Proceedings of the ASME 2020 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference
IDETC/CIE 2020
August 16-19, 2020, St. Louis, MO, USA

DETC2020-19461

STRESSGAN: A GENERATIVE DEEP LEARNING MODEL FOR 2D STRESS DISTRIBUTION PREDICTION

Haoliang Jiang Zhenguo Nie Roselyn Yeo Amir Barati Farimani Levent Burak Kara *
Department of Mechanical Engineering, Carnegie Mellon University, Pittsburgh, PA 15213, USA

ABSTRACT

Using deep learning to analyze mechanical stress distributions has been gaining interest with the demand for fast stress analysis methods. Deep learning approaches have achieved excellent outcomes when utilized to speed up stress computation and learn the physics without prior knowledge of underlying equations. However, most studies restrict the variation of geometry or boundary conditions, making these methods difficult to be generalized to unseen configurations. We propose a conditional generative adversarial network (cGAN) model for predicting 2D von Mises stress distributions in solid structures. The cGAN learns to generate stress distributions conditioned by geometries, load, and boundary conditions through a two-player minimax game between two neural networks with no prior knowledge. By evaluating the generative network on two stress distribution datasets under multiple metrics, we demonstrate that our model can predict more accurate high-resolution stress distributions than a baseline convolutional neural network model, given various and complex cases of geometry, load and boundary conditions.

1 Introduction

Structural stress analysis is a critically important foundational tool in many disciplines including mechanical engineering, material science, and civil engineering. It is used for predicting the stress distribution and the possibility of structural failure when the structure is subject to the applied load and boundary conditions [1–3]. Finite Element Analysis (FEA) is commonly used to discretize the domain and to solve the governing partial differential equations [4–7]. Traditional methods provide high fidelity solutions but require the solution of large linear systems which can be computationally prohibitive. With the demand for fast and accurate structural analysis in generative design, topology optimization technologies and online manufacturing monitoring, increasing the computational speed for stress analysis has become a focus of interest.

To achieve fast mechanics analysis, many prior works have focused on deep learning techniques to help compute computational engineering problems [8, 9]. Several approaches of accelerating mechanical stress analysis by deep learning methods have been carried out and achieved excellent outcomes in terms of computational speed and accuracy [6, 10–14]. These studies utilize deep learning models to predict residual stress, shear stress, maximum von Mises stress or distributions of the stress tensor. Once trained on large datasets, these approaches are able to generate accurate stress predictions. However, most previous work restricts the geometry or the boundary conditions that are applied, making the models difficult to be generalized to new problems.

In this work, we propose a conditional generative adversarial network we call StressGAN for stress distribution prediction. StressGAN takes as input arbitrary geometries, load and boundary conditions in the form of different input channels and predicts the von Mises stress distribution in an end-to-end fashion. A distinguishing feature of our approach is that we utilize a generative adversarial network instead of an autoencoder as our learning algorithm.

We evaluate StressGAN on two datasets: a fine-mesh multiple-structure dataset introduced by this work and a coarse-mesh cantilever beam dataset used in [6]. The fine-mesh dataset

*Address all correspondences to lkara@cmu.edu

contains 38,400 problem samples modeled as 128×128 meshes. Unlike the coarse-mesh dataset with identical shape, boundary and load conditions, the fine-mesh dataset includes ten patterns of load positions and eight patterns of boundary conditions. To explore the performance of StressGAN under different scenarios, we design two types of experiments : training and evaluating the network on entire dataset and training and evaluating the network on datasets with conditions of different categories (generalization experiments). As a result, StressGAN outperforms a selected baseline model, StressNet (SRN), proposed in [6], on the fine-mesh dataset and generates reasonably accurate results on the coarse-mesh dataset. Furthermore, StressGAN generates relatively accurate stress distributions for most test cases in the generalization experiments with sparse training dataset.

2 Background

We focus our review on finite element analysis, then on studies that focus on deep learning methods with emphasis on their applications in stress estimation and generative adversarial networks with emphasis on their applications in computational engineering.

2.1 Finite element analysis for stress computation

Typical linear finite element analysis (FEA) for stress calculations involve:

$$KQ = F \tag{1}$$

Where K denotes a global stiffness matrix; F denotes a vector describing the applied load at each node; Q denotes the displacement. K is composed of elemental stiffness matrices k_e for each element:

$$k_e = A_e B^T D B \tag{2}$$

where B is the strain/displacement matrix; D is the stress/strain matrix; A_e is the area of the element. B and D are determined by material properties and mesh geometry. Then the local stiffness matrix k_e will be added into the global stiffness matrix. The displacement boundary conditions are encoded into the global stiffness matrix K by operating on the corresponding rows and columns. Various direct factorization based or iterative solvers exist for the solution of Q.

After computing the global displacement using equation (1), the nodal displacement q of each element, followed by the stress tensor of each element:

$$\sigma = DBq \tag{3}$$

Where σ denotes the tensor of an element. Then, the von Mises Stress of each element is computed using the 2-D von Mises Stress form:

$$\sigma_{vm} = \sqrt{\sigma_x^2 + \sigma_y^2 - \sigma_x \sigma_y + 3\tau_{xy}^2} \tag{4}$$

where σ_{vm} is von Mises Stress; σ_x, σ_y, τ_{xy} are the normal and shear stress components.

2.2 Deep learning in mechanical stress analysis

Most of the early attempts to use deep learning in speeding up mechanical stress analysis focus on integrating the models in a FEA software. These models are to solve some auxiliary tasks including updating FEA model [15, 16], checking plausibility of a FE simulation [17], modeling the constitutive relation of a material [18] and optimizing the numerical quadrature in the element stiffness matrix on a per element basis [19] . These works alleviate the complexity of FEA software to some extent. However, our approach could be used as a surrogate model to a FEA software. It avoids the computation bottlenecks in a FEA software and its computation cost could be controlled by modifying the architecture.

Deep learning methods are proposed as surrogate models to approximate residual stress in girth welded pipes [13], shear stress in circular channels [14] or stress in 3D trusses [11]. These methods use manually assigned features to represent a fixed geometry or a part of the geometry. The deep learning models will estimate a stress value based on the input parameters. In our work, the deep learning method learns to filter useful features and generates a representation for each combination of the geometry, external load and boundary condition. A decoder follows the data representation and predicts a stress distribution on the geometry.

