3rd International Conference on Innovative Mechanisms for Industry Applications (ICIMIA 2023)
DVD Part Number: CFP23K58-DVD; ISBN: 979-8-3503-4362-5

Convolutional Neural Network Approach for Surrogate Modelling of the Torsion Problem

Jordan Tsz Chun Fung
Faculty of Mechanical Engineering
Technion – Israel Institute of Technology
Haifa, Israel
tsz-chun-f@campus.technion.ac.il

***Abstract*—This study explores the use of Convolutional Neural Networks (CNNs) in surrogate modelling for solving the torsion problem of elastic bars with arbitrary cross-section profiles. The conventional approach to solving this problem, the finite elements method (FEM), is computationally expensive and time-consuming, limiting its use in real-time applications. To address this, Convolutional Neural Networks (CNNs), a deep learning technique that has shown success in various domains is used. By training CNN models on datasets generated using FEM, this study bypass the FEM process and efficiently obtain the desired stress information. Further, a comparative analysis of different CNN architectures are done to obtain the optimal trade-off between accuracy and efficiency. This study adds to the expanding collection of literature exploring the application of deep learning methods in the field of computational mechanics.**

***Index Terms*—Computational mechanics, convolutional neural network, deep learning, finite elements method, machine learning, surrogate modelling, stress analysis, theory of elasticity**

I. Introduction

Deep learning techniques have achieved remarkable success across a variety of domains in recent years. [1] In particular, convolutional neural networks (CNNs) have been proven to be highly effective for tasks such as image processing and recognition. Through supervised learning, CNNs comprised of multiple processing layers with adjustable parameters allow computers to learn features and representations of an input-output dataset through stochastic gradient descent and back-propagation.

Elastic bars are widely used in mechanical designs. First formulated by Saint-Venant in 1847, stress analysis of an elastic bar under torsional load is an important problem in elasticity theory and engineering. [2] Knowledge of the shear stress distribution in the elastic bars, in particular the magnitude and location of maximal stress, is crucial to identifying potential points of failure. However, analytical solutions are available only for a limited number of cross-section profiles. [3]

To solve the torsion problem for a bar with an arbitrary cross-section profile, the finite elements method (FEM) can be used. [4] A mesh is constructed to divide the domain into local elements with discrete nodal points. The weak formulation of the partial differential equations governing the problem is then applied to these elements, creating a linear system of equations that can be solved for the nodal solutions. Lastly, post-processing is performed to obtain the desired information.

The FEM process is computationally expensive and time-consuming, which precludes its use in applications where real-time computation is required, such as design optimization and post-disaster recovery. To provide approximate solutions that are more efficient to compute, surrogate modelling techniques based on CNNs can be used, which involves the training of CNN models on datasets generated using FEM. Subsequently, this study bypass the FEM process and instead perform inference of the CNN model on the input to obtain the desired information at a fraction of the time and computational cost of FEM. [5]

Surrogate modelling techniques based on CNNs have recently been used to solve various problems in computational mechanics. In [6] and [7], Trent et al. and Bolando et al. used such techniques to model the stress distribution of loaded plates with rectangular and pentagonal shapes respectively. In [8], Hashemi et al. used such techniques to model the dynamical response of a truss structure under sinusoidal load. In [9] and [10], Mianroodi et al. and Khorrami et al. used such techniques to model the stress distribution in material microstructures. In [11], Lin et al. used such techniques to model the stress distribution of steel samples under tensile test. In [12], Viquerat et al. used such techniques to model the fluid drag over arbitrary 2D shapes in laminar flow. In [13], Liang et al. used such techniques for stress analysis of the aorta artery.

In the present study, we apply surrogate modelling techniques based on CNNs to solve the torsion problem of an elastic bar with an arbitrary cross-section profile. We further provide a comparative analysis of different CNN architectures to obtain the optimal trade-off between accuracy and efficiency. This study adds to the expanding collection of literature exploring the application of deep learning methods in the field of computational mechanics.

II. Methodology

We first define the boundary value problem (BVP) governing the torsion problem of an elastic bar. To prepare the dataset for training of CNNs, we generate a set of random 2D shapes which are used as the dataset's input bar cross-section profiles. For the output labels, we use MATLAB to compute the FEM solution of the BVP in these cross-section domains. Several CNN architectures are presented, and for each architecture, a

CNN is trained on the dataset, taking the bar cross-section profiles as input and predicting the magnitude and location coordinates of maximal stress.

