

Supplementary Information

Deep Learning Framework for Material Design Space Exploration using Active Transfer Learning and Data Augmentation

Yongtae Kim^{1†}, Youngsoo Kim^{2†}, Charles Yang³, Kundo Park¹, Grace X. Gu³, Seunghwa Ryu

^{1*}

Affiliations

¹Department of Mechanical Engineering & KI for the NanoCentury, Korea Advanced Institute of Science and Technology (KAIST), 291 Daehak-ro, Yuseong-gu, Daejeon 34141, Republic of Korea

²Department of Nature-Inspired System and Application, Korea Institute of Machinery and Materials, Daejeon, (34103) 156, Republic of Korea

³Department of Mechanical Engineering, University of California, Berkeley, CA 94720, USA

[†]These authors contributed equally to this work.

*corresponding author: ryush@kaist.ac.kr

Keywords: Deep Learning; Active-Transfer learning; Data augmentation; Regression; Material design;

Supplementary Note 1. Prediction of deep neural network (DNN) upon unseen output value ranges with different activation functions.

To analyze the effect of different activation functions, we compared the predictive power of deep neural networks (DNNs) upon unseen output value ranges by employing different activation functions including hyperbolic tangent (Tanh), Rectified linear activation function (ReLU), Leaky ReLU, Exponential Linear Unit (eLU), Clipped ReLU. The activation functions are given as follows:

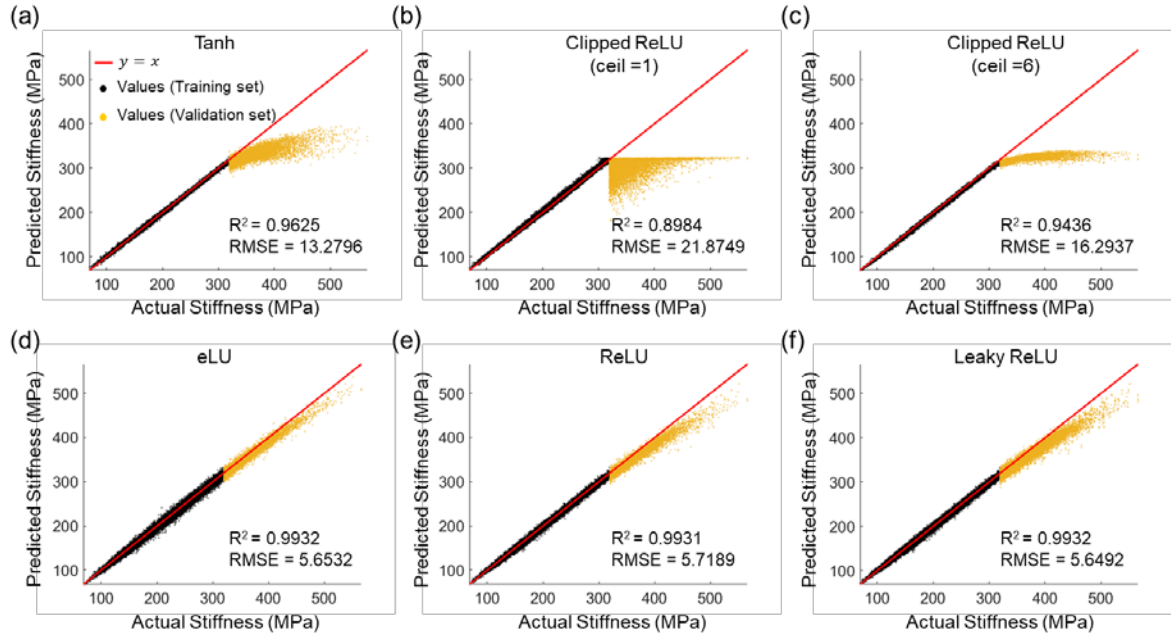
$$f^{\text{Tanh}} = \tanh(x), \quad f^{\text{Clipped ReLU}} = \begin{cases} \text{ceil} & x > \text{ceil} \\ \max(0, x) & x \leq \text{ceil} \end{cases}$$

$$f^{\text{ReLU}} = \max(0, x), \quad f^{\text{Leaky ReLU}} = \begin{cases} x & x > 0 \\ ax & x \leq 0 \end{cases},$$

$$f^{\text{eLU}} = \begin{cases} x & x > 0 \\ a(e^x - 1) & x \leq 0 \end{cases},$$

