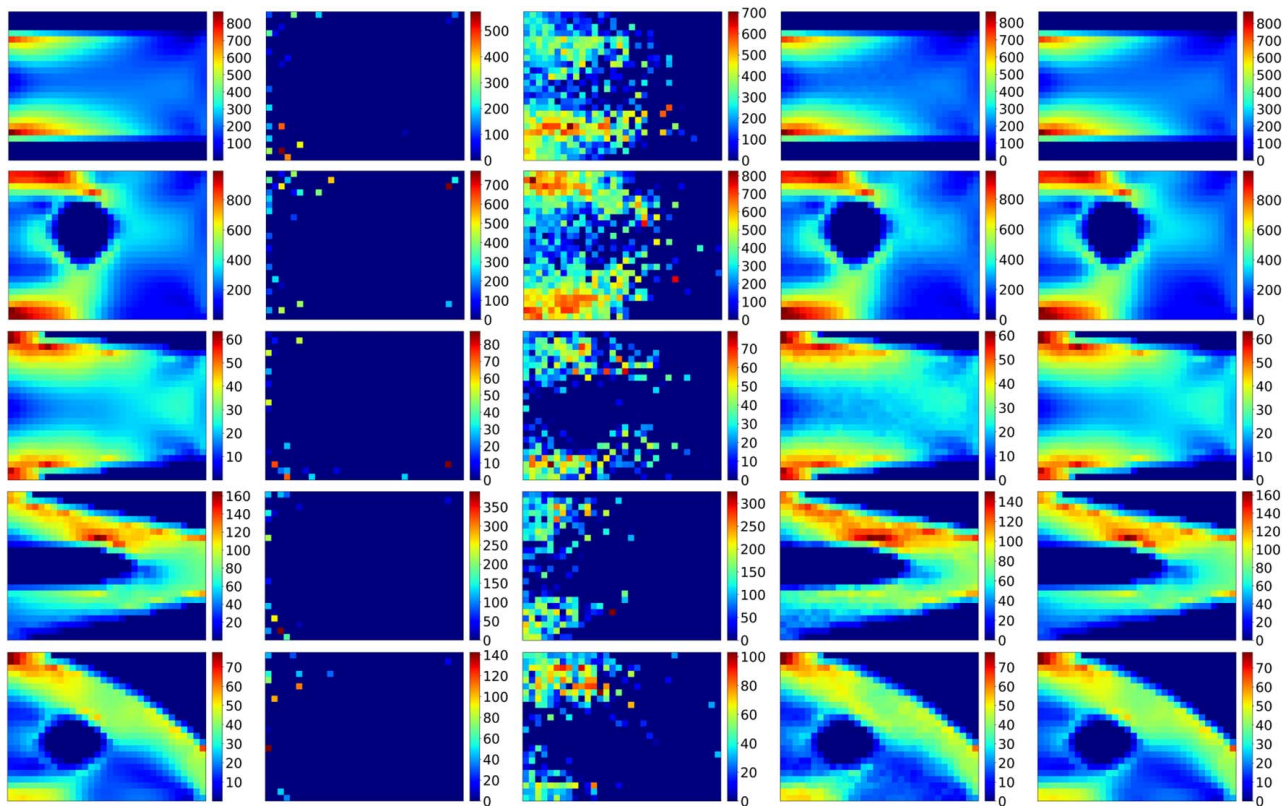| Metric	MSE (MPa^2)		MAE (MPa)		MRE (%)	
Model	SCS	SN	SCS	SN	SCS	SN
Training	83.63	0.14	4.28	0.22	10.40	1.99
Testing	84.07	0.15	4.30	0.23	10.43	2.04

Fig. 11 Computed stress fields by StressNet: ground truth and evolutive predictions. Each row denotes a randomly selected sample from the whole datasets. From left to right: (1) ground truth, (2) epoch = 1, (3) epoch = 10, (4) epoch = 100, and (5) epoch = 5000. The ratio of training data and testing data is 100,000/20,960.

order, are evolutive predictions of the von Mises stress fields. Predictions in the first epoch just contain a few nonzero scattering points, and then in the tenth epoch have a rudimentary form, especially on the left half. Since from the hundredth epoch, the predictions have no visual distinction with the ground truth.