Liang et al. [10] have developed an image-to-image deep learning framework as an alternative to predicting aortic wall stress distributions by expanding aortic walls into a topologically equivalent rectangle. Khadilkar et al. [12] propose a two-stream deep learning framework to predict stress fields in each of the 3D printing process. The network encodes 2D shapes of each layer and the point clouds of 3D models based on a CNN architecture and a PointNet [20]. Madani et al. [21] propose a transfer learning model to predict the value and position of the maximum von Mises stress on arterial walls in atherosclerosis. Our model also use an image-to-image translation model to estimate the stress distribution. However, we utilize image-based data representation on both the geometry and the input conditions. Thus, our model can be used to analyze arbitrary 2D stress distribution cases after proper training.

Most related to our work, Nie et al. [6] propose an end-to-end convolutional neural network called StressNet to predict

2D stress distributions given multi-channel data representations of geometry, load and boundary conditions of cantilever beams. The network contains three downsampling convolutional layers, five Squeeze-and-Excitation ResNet (SE-RES) blocks [22, 23] and three upsampling convolutional layers. Each SE-RES block is composed of two convolutional layers and a SE block which utilizes a global pooling and two fully connected layers to learn extra weights for each channel. Skip connections are used in each block. 9x9 kernels are used in the first and last convolutional layer and 3x3 kernels are used in all remaining convolutional layers. A dataset composed of 120,960 cases of cantilever beams modeled using 32×32 meshes is generated by a linear FEA software to train and evaluate the network. In our work, we aim at analyzing high-resolution cases and use an adversarial learning scenario additionally to capture features in stress distributions. More importantly, all previous work of deep learning methods in stress prediction focus on specific application cases lacking variety in geometry, external load and boundary conditions. Moreover, through testing our model by geometries or conditions from unseen domain, we show the potential of our deep learning model as a transfer learning model for stress field predictions.

2.3 Generative adversarial networks

GANs are an example of generative models. They model the training of a generative network as a two-player minimax game where a generator G is trained to learn a distribution f with a discriminator D [24]. Both of them represent a differentiable function contolled by the learned parameters. In a conventional GAN, the generator G learns to map a vector sampled from a latent space $z \sim p_z(z)$ to the space of ground truth samples. In the meantime, the discriminator D learns to map a sample to a probability that predicts if the presented sample is real or fake. The Nash equilibrium in training is that the generator forms the same distribution as the training data and the discriminator output 0.5 for all input data [25].

cGAN is built upon the learning algorithm of GAN and has been widely used to date [26–29]. cGAN develops a method to control the mapping from input to output by conditioning the standard generator G and discriminator D on extra information. Figure 1 demonstrates the framework of cGAN. Further, Isola et al. [26] propose a similar network for image-based task such as image-to-image translation. In the comparisons against networks plainly using MAE as a loss function, it shows the superiority of using cGAN framework in image-based tasks. Radford et al. [30] reinforce GAN's training stability by using all convolutional net [31], ReLU [32] and LeakyReLU [33] activations and batch-normalization layers [34].

Farimani et al. [35, 36] propose a cGAN architecture based on the network proposed by Isola et al. [26] to learn models of steady state heat conduction, incompressible fluid flow, and

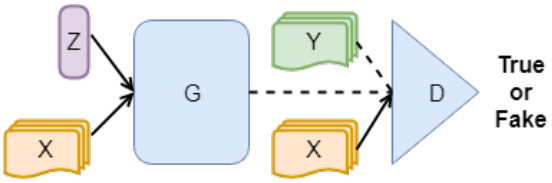

FIGURE 1. Framework of cGAN. The generator G and discriminator D are conditioned on information X. A latent vector Z and X compose the input to G. D learns to tell whether its input regarding X is from real samples Y or output of G.

phase segregation. S. Lee et al. use GAN in the prediction of unsteady flow over a circular cylinder with various Reynolds numbers [37]. Paganini et al. [38] use a revised DCGAN is developed to model electromagnetic showers in a longitudinally segmented calorimeter. The deep learning method speeds up the calculations by more than 100 times. K. Enomoto et al. also utilize a DCGAN architecture for cloud removal in climate images [39]. In the field of astronomy, GANs are used to generate images of galaxies [40,41] and 2D mass distributions [42]. In our work, we use the architecture and learning algorithm introduced by Radford et al. [30] and Isola et al. [26] to build our neural network for stress field predictions cross varying input geometries and boundary conditions.

3 Technical Approach

3.1 Neural Network Architecture

The architecture of StressGAN is shown in figure 2. We design the generator as an encoder-decoder network which generates a feature vector with a size of 512 in the bottleneck. The input of the generator is a case of conditions and geometry modeled by $m \times m$ meshes. Three $m \times m$ resolution images are used to represent geometry, boundary conditions and the applied load. To increase data intensity, we represent geometry and boundary conditions on one image. We use numbers 0, 1, 2, 3, 4 in geometries to represent various boundary conditions, where 0 is void, 1 means free solid node, 2 means node affixed horizontally, 3 means node affixed vertically, 4 means node affixed on both directions. The remaining two images record magnitudes of vertical or horizontal loads in the corresponding pixel. The output of the generator is a $m \times m$ mesh describing the von Mises stress distributions. The encoder is comprised of $\log_2(m)$ downsampling blocks with a convolutional layer, a batch normalization layer and a LeakyReLU layer. Similarly, the decoder is comprised of $\log_2(m)$ upsampling blocks with a deconvolutional layer, a batch normalization layer and a ReLU layer. When the network is trained and tested using the coarse-mesh dataset, we remove four blocks close to the bottleneck to keep the bottleneck representation of input conditions as a 512 feature vector. We remove the ReLU layer in the last upsampling block with the considera-

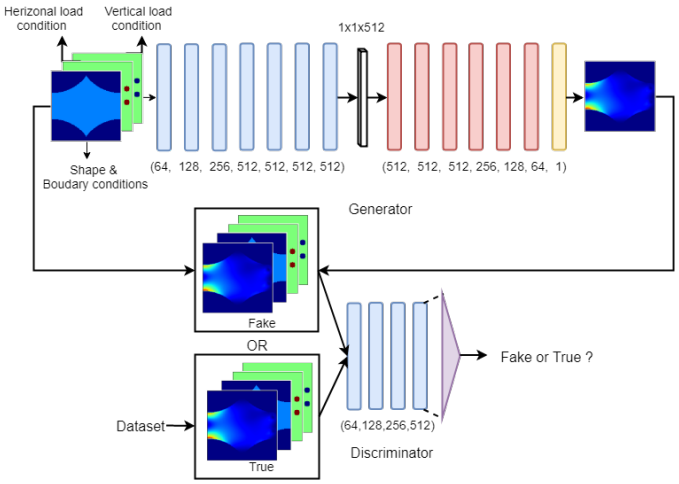

FIGURE 2. Architecture of StressGAN. The generator (top) and the discriminator (bottom) are constructed with downsampling blocks (blue) and upsampling blocks (red). For the last upsampling block of the generator (yellow), we remove the ReLU activation. The numbers indicate channel dimensions of the output of each blocks. The purple triangle means a reshape layer followed by a linear layer and a Sigmoid activation.

tion that mechanics analysis results other than von Mises Stress might contain negative values. The convolutional layers and deconvolutional layers both have kernel sizes as 5×5 and stride size as 2.