The Tanh and Clipped ReLU have bounded output values ranges $(-1, 1)$, and $(0, \text{ceil})$, respectively. We tested two different ceil values of 1 and 6 for Clipped ReLU. In contrary, ReLU, Leaky ReLU, and eLU have unbounded output value ranges. The coefficients for the Leaky ReLU and eLU are arbitrarily assigned as 0.1, and 1, respectively. For the fair comparison, other training conditions such as architectures, number of epochs, learning rate, and etc. are equally assigned. The DNN composed of these activation functions are trained on the dataset having lowest 90% stiffness values, and are tested to predict the dataset with highest 10% stiffness values. Because the DNN are trained on the dataset with low output value, the validation results can be regarded as the predictive performance of the DNN in unseen domain during the optimization process. The test results are visualized in Supplementary Figure 1. The DNN composed of bounded activation functions show relatively weak predictive power upon the unseen output value range, as represented in Supplementary Figure 1 (a-c). Especially,

DNN composed of Clipped ReLU with $\text{ceil} = 1$ show clearly bounded output values prediction. In contrary, as shown in Supplementary Figure 1 (d-f), unbounded activation functions show better predictive performances upon unseen domain with higher coefficient of determination (R^2) values and smaller root mean squared error (RMSE), as consistent with previous research¹⁻³. In the present study, we employed the DNN with the leaky ReLU because it shows the least RMSE value.

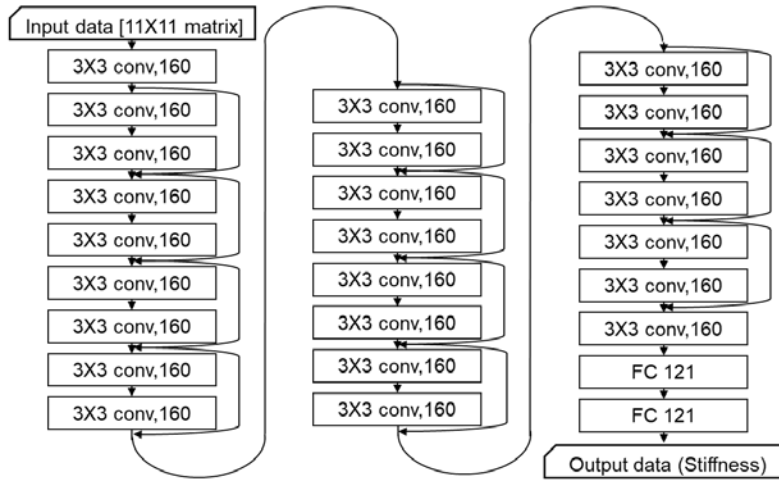


Supplementary Figure 1. Training results on the stiffness with DNNs composed of different activation functions. The training sets and the validation sets are selected as the dataset with the lowest 90% stiffness values and the highest 10% stiffness values, respectively. The coefficient of determination (R^2) is calculated by following equation: $R^2 = 1 - \frac{\sum (y_i - f_i)^2}{\sum (y_i - \bar{y})^2}$. y_i , f_i and $\bar{y}$ represent the actual value, fitted value and mean of actual values. The root mean squared error (RMSE) is calculated by following equation: $RMSE = \sqrt{\sum (y_i - f_i)^2 / N}$.