4.2 Effect of the Training Data Size on Performance.
Besides using all the data for training and testing with the ratio of 100,000/20,960, we also reduce the training data size from 100,000 to 20,000 to demonstrate the effect of training size. Figure 12 shows the variations of MSE and MAE for both training and testing with different training sizes. The variations of MSE and MAE follow similar trends. With the increase in the training data, both MSE and MAE decline gradually. The standard deviation across multiple runs at a fixed training data size (error bar) also decreases with increasing training data. MSE and MAE of the testing data are expectantly slightly larger than those of the training data. It means that the larger training data, the better prediction performance. Similarly, we randomly plot the computed stress fields with different training data size as shown in Fig. 12. By using different sizes of the training data, StressNet training is terminated at the five thousandth epoch. For each sample in Fig. 13, the stress fields predicted by StressNet look like the same with different training data sizes. However, they have different color bars, which means the outputs are similar but still different.

4.3 Prediction of Maximum Stress. As a measure of how well the results of StressNet can be used for predicting failure, we also examine how well StressNet can predict the maximum von Mises stress obtained in the ground truth. An analysis of the testing data shows a coefficient of determination R^2 of 0.99. This implies that StressNet is able to predict the maximum stress with significant accuracy.

4.4 Effect of the Hierarchical Nature of Deep Learning on Prediction Accuracy. In addition to the two architectures described above, we also deployed and studied an inbetween architecture shown in Fig. 14 which keeps the SCSNet structure but uses StressNet's multiple channeled inputs. The five channels are classified into three categories: geometry, loads, and boundary conditions similar to the way described for StressNet. All three encoders are independent and parallel to each other. After the fully connected layers, three FRs 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 StressNet.
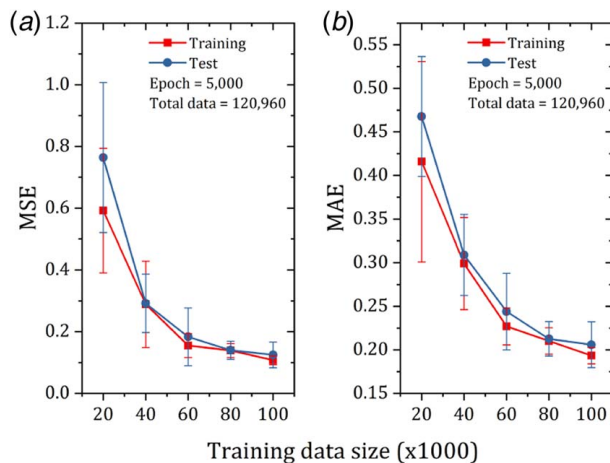