For the discriminator, we adopt a downsampling structure. The input is a stress distribution case described by four $m \times m$ images including the fake or ground truth sample stress distribution and its conditions. The architecture of the discriminator is fixed when experimented on different datasets. It outputs a probability value which describes whether the input analysis result is true regarding the conditions and geometry. Four downsampling blocks are followed by a reshape layer, a fully connected layer, and a Sigmoid activation.

3.2 Loss function and metrics

Loss function Our loss function consists of an L2 distance loss and a cGAN objective function:

$$\mathrm{L_{L2}}(G) = \mathrm{E_{x,y}}[||y - \mathrm{G}(x)||_2] \tag{5}$$

where y is ground truth stress distributions; x stands for conditions and geometries, G denotes the generator. Previous work has shown that utilizing L2 distance (MSE) to train networks for predicting stress distributions works well. Thus, we use L2 distance as a loss in StressGAN's loss function.

The loss function of cGAN used in our model can be expressed as:

$$\begin{aligned} \mathrm{min_G max_D V}(G,D) = \mathrm{E_{x,y}}[\log(\mathrm{D}(x,y))] + \\ \mathrm{E_x}[\log(1 - \mathrm{D}(x, \mathrm{G}(x)))] \end{aligned} \tag{6}$$

cGAN loss function shows the adversarial relationship between the generator G and the discriminator D. Note that in our cases where the network should output a particular analysis result given the conditions and a geometry, we eliminate the Gaussian noise vector z which is usually an input of the generator to add more variation to the output.

The final loss is:

$$\mathrm{min_G max_D V}(G,D) + \lambda \mathrm{L_{L2}}(G) \tag{7}$$

where a hyperparameter λ is to balance the loss function following [26, 43].

Metrics In addition to MSE, four metrics are introduced to assess the performance of StressGAN: mean absolute error (MAE), percentage mean absolute error (PMAE), peak stress absolute error (PAE) and percentage peak stress absolute error (PPAE). These four metrics, whether related to MSE or not, are not an explicit goal of minimizing MSE. Using these four metrics, we can provide an assessment of stress prediction qualities.

Using MAE and a normalized version of MAE, PMAE, helps evaluate the overall quality of a predicted stress distribution. MAE is defined as:

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

where y_j is the value at element j in a ground truth sample; $\hat{y_j}$ is the estimated value at element j; n denotes the number of elements of samples. PMAE is defined as:

$$\mathrm{PMAE} = \frac{\mathrm{MAE}}{\max\{Y\} - \min\{Y\}} \times 100\% \tag{9}$$

where Y denotes a set of all ground truth stress values in a case; $\max\{Y\}$ is the maximum value in a set of ground truth stress values Y; $\min\{Y\}$ is the minimum value in a set of ground truth stress values Y.

PAE and PPAE measure the accuracy of the most considerable stress value in a predicted stress distribution which is the most critical local value of stress distributions in engineering applications. PAE is defined as:

$$\text{PAE} = |\max\{Y\} - \max\{\hat{Y}\}| \tag{10}$$

where $\hat{Y}$ is the set of all predicted stress values in a case; $\max\{\hat{Y}\}$ is the maximum value in a set of all predicted stress values $\hat{Y}$.

PPAE is defined as:

$$\text{PPAE} = \frac{\text{PAE}}{\max\{Y\}} \times 100\% \tag{11}$$

4 Experiments

4.1 Dataset and implementation

Fine-mesh multiple structure dataset To provide a broad evaluation of our network's performance, we introduce a stress distribution dataset composed of multiple structures each modeled as a 128×128 elements. The dataset is generated using A 2D FEM software SolidsPy [44]. All elements in the domain is a 4-node quadrilateral with a size of 1×1 (mm). Void regions are modeled using a Young's modulus of infinitesimal value. The dataset contains 60 geometries, ten patterns of load conditions and eight patterns of boundary conditions, in total, 38,400 instances. The shapes, load conditions and boundary conditions are not limited to cantilever beams which are affixed on one end and bearing loads on the other end. Samples of geometry, load and boundary conditions are demonstrated in Figure 3. Also, for each load pattern, the orientations of the loads can vary from 0 degrees to 315 degrees with a step of 45 degrees. We normalize the load magnitudes in the dataset to reduce the input space since the linear characteristic of homogeneous and isotropic elastic material results in a linear relationship between the loads and the stresses.

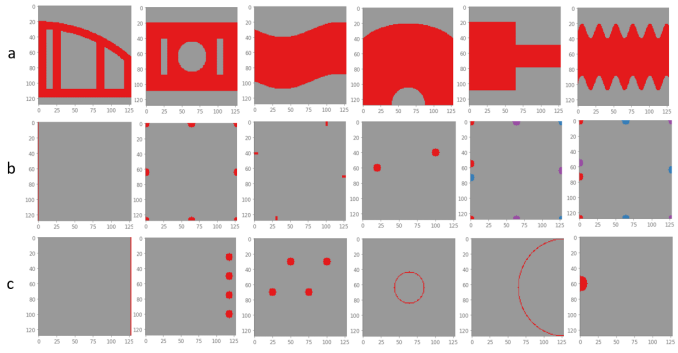

FIGURE 3. Data samples in fine-mesh dataset. a. Geometries. b. Boundary conditions. c. Load positions.

Coarse-mesh cantilever beam dataset The course-mesh stress distribution dataset is proposed by Nie et al. [6]. The dataset consists of six categories of geometries, in total, 80 geometries. Examples of categories of geometry are shown in Figure 4. Load is applied on the right end of the beam. The left end of the beam is fixed. For each geometry, load orientation changes from 0 degrees to 355 degrees, in 5 degree increments. For each orientation, the load magnitudes varies from 0N to 100N by a step of 5N. In total, the dataset includes 120,960 instances with various shapes, load orientations, and load magnitudes.