A. *Torsion problem of an elastic bar*

We consider an isotropic homogeneous long prismatic bar with a simply connected cross-section domain Ω and boundary Γ in the y, z plane, which is fixed at $x = 0$ and twisted by angle $l\alpha$ at $x = l$, as depicted in Fig. 1.

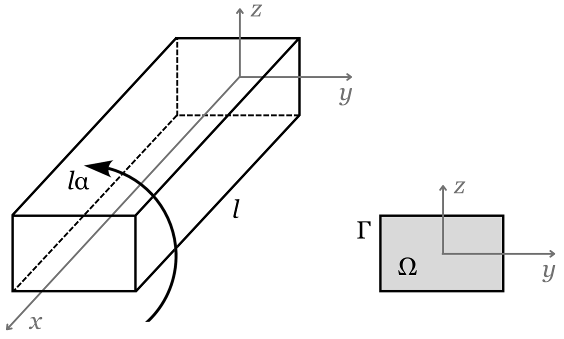

Fig. 1. An elastic bar under torsional load.

The torsion problem of such bars can be formulated using the Airy stress function $\Phi(y, z)$ as a BVP. [3]

$$\Delta\Phi = \frac{\partial^2\Phi}{\partial y^2} + \frac{\partial^2\Phi}{\partial z^2} = -2 \quad \text{in} \quad \Omega \tag{1}$$

$$\Phi = 0 \quad \text{on} \quad \Gamma \tag{2}$$

The stress components τ_{xy} and τ_{xz} can then be given by partial derivatives of $\Phi(y, z)$, where μ is the shear modulus.

$$\tau_{xy} = \alpha\mu\frac{\partial\Phi(y,z)}{\partial z}, \quad \tau_{xz} = -\alpha\mu\frac{\partial\Phi(y,z)}{\partial y} \tag{3}$$

The modulus of shear stress $|T|$ is defined accordingly. In subsequent sections, we consider the case where $\alpha = \mu = 1$.

$$|T| = \sqrt{\tau_{xy}^2 + \tau_{xz}^2} \tag{4}$$

B. *Generation of random 2D shapes*

To generate random 2D shapes for the dataset's input bar cross-section profiles, we use composite cubic Bézier curves to interpolate through a number of random points. [12] The general equation for a segment of Bézier curve is:

$$C(t) = \sum_{i=0}^{n} \binom{n}{i} t^i (1-t)^{n-i} P_i \tag{5}$$

Where, for cubic Bézier curves, $n = 3$ and four control points P_i are required, and parameter $t \in [0, 1]$. The segment starts at the first control point and ends at the fourth. For continuity, the second control point of one segment should be the reflection of the third control point of the previous segment across the joining point. Fig. 2 shows a shape generated using composite cubic Bézier curves.

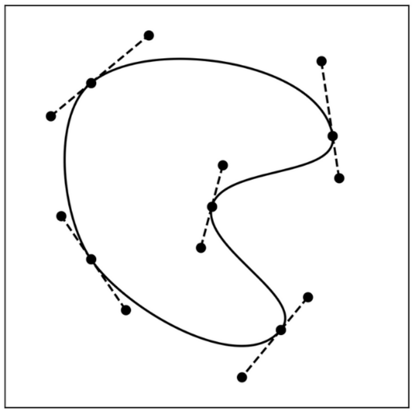

Fig. 2. A shape generated using composite cubic Bézier curves with control points and joint tangents indicated.

The number of random points used and the magnitude of the joint tangents, which relates to the smoothness of the curve, are randomised when generating shapes. A total of 50,000 shapes are generated, which are stored as binary images of 128×128 pixels in comma-separated values (CSV) format. Some examples of the generated random shapes are presented in Fig. 3.