Fig. 12 Effect of training data size on the prediction performance of StressNet
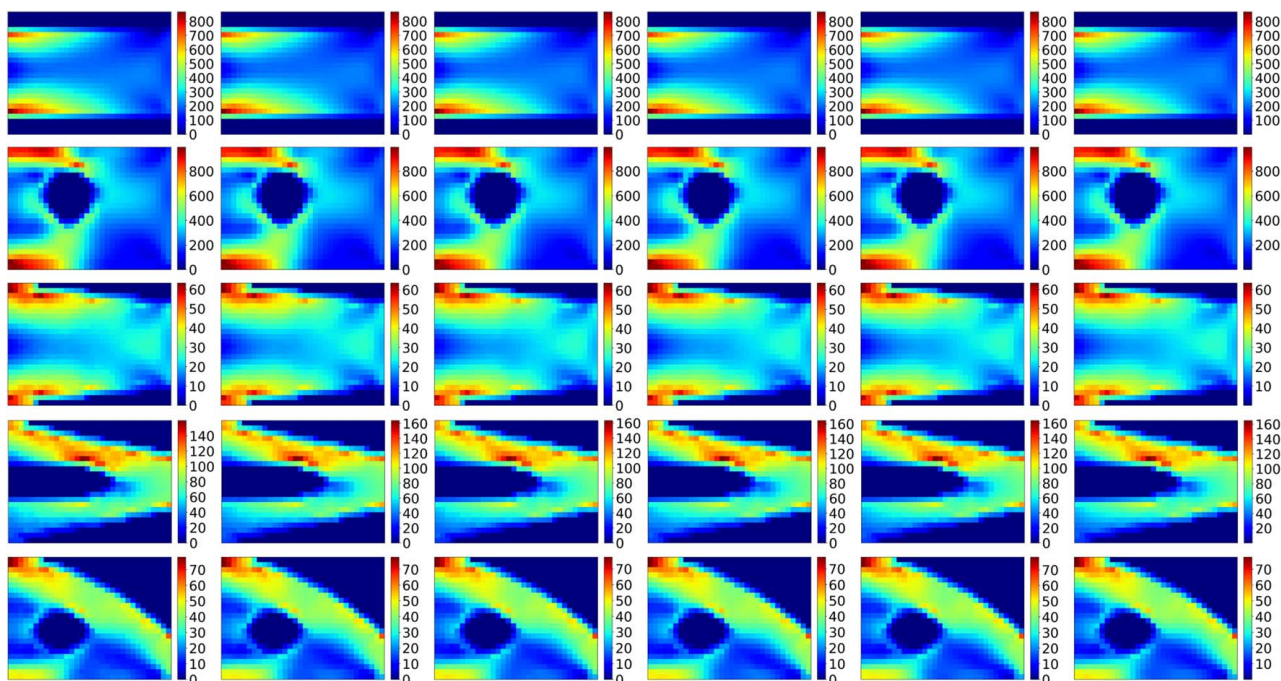
Fig. 13 Computed stress fields with different training data sizes by StressNet. From left to right, the training data size is: (1) 20,000, (2) 40,000, (3) 60,000, (4) 80,000, (5) 100,000, and (6) ground truths.
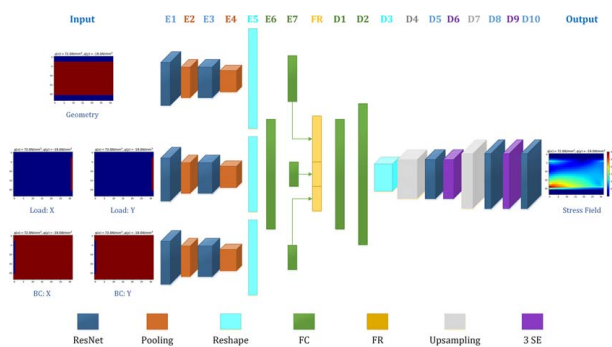

Fig. 14 An inbetween architecture with representation fusion and multiple input channels

We trained this model on the same GPU, and the training result shows that this model has no improvement in training accuracy relative to SCSNet. This supports our hypothesis that as the hierarchical architecture becomes deeper as it does with StressNet, its prediction becomes more accurate, although this deepening has to be judicious in light of the image resolution, training size, and data variability. Additionally, the input channel configuration may not have a significant impact on prediction accuracy.

Comparatively speaking, SCSNet can reduce the training time, and reach an acceptable accuracy where the MRE is 10.40%. In addition, due to its fully convolutional network architecture, it is able to account for large variations in the size of the 2D FEM data. This makes StressNet a great alternative to classical FEM once the training data is sufficient to train the network.

5 Conclusion

In this paper, we present an end-to-end deep learning-based approach for stress field prediction in cantilevered structures. Two architectures are implemented: a single-channel stress prediction neural network (SCSNet) and a multichannel stress prediction neural network (StressNet) with SE-ResNet modules. A 2D FEM

software SolidsPy is used to generate the training and testing data, which contains a total of 120,960 FEM samples.