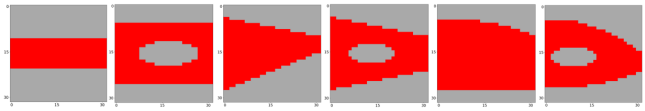

FIGURE 4. Categories of geometry in coarse-mesh dataset. From left to right: rectangular beam; rectangular beam with a cellular opening; trapezoidal beam; trapezoidal beam with a cellular opening; beam with parabola contours; beam with parabola contours and a cellular opening

Implementation detail We train StressGAN using a learning rate of 0.001 by the Adam optimizer [45] with a batch size of 64. We use 1 or 10 as the value of λ in StressGAN's loss in the experiments with the fine-mesh dataset and coarse-mesh dataset respectively. The learning rate and batch size are decided by a grid search on potential values. The performance of the model and fitting GPU memory size are main considerations in the grid search. The selection of λ is up to the balance between L2 loss and cGAN loss. We aim to keep the losses at the same weight for stabilizing the training process. In each training epoch, we train the discriminator once and the generator twice to keep the training stable. In all experiments, we use an NVIDIA GeForce GTX 1080Ti GPU. Under our experiment setting, each cases in fine-mesh dataset take approximately 0.003 seconds to analyze.

4.2 Experiment Design

Entire dataset training and evaluation In this experiment, we randomly divide both the fine-mesh dataset and the coarse-mesh dataset into train/test sets of $80\% - 20\%$ split respectively. We then train and evaluate StressGAN on the datasets to demonstrate its effectiveness. Additionally, we train and evaluate our baseline model SRN under the same scenario to compare their performances.

Generalization training and evaluation To further study StressGAN's performances in general engineering scenarios such as unseen geometries or unseen applied loads, we set three sub-experiments where training and test sets belonging to different categories of geometry or load orientation. The whole experiment is set based on coarse-mesh dataset since it is easier to separate geometries into semantic categories. In the first and

second sub-experiments, we train and evaluate the networks using samples from different geometry categories respectively. In the first sub-experiment, we train the networks with samples in the categories of rectangular beams, trapezoidal beams, rectangular beams with cellular openings and trapezoidal beams with cellular openings and evaluate the networks with beams with a parabola contour. In the second experiment, we train the networks using beams without holes and evaluate the networks using beams with cellular openings. The third sub-experiment is to study how the network performs when trained and evaluated by cases with different load orientations. The load orientations are split up by quadrants. We randomly select loads in three quadrants for training and use loads in the remaining quadrant for testing. We normalize the load magnitudes in all training and test datasets, which reduces the size of all training datasets to less than 5000 samples.

5 Results and Discussions

5.1 Entire dataset evaluation

As stated previously, we train and test our model using the fine-mesh dataset with a split of 80% - 20%. Meanwhile, we train and test SRN on the same training and testing dataset. The evaluation results of the three networks are shown in Table 1. The best performance under each metric are shown in bold. StressGAN attains a PMAE of 0.21% and a PPAE of 1.47%, which indicates StressGAN can produce accurate fine-mesh stress distribution given complex cases. The statistical accuracy of StressGAN on the test dataset is shown in Figure 5. The most inaccurate predictions are shown in Figure 6. Even with the highest related error rates, these predictions still provide useful information. Table 1 shows that StressGAN outperforms SRN with a significant margin in all metrics. Figure 7 shows comparisons of the evaluation results of StressGAN and SRN. As shown in the visualizations, the predicted stress distributions of StressGAN are sharper than the predictions of SRN, especially around the edges of the void versus material boundaries. Additionally, StressGAN's predictions of the critical areas are comparatively more accurate.

We also visualize the activation layers of a random sample and test the discrimination of the discriminator. Figure 8 shows the output activation layers of the convolutional layers in the generator. From the figures in the upper row, it can be clearly observed that the positions of boundary conditions and external forces are highlighted by the filters, which demonstrates the ability of the network to capture and transfer input conditions. The figures in the bottom row provides insights into how the network computes the stress distribution based on the encoded information. Ground truths and predictions of test dataset are fed into the discriminator. The average output of the discriminator given the ground truths and predictions is 0.899 and 0.002, respectively. This shows that the discriminator has learned the implicit features of stress distributions. Even with test data, it is able to distinguish the ground truth distribution from the predicted ones.

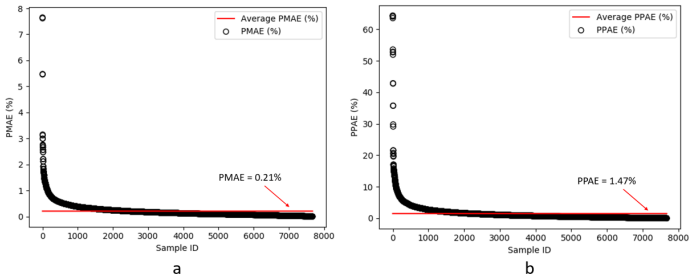

FIGURE 5. Statistical accuracy of StressGAN on fine mesh dataset. a. PMAE of each sample and average PMAE on the test dataset. b. PPAE of each sample and average PPAE on the test dataset.

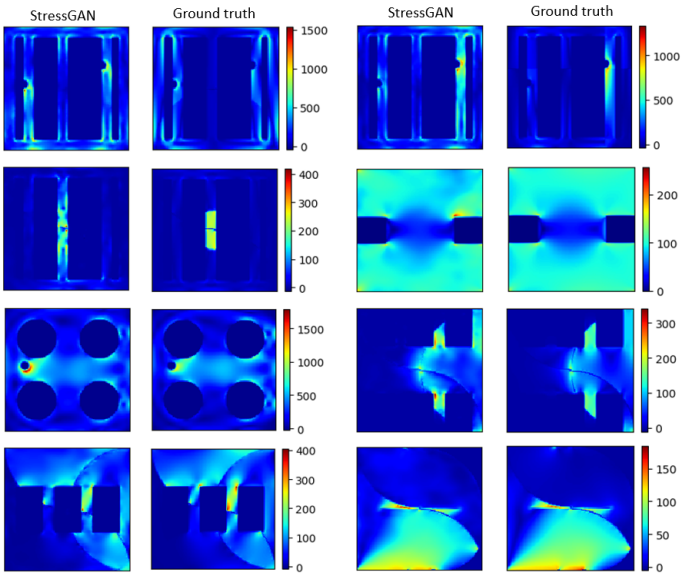

FIGURE 6. The worst predictions of StressGAN on fine-mesh dataset.