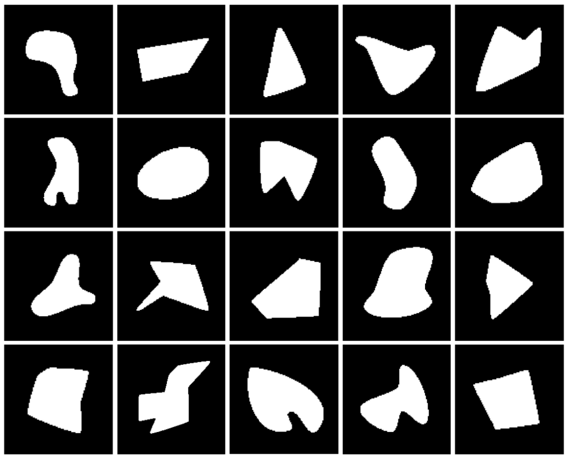

Fig. 3. Examples of shapes generated using composite cubic Bézier curves with the number of points used and joint tangents magnitude randomised.

C. *FEM solution*

For each of the shapes generated, we use MATLAB to compute the FEM solution of the BVP in the cross-section domain. First, the Image Processing Toolbox is used to obtain the shape contours of the input image. The shape is then meshed into linear triangular elements. The Partial Differential Equations (PDE) Toolbox is used to solve for the nodal solutions of $\Psi(y, z)$. Lastly, post-processing is performed to compute the modulus of shear stress $|T|$. The magnitude and location coordinates of maximal stress are stored as output labels in CSV format.

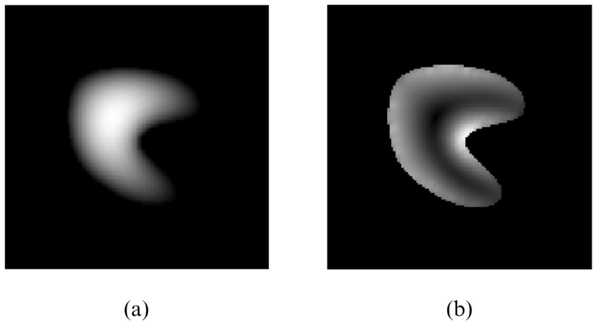

Fig. 4. FEM results for the torsion problem of an elastic bar with the cross-section profile in Fig. 2: (a) Airy stress function Ψ, (b) modulus of shear stress $|T|$ in normalized greyscale images.

D. CNN architecture

CNN model architectures typically comprises multiple convolutional layers, pooling layers, non-linear activation layers, and fully-connected layers. [17] Convolutional layers perform the convolution or cross-correlation operation on the input image using a number of kernels to compute feature maps. In pooling layers, such feature maps are sub-sampled to reduce spatial dimensions and to extract the dominant features. An activation function is then applied to introduce non-linearity, such that complex decision boundaries can be learned. Fully-connected layers are typically used at the end of a CNN to flatten and combine the features in order to obtain the final output.

Early CNN architectures include LeNet [18], which is applied to the recognition of handwritten digits. In [19], Krizhevsky et al. proposed AlexNet which pioneered the use of dropout layers and rectified linear unit (ReLU) activation. VGGNet [20] increased the depth of the CNN and used small-size kernel filters. To address the problem of vanishing gradients, ResNet [14] uses residual blocks with shortcut connections, as shown in Fig. 5(a). This architecture is further improved with ResNetV2. [21]

In [15], Szegedy et al. proposed Inception blocks which concatenate the outputs from kernels of different sizes, as shown in Fig. 5(b). This architecture is further improved with InceptionV3 [22], Xception [23], as well as InceptionResNet [24] which incorporates residual connections. Other architectures under consideration include DenseNet blocks [16] which concatenate outputs from all preceding layers, as depicted in Fig. 5(c), and EfficientNet [25] [26] which incorporated the compound coefficient technique for model scaling.

In the present study, the CNN architectures are adapted to account for the input image shape of (128,128,1) and an output dimension of 3 with linear activation.

E. Training of the CNNs

Prior to the training of the CNNs, we pre-process the dataset by performing feature scaling on the coordinate and stress values, which are normalized to [0,1]. We then perform data augmentation by transposing the images by the two diagonals, expanding the dataset to $n = 150,000$ samples. We split the dataset randomly to assign 80% of samples to the training set and the remaining 20% to the validation set.

For the training of the CNNs, we define the mean squared error (MSE) loss function, where Y_i and $\hat{Y}_i$ are the true and predicted labels respectively. The MSE can also be used as an evaluation metric for the maximal stress location coordinates, which is equivalent to the mean Euclidean squared distance between the true and predicted points.