Both architectures are stable and converged reliably in training and testing. MSE and MAE results show that StressNet can obtain higher accuracy than SCSNet. The MRE of the StressNet model is just 2.04% for testing with respect to the ground truth. The effect of training data size on performance is also studied. With the increase in 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 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 hierarchical architecture becomes deeper, its prediction becomes more accurate. For better prediction accuracy, StressNet is a reasonable choice. It encodes and preserves more information of the input and enables a richer set of nonlinear 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 input geometry, boundary conditions, and loads.

References

[1] Roy, A. G., Conjeti, S., Karri, S. P. K., Sheet, D., Katouzian, A., Wachinger, C., and Navab, N., 2017, "Relaynet: Retinal Layer and Fluid Segmentation of Macular Optical Coherence Tomography Using Fully Convolutional Networks," Biomed. Opt. Express, **8**(8), pp. 3627–3642.

[2] Mohan, A. T., and Gaitonde, D. V., 2018, "A Deep Learning based Approach to Reduced Order Modeling for Turbulent Flow Control using LSTM Neural Networks," e-print arXiv:1804.09269.

[3] Farimani, A. B., Gomes, J., and Pande, V. S., 2017, "Deep Learning the Physics of Transport Phenomena," CoRR, abs/1709.02432.

[4] Kim, B., Azevedo, V. C., Thuerey, N., Kim, T., Gross, M. H., and Solenthaler, B., 2018, "Deep Fluids: A Generative Network for Parameterized Fluid Simulations," CoRR, abs/1806.02071.

[5] Umetani, N., 2017, "Exploring Generative 3D Shapes Using Autoencoder Networks," SIGGRAPH Asia 2017 Technical Briefs, ACM, pp. 1–24.

[6] Yu, Y., Hur, T., and Jung, J., 2018, "Deep Learning for Topology Optimization Design," CoRR, abs/1801.05463.

[7] Zhang, W., Jiang, H., Yang, Z., Yamakawa, S., Shimada, K., and Kara, L. B., 2018, "Data-Driven Upsampling of Point Clouds," CoRR, abs/1807.02740.

[8] Ulu, E., Zhang, R., Yumer, M. E., and Kara, L. B., 2014, "A Data-Driven Investigation and Estimation of Optimal Topologies Under Variable Loading Configurations," Computational Modeling of Objects Presented in Images. Fundamentals, Methods, and Applications, Springer International Publishing, Berlin, pp. 387–399.

[9] Goh, G. B., Hodas, N. O., and Vishnu, A., 2017, "Deep Learning for Computational Chemistry," J. Comput. Chem., **38**(16), pp. 1291–1307.

[10] Mardt, A., Pasquali, L., Wu, H., and Noé, F., 2018, "VAMPnets for Deep Learning of Molecular Kinetics," Nat. Commun., **9**(Jan), p. 5.

[11] Montavon, G., Rupp, M., Gobre, V., Vazquez-Mayagoitia, A., Hansen, K., Tkatchenko, A. R., Müller, K.-R., and Anatole von Lilienfeld, O., 2013, "Machine Learning of Molecular Electronic Properties in Chemical Compound Space," New. J. Phys., **15**(Sep), p. 095003.

[12] Tribello, G. A., Ceriotti, M., and Parrinello, M., 2010, "A Self-learning Algorithm for Biased Molecular Dynamics," Proc. Natl. Acad. Sci. U. S. A., **107**(41), pp. 17509–17514.

[13] Raissi, M., Perdikaris, P., and Karniadakis, G., 2018, "Multistep Neural Networks for Data-Driven Discovery of Nonlinear Dynamical Systems." arXiv preprint arXiv:1801.01236.