TABLE 1. Quantitative evaluation of StressGAN and SRN with find-mesh dataset. The best result under each metric is shown in bold. (Units: mm-MPa-N)

Metric	MSE	MAE	PMAE	PAE	PPAE
StressGAN	**77.31**	**1.83**	**0.21%**	**20.29**	**1.47%**
SRN	1119.75	10.88	1.20%	132.64	19.80%

We also train and test our model using the coarse-mesh dataset with a split of 80% - 20% of training and test dataset. The evaluation results of StressGAN and SRN are shown in Ta-

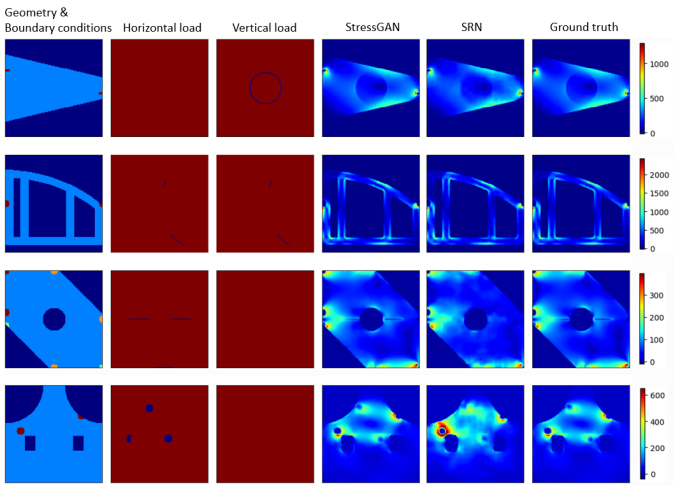

FIGURE 7. Evaluation of StressGAN and SRN on fine-mesh dataset. Four evaluation cases are shown by each row. From left to right: 1) Geometry (light blue) and boundary conditions (cyan: horizontal, orange: vertical, red: vertical and horizontal); 2) horizontal load positions; 3) vertical load positions; 4) predictions of StressGAN; 5) predictions of SRN; 6) ground truths. The load magnitudes of each case: 1) $q(x) = 0.0N/mm^2$, $q(y) = -88.4N/mm^2$; 2) $q(x) = 125.0N/mm^2$, $q(y) = 125.0N/mm^2$; 3) $q(x) = 100.0N/mm^2$, $q(y) = 0.0N/mm^2$; 4) $q(x) = -68.8N/mm^2$, $q(y) = 0.0N/mm^2$. (Units: mm-MPa-N)

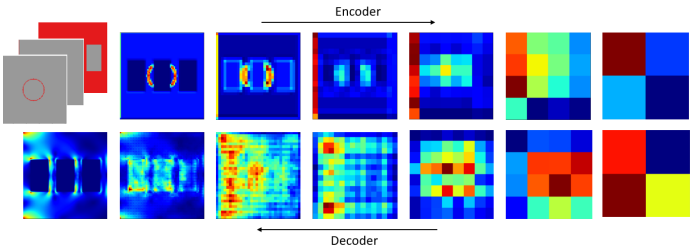

FIGURE 8. Output activation layers of the generator. The output activation layers of convolutional layers are shown to display the encoding and decoding processes.

TABLE 2. Quantitative evaluation of StressGAN and SRN with coarse-mesh dataset. The best performance under each metric is shown in bold. (Units: mm-MPa-N)

Metric	MSE	MAE	PMAE	PAE	PPAE
StressGAN	1.08	0.60	0.59%	2.17	2.11%
SRN	**0.15**	**0.20**	**0.15%**	**0.50**	**0.37%**

ble 2. Developed initially for this dataset, SRN performs well in this experiment and attains a better evaluation performance than StressGAN. With four layers removed, StressGAN still achieves

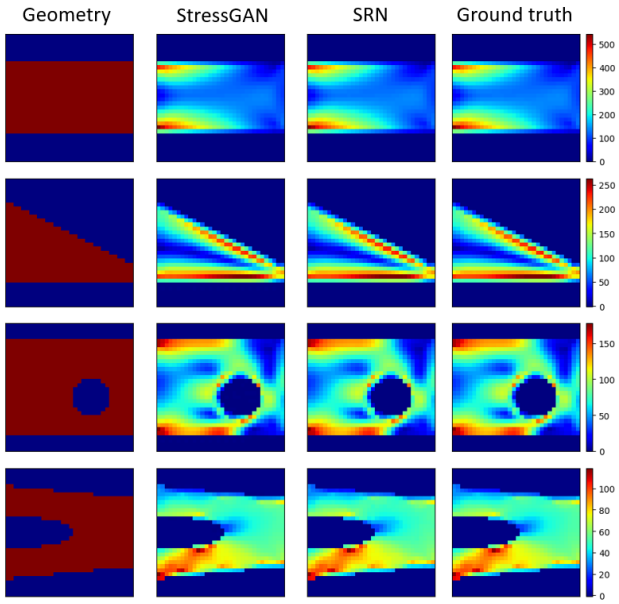

FIGURE 9. Evaluation of StressGAN and SRN on coarse-mesh dataset. Four evaluation cases are shown by each row. The visualizations of results of StressGAN and SRN are identical to the ground truth stress distributions. From left to right: 1) geometry (red); 2) predictions of StressGAN; 3) predictions of SRN; 4) ground truth stress distributions. The load magnitudes of each case: 1) $q(x) = 27.5N/mm^2$, $q(y) = -47.6N/mm^2$; 2) $q(x) = -43.0N/mm^2$, $q(y) = 61.4N/mm^2$; 3) $q(x) = -3.5N/mm^2$, $q(y) = 19.7N/mm^2$; 4) $q(x) = -54.8N/mm^2$, $q(y) = -4.8N/mm^2$. (Units: mm-MPa-N)

a reasonably low error rate with a PMAE of 0.5%. This error rate is low enough that it is difficult to tell the differences between the predicted results of the two networks through visualizations in Figure 9.

5.2 Generalization evalution

We conduct generalization experiments to explore our method's performance in situations where the training dataset is sparse and testing data contains unseen cases. We include SRN and StressGAN into this experiment to compare their performances and demonstrate the characteristics of each network. The parametric results of the three experiments are shown in Table 3, 4 and 5. The best performance under each metric is shown in bold. The selected samples of prediction results are shown in figure 10, 11 and 12 respectively. In general, StressGAN gives a better performance concerning the average prediction accuracy. The best PMAEs in three experiments are 8.23%, 6.80%, and 1.49% respectively, which are all obtained by StressGAN.