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2 \tag{6}$$

We also define the mean absolute percentage error (MAPE) as an evaluation metric for the maximal stress magnitude.

$$\text{MAPE} = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{Y_i - \hat{Y}_i}{Y_i}\right| \tag{7}$$

We use the Adaptive Moment Estimation (Adam) optimizer to train the CNN models. [27] In every training step, the parameters of the CNN model are updated to minimise the loss function through gradient descent and back-propagation.

III. Results

The CNN models are implemented in TensorFlow and trained on a high-performance computer equipped with an NVIDIA A100 80GB GPU using a batch size of 64. To ensure the models will generalize well to new input cross-section profiles, we train for up to 20 epochs following the last improvement in total MSE loss on the validation dataset, and save the model with the lowest validation loss. The training results of the CNN models, including inference time per step and evaluation metrics, are presented in Table I.

A. Time efficiency

When choosing the most appropriate CNN surrogate model architecture, it is important to consider the trade-off between the accuracy of results and time efficiency. An inaccurate model is of no practical engineering use, but a slow inference time defeats the purpose of using a surrogate model over FEM.

One major factor is the computational resources available for inference of the CNN models. There might be use-case scenarios where high-performance GPUs are not available, and only CPUs are used. For comparison, we measure the inference time of the CNN models on a machine equipped with an NVIDIA A100 80GB GPU, as well as on a MacBook Pro with an Intel CPU. Fig. 6 and 7 show the plots of validation losses against the GPU and CPU inference time per step of the trained CNN models.

B. Model size

Another criterion for choosing a CNN surrogate model architecture is the model size or the number of model parameters. A larger model will require more memory and storage space for performing inference. Fig. 8 shows the plots of validation losses against the number of model parameters.