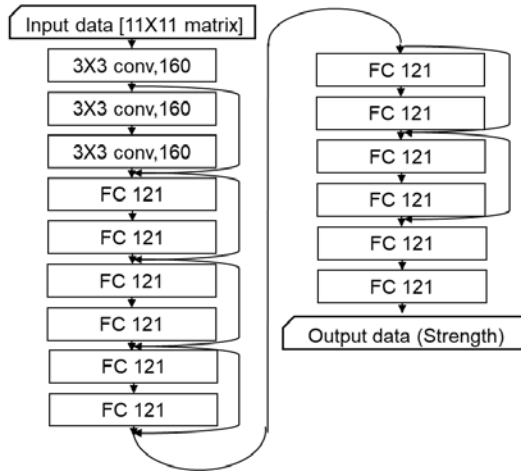
Supplementary Note 2. Architecture of deep neural network for stiffness, strength, and toughness prediction.

The architectures of deep neural network (DNN) adopted for stiffness, strength, toughness predictions are represented in Supplementary Figure 2 (a), (b), (c). These architectures are constructed based on the residual network ⁴, to secure enough number of learnable parameters without exploding or vanishing gradient problems. In addition, because it is known that the full pre-activation networks have good generalization performance, each residual unit is constructed as depicted in Supplementary Figure 2(d) ⁵. To obtain proper architectures, first, the optimal number of unit structures are explored by increasing the number of unit structures with the data that are randomly split into training and validation data with 9:1 ratio. After finding the optimal number with the lowest validation error, the architectures are fine-tuned by controlling the number of learnable parameters, e.g., the size of filters, the number of filters at each convolutional layer, the number of fully connected layers, the number of neurons at fully connected layer. We train the DNN having the fine-tuned architectures with the data which are split into training and validation data with 9:1 ratio in ascending order. If we attain the sufficient generalization performance, the architectures are finally adopted, and if not, the number of learnable parameters is reduced. The flowchart representing the process of constructing the DNN architecture is depicted in Supplementary Figure 3. In case of toughness prediction represented in Supplementary Figure 4, we found that the interpolation task on seen domain was possible (despite of relatively large R^2 value), but the predictive performance on unseen domain turns out to be very weak.

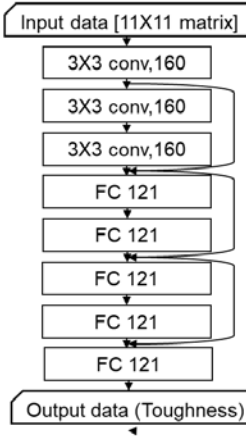
(a)



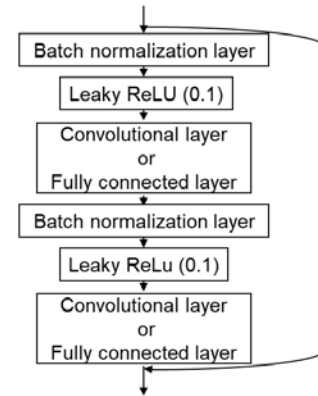
(b)