In the two cross-geometry experiments, we can study the characteristics of StressGAN and SRN including their advantages and disadvantages. Figure 10 shows the visualizations of

TABLE 3. Quantitative evaluation of StressGAN and SRN with training data of rectangular beams and trapezoidal beams and testing data of beams with a parabolar contour. The best performance under each metric is shown in bold. (Units: mm-MPa-N)

Metric	MSE	MAE	PMAE	PAE	PPAE
StressGAN	**28.91**	**2.80**	**7.50%**	**6.85**	**18.10%**
SRN	43.14	3.28	9.30%	12.55	38.39%

TABLE 4. Quantitative evaluation of StressGAN and SRN in the second sub-experiment with training data of beams without openings and testing data of beams with cellular openings. The best performance under each metric is shown in bold. (Units: mm-MPa-N)

Metric	MSE	MAE	PMAE	PAE	PPAE
StressGAN	**77.20**	**4.40**	**6.80%**	16.96	24.10%
SRN	95.36	4.59	7.54%	**14.09**	**23.62%**

TABLE 5. Quantitative evaluation of StressGAN and SRN in the experiment of cross-load direction training and evaluation. The best performance under each metric is shown in bold. (Units: mm-MPa-N)

Metric	MSE	MAE	PMAE	PAE	PPAE
StressGAN	**3.71**	**0.84**	**1.49%**	**3.15**	**4.86%**
SRN	6.86	1.29	2.58%	6.72	11.66%

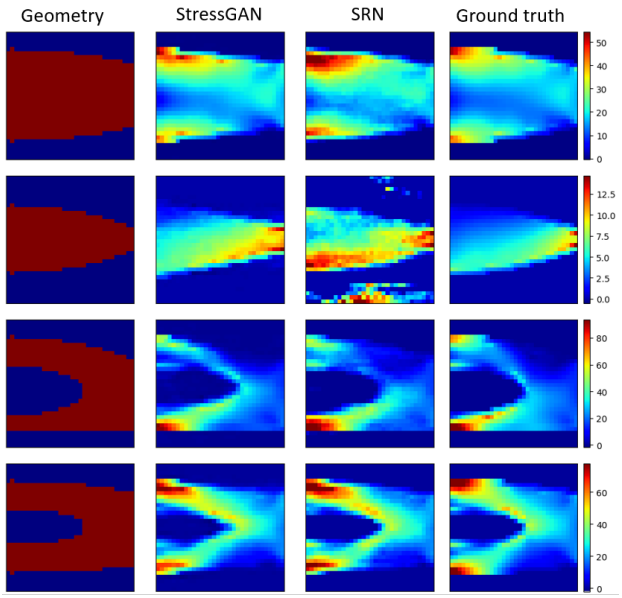

FIGURE 10. Evaluation of StressGAN and SRN on cases of different contours. Four evaluation cases are shown by each row. From left to right: 1) geometry (red); 2) predictions of StressGAN; 3) predictions of SRN; 4) ground truth stress distributions. The load magnitudes of each case: 1) $q(x) = -5.7N/mm^2$, $q(y) = -8.2N/mm^2$; 2) $q(x) = 10.0N/mm^2$, $q(y) = -0.9N/mm^2$; 3) $q(x) = -7.7N/mm^2$, $q(y) = 6.4N/mm^2$; 4) $q(x) = 5.0N/mm^2$, $q(y) = 8.7N/mm^2$. (Units: mm-MPa-N)

the ground truth stress distributions and prediction stress distributions in the first cross-geometry experiment. Although the contour information of the input geometries is hard for StressGAN to capture, StressGAN outputs stress distributions closer to the samples in the dataset, especially in the regions of high stresses. Additionally, it generates a sharper (less blurred) prediction. Figure 11 shows similar trends for the second cross-geometry experiment. On the one hand, StressGAN failed to predict stresses around the openings correctly. On the other hand, StressGAN generates more reasonable stress distributions which are more similar to the ground truth samples. Additionally, SRN could recognize the openings and predict zero stresses in void areas in some test cases. Since cellular openings have a considerable influence on stress concentrations and the networks have no explicit training on this phenomenon, large errors occur when we evaluate the predicted largest stress values as shown in Table 4.

The results of the cross-orientation experiment are shown in Figure 12. The output stress distributions from StressGAN are quite similar to the ground truths. From Table 5, it can be seen that among the three generalization experiments, the cross-orientation experiment attains the best evaluation results. Since we use two images to express the load positions and magnitudes along the horizontal and vertical directions respectively, the deep learning method has a potential to learn the influence of the horizontal and vertical loads from the training dataset separately (essentially, the principle of superposition by exploting the linear nature of FEA) and synthesize reasonable results when tested on unseen load orientations. This is especially useful in compressing the size of the training dataset for data efficiency without significantly increasing the error rate.

6 Conclusions

In this work, we develop a conditional generative adversarial network we call StressGAN for von Mises stress distribution prediction. StressGAN learns to predict the stress distribution given the geometries, load, and boundary conditions through a 2-player minimax game between its generator and discriminator. A fine-mesh stress distribution dataset composed of 38,400 cases of various geometries, load, and boundary conditions is proposed for evaluating the network's performance of complex stress prediction cases.

StressGAN achieves high accuracy in both experiments under multiple metrics, in evaluations of two stress distribution datasets. StressGAN outperforms the baseline model in both

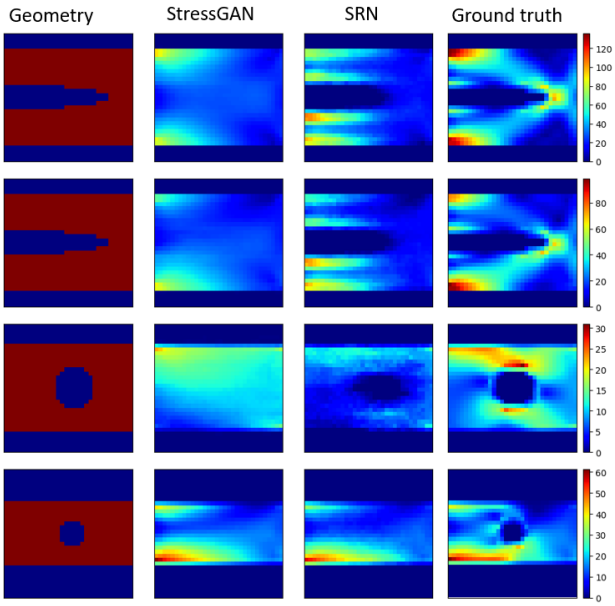

FIGURE 11. Evaluation of StressGAN and SRN on cases of cantilever beams with cellular openings. Models are trained with cantilever beams with openings and tested with cantilever beams without openings. Four evaluation cases are shown by each row. From left to right: 1) geometry (red); 2) predictions of StressGAN; 3) predictions of SRN; 4) ground truth stress distributions. The load magnitudes of each case: 1) $q(x) = 1.7N/mm^2$, $q(y) = 9.8N/mm^2$; 2) $q(x) = -7.7N/mm^2$, $q(y) = 6.4N/mm^2$; 3) $q(x) = 10.0N/mm^2$, $q(y) = 0.9N/mm^2$; 4) $q(x) = -9.1N/mm^2$, $q(y) = 4.2N/mm^2$. (Units: mm-MPa-N)