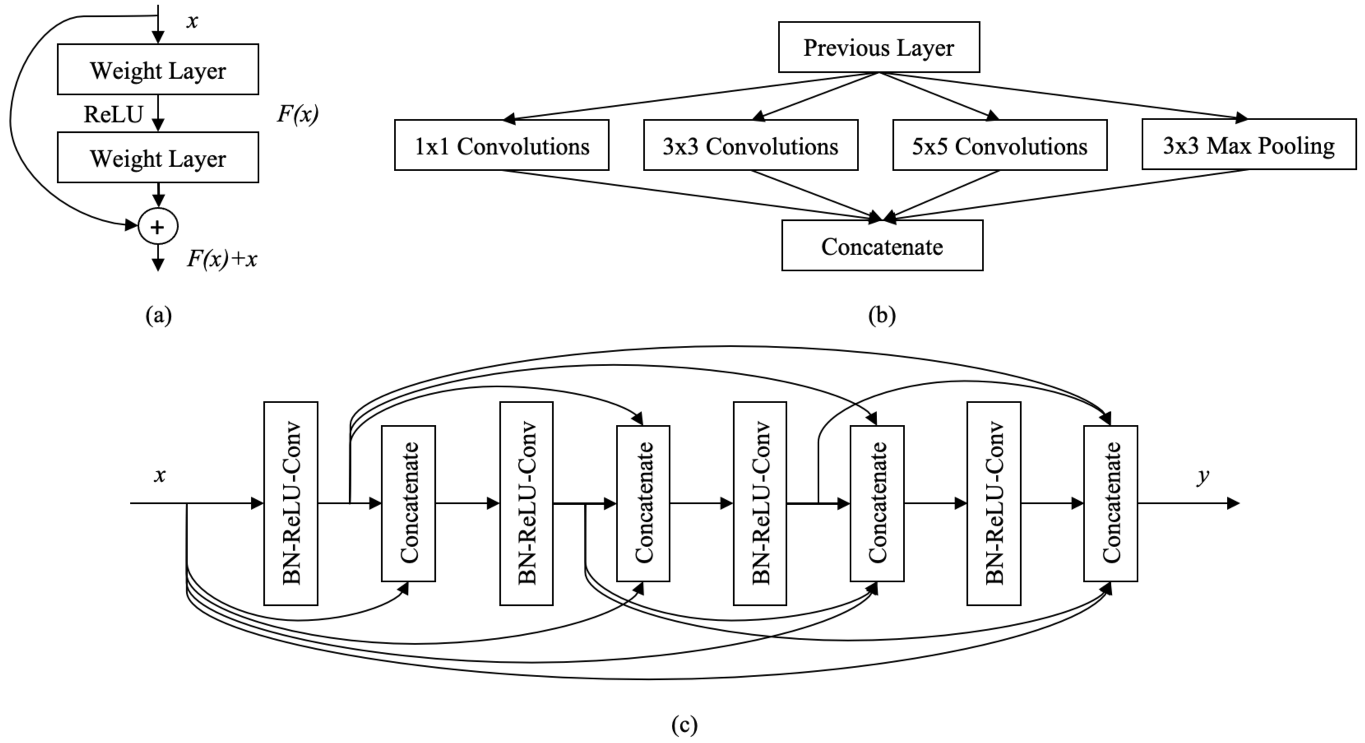

Fig. 5. CNN architectures: (a) Residual block [14], (b) Naïve version of Inception block [15], (c) DenseNet block [16]

C. Optimal CNN architecture

Based on these considerations and the training results, we evaluate the CNN model architectures. We can observe that the EfficientNet models perform worse than all other architectures, and thus we will not consider these models.

The best performance by total MSE is achieved with DenseNet201 and InceptionResNetV2 models, but this comes at a cost of time efficiency as both have a GPU inference time per step of 21 ms. In addition, while these models perform well in predicting the maximal stress location coordinates, their performance is average in predicting the stress magnitude.

The third-best model architecture, DenseNet121, exhibits good performance in predicting both the magnitude and the location coordinates of maximal stress. At just 7.03 million parameters and a GPU inference time per step of 12 ms, it achieved an MSE of 3.5646×10^{-3} for the maximal stress location coordinates and an MAPE of 2.9057% for the magnitude. It appears to be the optimal trade-off between accuracy of results and efficiency. Fig. 9 shows examples of inference results of the trained DenseNet121 model on some input cross-section profiles.

IV. Conclusions

In the present study, we trained surrogate models using convolutional neural networks (CNNs) to solve the torsion problem of elastic bars with arbitrary cross-section profiles.

To generate the dataset for training of CNNs, we first used composite cubic Bézier curves to generate a set of 50,000 random 2D shapes which are stored as binary images. Taking these shapes as input cross-section profiles, we use MATLAB to compute the finite elements method (FEM) solutions to the boundary value problem governing the torsion of an elastic bar and stored the magnitude and location coordinates of maximal stress as output labels. We performed feature scaling and data augmentation to expand the dataset to 150,000 samples.

We trained the CNN models with 12 different architectures and conducted a comparative analysis based on the evaluation metrics and inference time. We concluded that among the architectures considered, DenseNet121 presents the optimal trade-off between accuracy and efficiency. With 7.03 million parameters and a GPU inference time per step of 12 ms, it achieved an MSE of 3.5646×10^{-3} for the maximal stress location coordinates and an MAPE of 2.9057% for the magnitude.

The findings of the present study show the viability of using surrogate models based on CNNs to bypass the time-consuming and computationally expensive FEM process in the stress analysis of elastic bars with arbitrary cross-section profiles under torsion.

This study adds to the expanding collection of literature exploring the application of deep learning methods in the field of computational mechanics. Future research could investigate the use of alternative CNN architectures to improve performance, as well as the application of surrogate modelling techniques based on CNNs in other engineering problems.

TABLE I
TRAINING RESULTS OF THE CNN MODELS

CNN Architecture				**Inference Time Per Step**		**Evaluation Metrics (Validation)**		
#	Name	Ref	Parameters	GPU (ms)	CPU (ms)	Total MSE (e-3)	Coord MSE (e-3)	Max MAPE (%)
R1	ResNet50V2	[21]	23.6 M	7	33	2.5332	3.6807	3.2987
R2	ResNet101V2	[21]	42.6 M	11	59	2.5455	3.6974	3.2646
R3	ResNet152V2	[21]	58.3 M	16	82	2.5760	3.7034	3.7070
I1	InceptionV3	[22]	21.8 M	10	23	2.5849	3.7369	4.0755
I2	Xception	[23]	20.9 M	5	37	2.5966	3.6261	4.5185
I3	InceptionResNetV2	[24]	54.3 M	21	53	2.4259	3.4810	3.2870
E1	EfficientNetV2B0	[26]	5.92 M	9	14	2.9678	4.2907	4.1292
E2	EfficientNetV2S	[26]	20.3 M	16	38	2.8995	4.2328	3.2296
E3	EfficientNetV2M	[26]	53.2 M	25	73	2.8504	4.1497	3.3052
D1	DenseNet121	[16]	7.03 M	12	44	2.4411	3.5646	2.9057
D2	DenseNet169	[16]	12.6 M	16	57	2.4881	3.6230	2.9073
D3	DenseNet201	[16]	18.3 M	21	71	2.4205	3.5169	3.5357