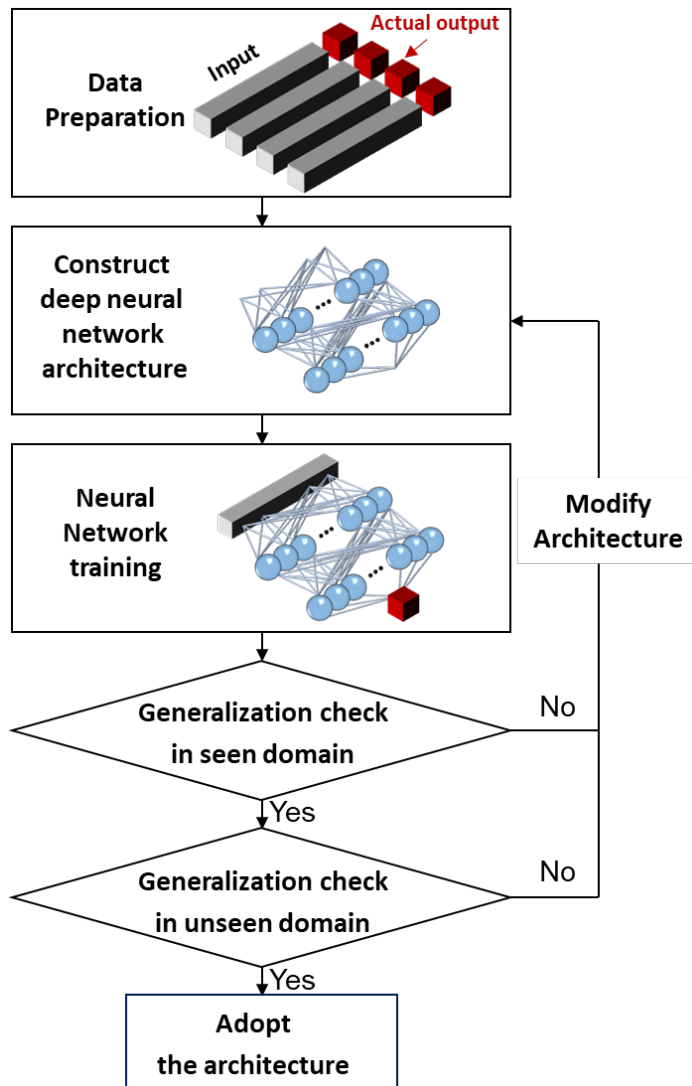
(c)



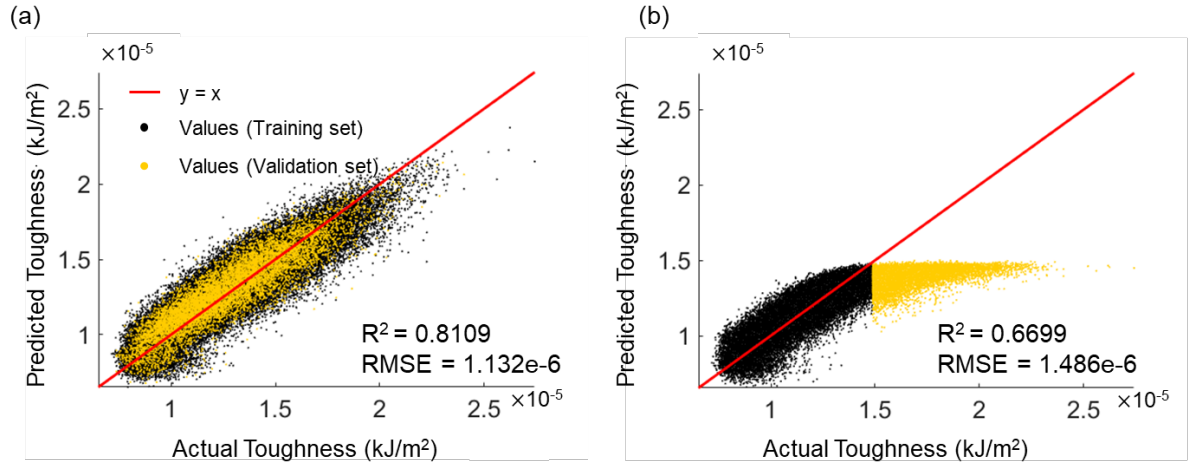
(d)



Supplementary Figure 2. (a-c) Neural network architecture for prediction of stiffness, strength, and toughness. The architectures are constructed based on Residual network, and (d) skipped layer are constructed with pre-activation to improve generalization of neural network.