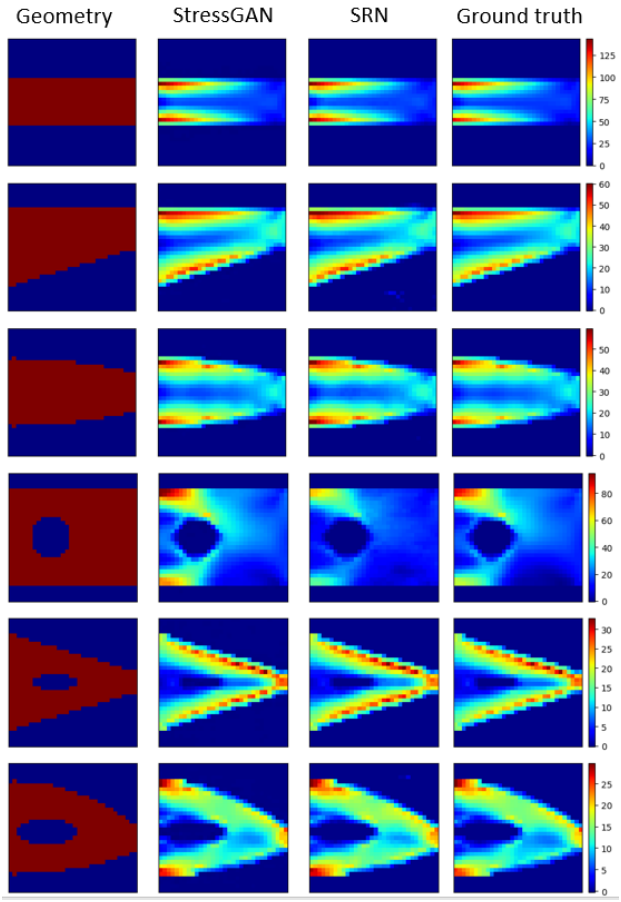

FIGURE 12. Evaluation of StressGAN and SRN on cases of different load orientations. This figure shows six evaluation cases of StressGAN and SRN when trained and tested with load conditions in different quadrants. From left to right: 1) geometry; 2) predictions of StressGAN; 3) predictions of SRN; 4) ground truth stress distributions. The load magnitudes of each case: 1) $q(x) = -4.2N/mm^2$, $q(y) = -9.1N/mm^2$; 2) $q(x) = -4.2N/mm^2$, $q(y) = -9.1N/mm^2$; 3) $q(x) = -3.4N/mm^2$, $q(y) = -9.4N/mm^2$; 4) $q(x) = -7.7N/mm^2$, $q(y) = -6.4N/mm^2$. 5) $q(x) = -3.4N/mm^2$, $q(y) = -9.4N/mm^2$. 6) $q(x) = -4.2N/mm^2$, $q(y) = -9.1N/mm^2$. (Units: mm-MPa-N)

qualitative and quantitative evaluations in predicting the stress distributions given complicated geometry, displacement, and load boundary conditions. It achieves an average error rate less than 0.21% on all stress values and 1.47% on the maximum stress value.

Moreover, StressGAN's performance under general scenarios is studied. StressGAN generates stress distributions more similar to samples in the dataset which shows it is a more effective learner in capturing the underlying knowledge of ground truth stress distributions. Furthermore, StressGAN is more efficient when facing unseen conditions. Although some cases that lead to stress concentration such as holes in geometries result in inaccurate predictions from StressGAN, the computed stress distributions still embody useful information such as the location of the highly stressed regions. The stress distributions are more similar to ground truths compared to the baseline method regardless of the conditions. In contrast, our baseline model SRN is better at correctly estimating zero stresses in void areas but produces overall less accurate stress distributions under the same problem inputs. Furthermore, both StressGAN and SRN perform well given unseen load orientations compared to the cases where unseen geometries are involved.

In this work, the potential of generalizing the stress prediction ability to different categories is shown in generalization experiments. These findings constitute a one step toward generating data-driven analysis approaches that can generalize well to previously unseen problem configurations.

REFERENCES

[1] Yang, R., and Chen, C., 1996. "Stress-based topology optimization". *Structural optimization,* ***12***(2-3), pp. 98–105.

[2] Wang, D., Lee, J., Holland, K., Bibby, T., Beaudoin, S., and Cale, T., 1997. "Von mises stress in chemical-mechanical polishing processes". *Journal of the Electrochemical Society,* ***144***(3), pp. 1121–1127.

[3] Chen, J., Akyuz, U., Xu, L., and Pidaparti, R., 1998. "Stress analysis of the human temporomandibular joint". *Medical Engineering & Physics,* ***20***(8), pp. 565–572.

[4] Segerlind, L. J., 1976. *Applied finite element analysis*, Vol. 316. Wiley New York.

[5] Cook, R. D., et al., 2007. *Concepts and applications of finite element analysis*. John Wiley & Sons.

[6] Nie, Z., Jiang, H., and Kara, L. B., 2019. "Stress Field Prediction in Cantilevered Structures Using Convolutional Neural Networks". *Journal of Computing and Information Science in Engineering,* ***20***(1), 09. 011002.

[7] Mises, R. v., 1913. "Mechanik der festen körper im plastisch- deformablen zustand". *Nachrichten von der Gesellschaft der Wissenschaften zu Göttingen, Mathematisch-Physikalische Klasse,* ***1913***, pp. 582–592.

[8] Adie, J., Yang, J., Zhang, M., and See, S., 2018. "Deep learning for computational science and engineering". In GPU Technology Conference, no. S8242.

[9] LeCun, Y., Bengio, Y., and Hinton, G., 2015. "Deep learning". *nature,* ***521***(7553), p. 436.

[10] 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". *Journal of The Royal Society Interface,* ***15***, 01.

[11] Nourbakhsh, M., Irizarry, J., and Haymaker, J., 2018. "Generalizable surrogate model features to approximate stress in 3d trusses". *Engineering Applications of Artificial Intelligence,* ***71***, pp. 15–27.

[12] Khadilkar, A., Wang, J., and Rai, R., 2019. "Deep learning–based stress prediction for bottom-up sla 3d printing process". *The International Journal of Advanced Manufacturing Technology*, pp. 1–15.

[13] Mathew, J., Moat, R., Paddea, S., Fitzpatrick, M. E., and Bouchard, P., 2017. "Prediction of residual stresses in girth welded pipes using an artificial neural network approach". *International Journal of Pressure Vessels and Piping,* ***150***, pp. 89–95.