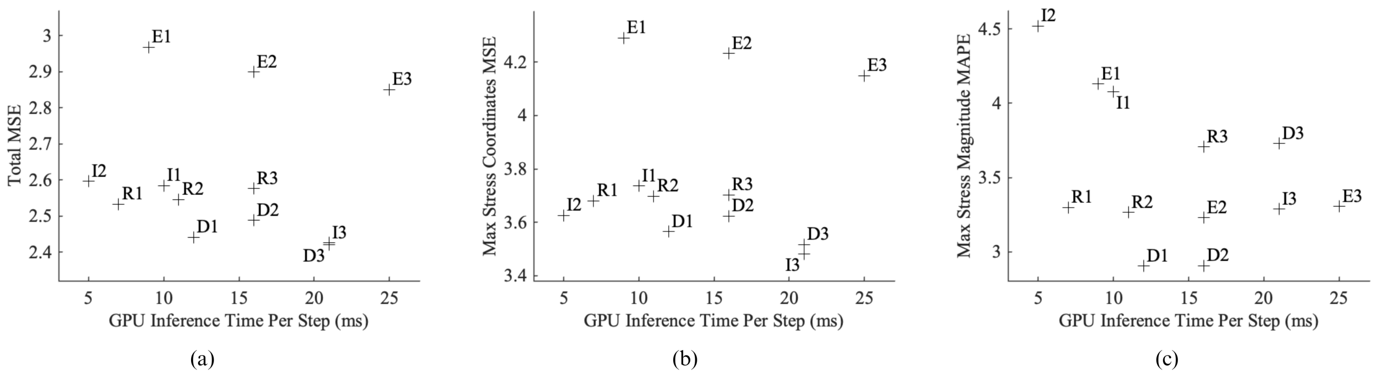

Fig. 6. GPU inference time plotted against (a) total MSE, (b) max stress coordinates MSE and (c) max stress magnitude MAPE of the CNN models.

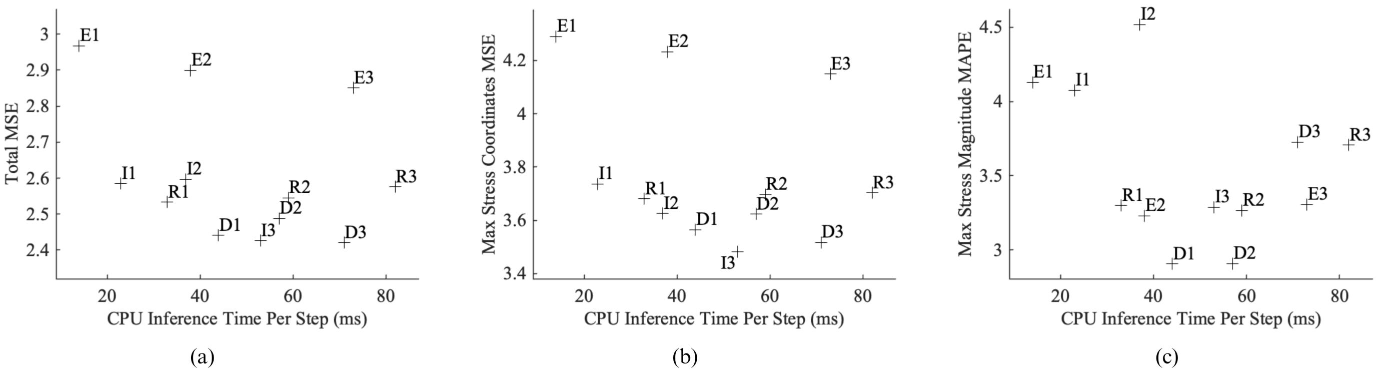

Fig. 7. CPU inference time plotted against (a) total MSE, (b) max stress coordinates MSE and (c) max stress magnitude MAPE of the CNN models.