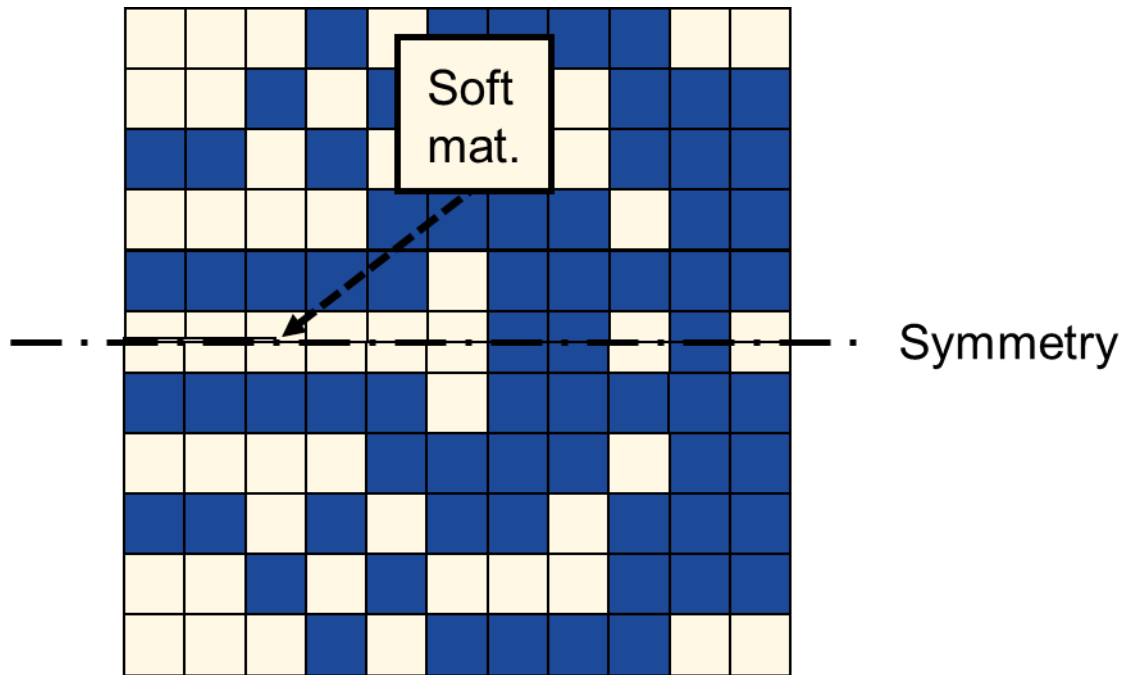
Supplementary Figure 3. Flow chart of constructing DNN architecture adopted for optimization leveraging the prediction upon unseen domain. The generalization performance of DNN should be double-checked on the basis of seen domain and unseen domain.



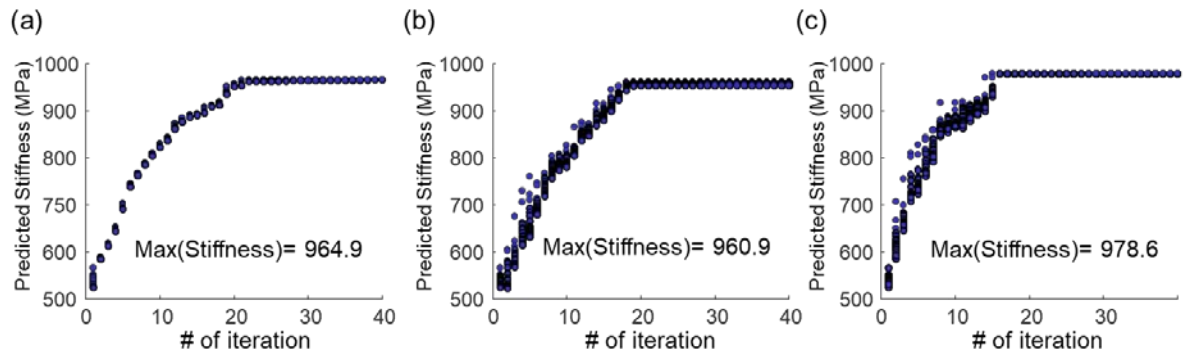
Supplementary Figure 4. (a) The DNN training result for toughness. The coefficient of determination (R^2) are calculated by following equation: $R^2 = 1 - \sum(y_i - f_i)^2 / \sum(y_i - \bar{y})^2$. y_i , f_i and $\bar{y}$ represent the actual value, fitted value and mean of actual values. The root mean squared error (RMSE) are calculated by following equation: $RMSE = \sqrt{\sum(y_i - f_i)^2 / N}$. (b) extrapolation validity check of trained architecture. The DNN is trained using 90 % of data having low values and validated using 10 % of data having high value in terms of stiffness and strength, respectively.