[14] Broecker, P., Carrasquilla, J., Melko, R. G., and Trebst, S., 2017, "Machine Learning Quantum Phases of Matter Beyond the Fermion Sign Problem," Scientific Reports.

[15] Schütt, K. T., Arbabzadah, F., Chmiela, S., Müller, K. R., and Tkatchenko, A., 2017, "Quantum-chemical Insights From Deep Tensor Neural Networks," Nat. Commun., **8**(Jan), p. 13890.

[16] Barati Farimani, A., Gomes, J., Sharma, R., Lee, F. L., and Pande, V. S., 2018, "Deep Learning Phase Segregation," e-print arXiv:1803.08993.

[17] Zhang, W., Jiang, H., Yang, Z., Yamakawa, S., Shimada, K., and Kara, L. B., 2018, "Data-Driven Upsampling of Point Clouds," CoRR, abs/1807.02740.

[18] Levin, R., and Lieven, N., 1998, "Dynamic Finite Element Model Updating Using Neural Networks," J. Sound Vib., **210**(5), pp. 593–607.

[19] Atalla, M., and Inman, D., 1998, "On Model Updating Using Neural Networks," Mech. Syst. Signal Process., **12**(1), pp. 135–161.

[20] Javadi, A., and Tan, P.T., 2003, "Neural Network for Constitutive Modelling in Finite Element Analysis," Compute. Assist. Mech. Eng. Sci., **10**(4), pp. 523–530.

[21] Oishi, A., and Yagawa, G., 2017, "Computational Mechanics Enhanced by Deep Learning," Comput. Methods Appl. Mech. Eng., **327**, pp. 327–351.

[22] Spruegel, T., Schröppel, T., and Wartzack, S., 2017, "Generic Approach to Plausibility Checks for Structural Mechanics with Deep Learning." Proceedings of the 21st International Conference on Engineering Design (ICED 17) Vol 1: Resource Sensitive Design, Design Research Applications and Case Studies, Vancouver, Canada, Aug. 21–25, pp. 299–308

[23] Liang, L., Liu, M., Martin, C., and Sun, W., 2018, "A Deep Learning Approach to Estimate Stress Distribution: a Fast and Accurate Surrogate of Finite-Element Analysis," J. R. Soc. Interface, **15**(138), pp. 20170844.

[24] LeCun, Y., Bengio, Y., and Hinton, G., 2015, "Deep Learning," Nature, **521**(05), pp. 436–444.

[25] Simonyan, K., and Zisserman, A., 2015, "Very Deep Convolutional Networks for Large-Scale Image Recognition," International Conference on Learning Representations, San Diego, CA, May 7–9.

[26] He, K., Zhang, X., Ren, S., and Sun, J., 2015, "Deep Residual Learning for Image Recognition," CoRR, abs/1512.03385.

[27] Jenkins, W., 1995, "Neural Network-Based Approximations for Structural Analysis," *Developments in Neural Networks and Evolutionary Computing for Civil and Structural Engineering*, Civil-Comp Press, Edinburgh, pp. 25–35.

[28] Waszczyszyn, Z., and Ziemiański, L., 2001, "Neural Networks in Mechanics of Structures and Materials–new Results and Prospects of Applications," Comput. Struct., **79**(22–25), pp. 2261–2276.

[29] Goh, G. B., Hodas, N. O., and Vishnu, A., 2017, "Deep Learning for Computational Chemistry," J. Comput. Chem., **38**(16), pp. 1291–1307.

[30] Abendroth, M., and Kuna, M., 2003, "Determination of Deformation and Failure Properties of Ductile Materials by Means of the Small Punch Test and Neural Networks," Comput. Mater. Sci., **28**(3–4), pp. 633–644.

[31] Wu, X., Ghaboussi, J., and Garrett Jr, J., 1992, "Use of Neural Networks in Detection of Structural Damage," Comput. Struct., **42**(4), pp. 649–659.