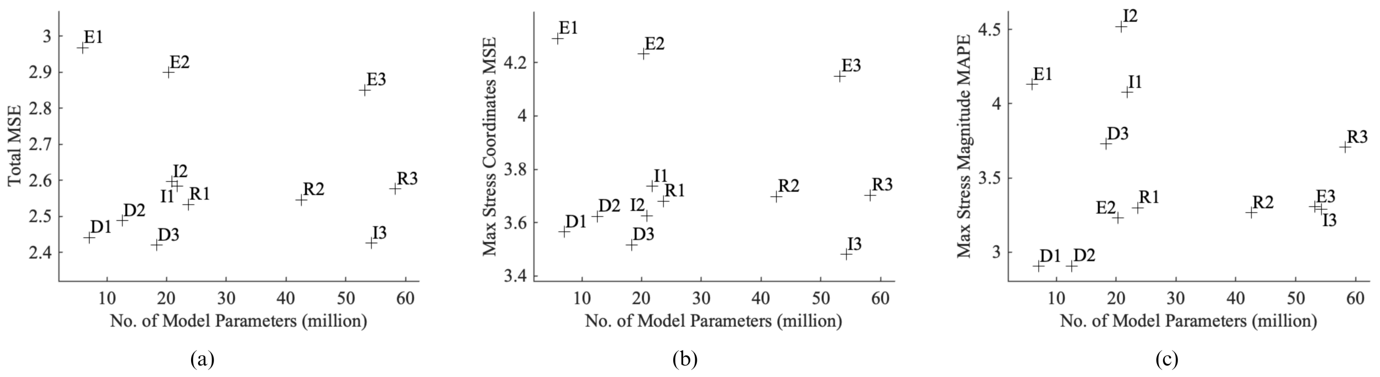

Fig. 8. Number of model parameters plotted against (a) total MSE, (b) max stress coordinates MSE and (c) max stress magnitude MAPE of the CNN models.

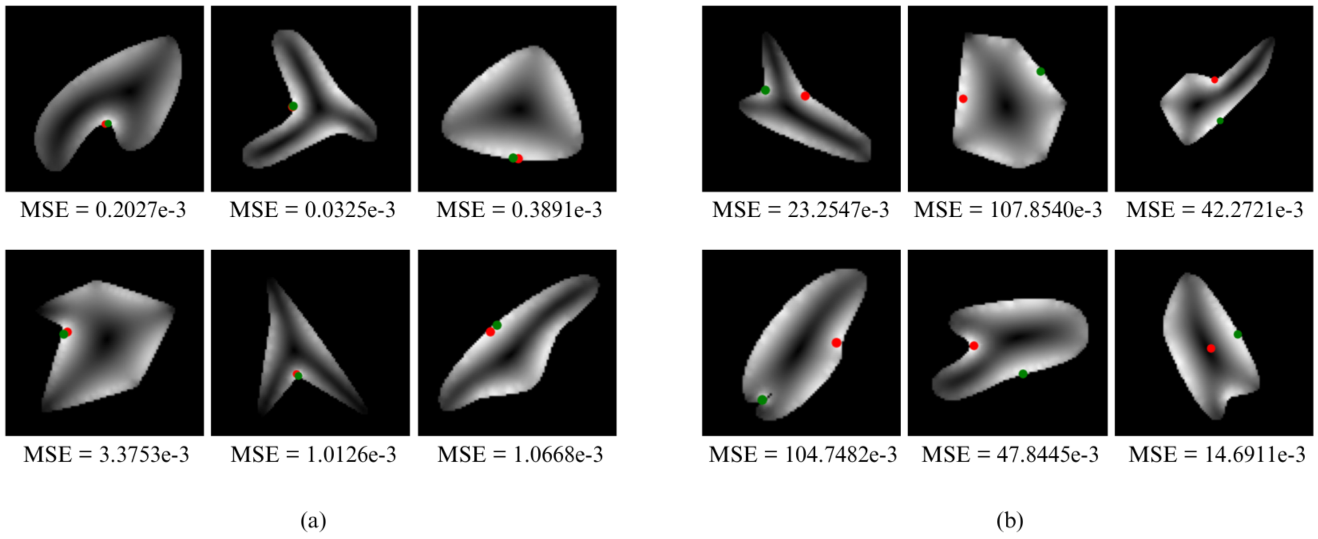

Fig. 9. Inference results of the trained DenseNet121 model on some input cross-section profiles (a) with satisfactory performance (low total MSE) and (b) with poor performance (high total MSE); green and red dots indicate the true and predicted points of maximal stress respectively.