[14] Khozani, Z. S., Bonakdari, H., and Zaji, A. H., 2017. "Estimating the shear stress distribution in circular channels based on the randomized neural network technique". *Applied Soft Computing,* ***58***, pp. 441–448.

[15] Levin, R., and Lieven, N., 1998. "Dynamic finite element model updating using neural networks". *Journal of Sound and Vibration,* ***210***(5), pp. 593 – 607.

[16] Atalla, M., and Inman, D., 1998. "On model updating using neural networks". *Mechanical Systems and Signal Processing,* ***12***(1), pp. 135 – 161.

[17] Spruegel, T., Schröppel, T., and Wartzack, S., 2017. "Generic approach to plausibility checks for structural mechanics with deep learning".

[18] Javadi, A., and P. Tan, T., 2003. "Neural network for constitutive modelling in flnite element analysis". *Computer Assisted Mechanics and Engineering Sciences,* ***10***, 01.

[19] Oishi, A., and Yagawa, G., 2017. "Computational mechanics enhanced by deep learning". *Computer Methods in Applied Mechanics and Engineering,* ***327***, pp. 327 – 351. Advances in Computational Mechanics and Scientific Computation—the Cutting Edge.

[20] Charles, R. Q., Su, H., Kaichun, M., and Guibas, L. J., 2017. "Pointnet: Deep learning on point sets for 3d classification and segmentation". *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jul.

[21] Madani, A., Bakhaty, A., Kim, J., Mubarak, Y., and Mofrad, M., 2019. "Bridging finite element and machine learning modeling: stress prediction of arterial walls in atherosclerosis". *Journal of biomechanical engineering*.

[22] He, K., Zhang, X., Ren, S., and Sun, J., 2016. "Deep residual learning for image recognition". In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 770–778.

[23] Hu, J., Shen, L., and Sun, G., 2018. "Squeeze-and-excitation networks". In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 7132–7141.

[24] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y., 2014. "Generative adversarial nets". In *Advances in Neural Information Processing Systems 27*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, eds. Curran Associates, Inc., pp. 2672–2680.

[25] Goodfellow, I., 2016. "Nips 2016 tutorial: Generative adversarial networks". *arXiv preprint arXiv:1701.00160*.

[26] Isola, P., Zhu, J., Zhou, T., and Efros, A. A., 2016. "Image-to-image translation with conditional adversarial networks". *CoRR,* ***abs/1611.07004***.

[27] Ledig, C., Theis, L., Huszár, F., Caballero, J., Cunningham, A., Acosta, A., Aitken, A., Tejani, A., Totz, J., Wang, Z., et al., 2017. "Photo-realistic single image super-resolution using a generative adversarial network". In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 4681–4690.

[28] Liu, M.-Y., Breuel, T., and Kautz, J., 2017. "Unsupervised image-to-image translation networks". In Advances in Neural Information Processing Systems, pp. 700–708.

[29] Chrysos, G. G., Kossaifi, J., and Zafeiriou, S., 2018. "Robust conditional generative adversarial networks". *arXiv preprint arXiv:1805.08657*.

[30] Radford, A., Metz, L., and Chintala, S., 2015. "Unsupervised representation learning with deep convolutional generative adversarial networks". *CoRR,* ***abs/1511.06434***.

[31] Springenberg, J. T., Dosovitskiy, A., Brox, T., and Riedmiller, M. A., 2014. "Striving for simplicity: The all convolutional net". *CoRR,* ***abs/1412.6806***.

[32] Nair, V., and Hinton, G. E., 2010. "Rectified linear units improve restricted boltzmann machines". In Proceedings of the 27th International Conference on International Conference on Machine Learning, ICML'10, Omnipress, pp. 807–814.

[33] Maas, A. L., 2013. "Rectifier nonlinearities improve neural network acoustic models".

[34] Ioffe, S., and Szegedy, C., 2015. "Batch normalization: Accelerating deep network training by reducing internal covariate shift". *CoRR,* ***abs/1502.03167***.

[35] Farimani, A. B., Gomes, J., Sharma, R., Lee, F. L., and Pande, V. S., 2018. "Deep learning phase segregation". *CoRR,* ***abs/1803.08993***.

[36] Farimani, A. B., Gomes, J., and Pande, V. S., 2017. "Deep learning the physics of transport phenomena". *CoRR,* ***abs/1709.02432***.

[37] Lee, S., and You, D., 2018. "Data-driven prediction of unsteady flow fields over a circular cylinder using deep learning". *arXiv e-prints*, Apr., p. arXiv:1804.06076.

[38] Paganini, M., de Oliveira, L., and Nachman, B., 2018. "CaloGAN: Simulating 3D high energy particle showers in multilayer electromagnetic calorimeters with generative adversarial networks". *,* ***97***, Jan., p. 014021.

[39] Enomoto, K., Sakurada, K., Wang, W., Fukui, H., Matsuoka, M., Nakamura, R., and Kawaguchi, N., 2017. "Filmy cloud removal on satellite imagery with multispectral conditional generative adversarial nets". *CoRR,* ***abs/1710.04835***.

[40] Schawinski, K., Zhang, C., Zhang, H., Fowler, L., and Santhanam, G. K., 2017. "Generative adversarial networks recover features in astrophysical images of galaxies beyond the deconvolution limit". *,* ***467***, May, pp. L110–L114.

[41] Ravanbakhsh, S., Lanusse, F., Mandelbaum, R., Schneider, J., and Poczos, B., 2016. "Enabling Dark Energy Science with Deep Generative Models of Galaxy Images". *arXiv e-prints*, Sept., p. arXiv:1609.05796.

[42] Mustafa, M., Bard, D., Bhimji, W., Lukić, Z., Al-Rfou, R., and Kratochvil, J., 2017. "CosmoGAN: creating high-fidelity weak lensing convergence maps using Generative Adversarial Networks". *arXiv e-prints*, June, p. arXiv:1706.02390.

[43] Ledig, C., Theis, L., Huszar, F., Caballero, J., Aitken, A. P., Tejani, A., Totz, J., Wang, Z., and Shi, W., 2016. "Photo-realistic single image super-resolution using a generative adversarial network". *CoRR,* ***abs/1609.04802***.

[44] Gómez, J., and Guarín-Zapata, N., 2018. "Solidspy: 2d-finite element analysis with python."". *Parameters,* ***50***, pp. 2–0.

[45] Kingma, D. P., and Ba, J., 2014. "Adam: A method for stochastic optimization". *CoRR,* ***abs/1412.6980***.