[32] Zang, C., and Imregun, M., 2001, "Structural Damage Detection Using Artificial Neural Networks and Measured Frf Data Reduced Via Principal Component Projection," J. Sound Vib., **242**(5), pp. 813–827.

[33] Tsou, P., and Shen, M.-H., 1994, "Structural Damage Detection and Identification Using Neural Networks," AIAA J., **32**(1), pp. 176–183.

[34] Huber, N., and Tsakmakis, C., 1999, "Determination of Constitutive Properties From Spherical Indentation Data Using Neural Networks. Part Ii: Plasticity With Nonlinear Isotropic and Kinematic Hardening," J. Mech. Phys. Solids., **47**(7), pp. 1589–1607.

[35] Zhang, Z., and Friedrich, K., 2003, "Artificial Neural Networks Applied to Polymer Composites: a Review," Compos. Sci. Technol., **63**(14), pp. 2029–2044.

[36] Settgast, C., Abendroth, M., and Kuna, M., 2019, "Constitutive Modeling of Plastic Deformation Behavior of Open-cell Foam Structures Using Neural Networks," Mech. Mater., **131**, pp. 1–10.

[37] Liu, M., Liang, L., and Sun, W., 2019, "Estimation of in Vivo Constitutive Parameters of the Aortic Wall Using a Machine Learning Approach," Comput. Methods Appl. Mech. Eng., **347**, pp. 201–217.

[38] Yildiz, A., Öztürk, N., Kaya, N., and Öztürk, F., 2003, "Integrated Optimal Topology Design and Shape Optimization Using Neural Networks," Struct. Multidiscipl. Optim., **25**(4), pp. 251–260.

[39] Papadrakakis, M., and Lagaros, N. D., 2002, "Reliability-based Structural Optimization Using Neural Networks and Monte Carlo Simulation," Comput. Methods Appl. Mech. Eng., **191**(32), pp. 3491–3507.

[40] Sosnovik, I., and Oseledets, I., 2017, "Neural Networks for Topology Optimization," e-print arXiv:1709.09578.

[41] Khadilkar, A., Wang, J., and Rai, R., 2019, "Deep Learning–based Stress Prediction for Bottom-up Sla 3d Printing Process," Int. J. Adv. Manuf. Technol., **102**(5–8), pp. 2555–2569.

[42] Gómez, J., Guarín-Zapata, N., 2018, "SolidsPy: 2D-Finite Element Analysis With Python."

[43] Masci, J., Meier, U., Cireşan, D., and Schmidhuber, J., 2011, "Stacked Convolutional Auto-Encoders for Hierarchical Feature Extraction", *International Conference on Artificial Neural Networks*, Springer, pp. 52–59.

[44] Geng, J., Fan, J., Wang, H., Ma, X., Li, B., and Chen, F., 2015, "High-Resolution Sar Image Classification Via Deep Convolutional Autoencoders," IEEE Geosci. Remote Sens. Lett., **12**(11), pp. 2351–2355.

[45] Holden, D., Saito, J., Komura, T., and Joyce, T., 2015, "Learning Motion Manifolds with Convolutional Autoencoders," *SIGGRAPH Asia 2015 Technical Briefs*, ACM, New York, p. 18.

[46] LeCun, Y., and Bengio, Y., 1998, "Convolutional Networks for Images, Speech, and Time Series," *The Handbook of Brain Theory and Neural Networks*, M. A. Arbib, ed., MIT Press, Cambridge, MA, pp. 255–258.

[47] Krizhevsky, A., Sutskever, I., and Hinton, G. E., 2017, "Imagenet Classification with Deep Convolutional Neural Networks," Commun. ACM, **60**(6), pp. 84–90.

[48] Johnson, J., Alahi, A., and Li, F., 2016, "Perceptual Losses for Real-Time Style Transfer and Super-Resolution," CoRR, abs/1603.08155.

[49] Hu, J., Shen, L., and Sun, G., 2017, "Squeeze-and-Excitation Networks," CoRR, abs/1709.01507.