References

[1] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, 2015.

[2] M. Sadd, *Elasticity: Theory, Applications, and Numerics*. Elsevier, 2014.

[3] J. Francu, P. Novackova, and P. Janicek, "Torsion of a non-circular bar," *Engineering MECHANICS*, vol. 19, 2012.

[4] M. G. Larson and F. Bengzon, *The Finite Element Method: Theory, Implementation, and Applications*. Springer-Verlag Berlin Heidelberg, 2013.

[5] J. Kudela1 and R. Matousek1, "Recent advances and applications of surrogate models for finite element method computations: a review," *Soft Computing*, 2022.

[6] S. Trent, J. Renno, S. Sassi, and M. S. Mohamed, "Using image processing techniques in computational mechanics," *Computers and Mathematics with Applications*, 2023.

[7] H. Bolando, X. Li, T. Salem, V. N. Boddeti, and N. Lajnef, "Bridging finite element and deep learning: High-resolution stress distribution prediction in structural components," *Front. Struct. Civ. Eng.*, 2022.

[8] A. Hashemi, J. Jang, and J. Beheshti, "A machine learning-based surrogate finite element model for estimating dynamic response of mechanical systems," *IEEE Access*, vol. 11, pp. 54 509–54 525, 2023.

[9] J. R. Mianroodi, N. H. Siboni, and D. Raabe, "Teaching solid mechanics to artificial intelligence—a fast solver for heterogeneous materials," *npj Computational Materials*, vol. 7, 2021.

[10] M. S. Khorrami1, J. R. Mianroodi, N. H. Siboni, P. Goyal, B. Svendsen, P. Benner, and D. Raabe, "An artificial neural network for surrogate modeling of stress fields in viscoplastic polycrystalline materials," *npj Computational Materials*, vol. 9, 2023.

[11] B. Lin, S. Medghalchi, S. Korte-Kerzel, and B.-X. Xu, "A machine learning enabled image-data-driven end-to-end mechanical field predictor for dual-phase steel," *PAMM*, vol. 22, no. 1, p. e202200110, 2023.

[12] J. Viquerat and E. Hachem, "A supervised neural network for drag prediction of arbitrary 2d shapes in laminar flows at low reynolds number," *Computers and Fluids*, 2020.

[13] L. Liang, M. Liu, C. Martin, and W. Sun, "A deep learning approach to estimate stress distribution: a fast and accurate surrogate of finite-element analysis," *Journal of the Royal Society Interface*, 2018.

[14] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015.

[15] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Los Alamitos, CA, USA: IEEE Computer Society, jun 2015, pp. 1–9.

[16] G. Huang, Z. Liu, L. V. D. Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Los Alamitos, CA, USA: IEEE Computer Society, jul 2017, pp. 2261–2269.

[17] L. e. a. Alzubaidi, "Review of deep learning: concepts, cnn architectures, challenges, applications, future directions," *Journal of Big Data*, 2021.

[18] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[19] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in Neural Information Processing Systems 25 (Neurips)*, 2012.

[20] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2015.

[21] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," 2016.

[22] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Los Alamitos, CA, USA: IEEE Computer Society, jun 2016, pp. 2818–2826.

[23] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Los Alamitos, CA, USA: IEEE Computer Society, jul 2017, pp. 1800–1807.

[24] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," in *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, ser. AAAI'17. AAAI Press, 2017, p. 4278–4284.

[25] M. Tan and Q. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," in *Proceedings of the 36th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, K. Chaudhuri and R. Salakhutdinov, Eds., vol. 97. PMLR, 09–15 Jun 2019, pp. 6105–6114.

[26] ——, "Efficientnetv2: Smaller models and faster training," in *Proceedings of the 38th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, M. Meila and T. Zhang, Eds., vol. 139. PMLR, 18–24 Jul 2021, pp. 10 096–10 106.

[27] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2017.