Supplementary Note 3. Effects of heuristics on the optimization process.

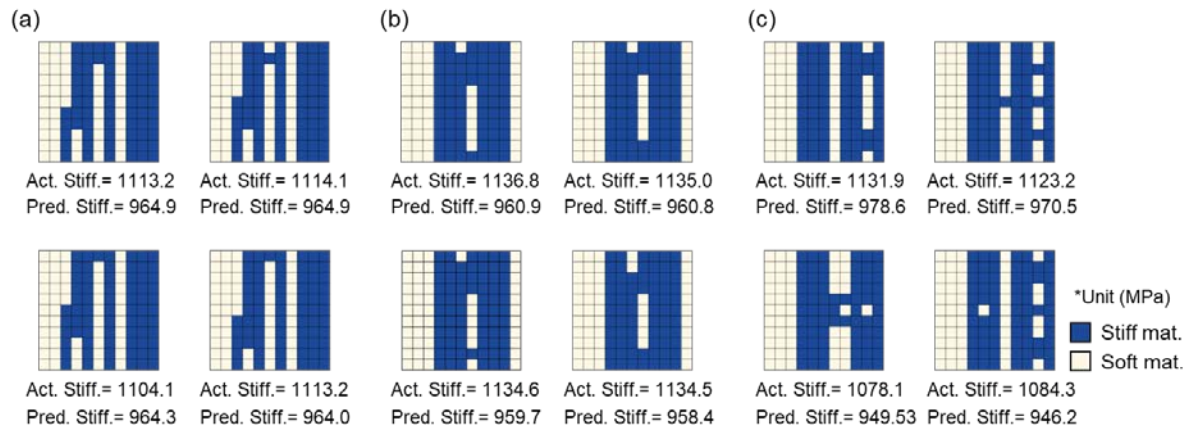
The intuition from the field of solid mechanics indicates that the symmetric structure is efficient for load bearing, and that the soft material at the crack is able to relieve the stress concentration at the crack tip, as shown in Supplementary Figure 5. These heuristic design principles are implemented in hyper-heuristic genetic algorithm by constraining the explorable design space to satisfy these two conditions. It accelerated the optimization process compared to conventional particle swarm optimization and conventional genetic algorithm without any heuristics, as represented Supplementary Figure 6. The comparisons are conducted with the DNN trained on initial training data set without the updates on DNN. The conventional particle swarm optimization process proceeds by random swapping stiff and soft blocks. The conventional genetic algorithm adopts the prescribed procedure in the main text without aforementioned heuristics. We find that the optimization with hyper-heuristic genetic algorithm generally converged to a higher optimal point with a smaller number of iterations compared to other algorithms. The composite structures designed by each optimization algorithms and corresponding stiffness values are represented in Supplementary Figure 7. Because the optimized structures are proposed by DNN on unseen domain, there are relatively large discrepancy between predicted stiffness and actual stiffness value. We note that most structures from optimization algorithms without the heuristics also has a soft material at the crack tip, and most stiff material blocks to the right side of structures in relatively symmetric manner, which is consistent with our intuition from solid mechanics.



Supplementary Figure 5. The composite structures have symmetrical structure with respect to the dashed line, and have soft material at the crack tip.



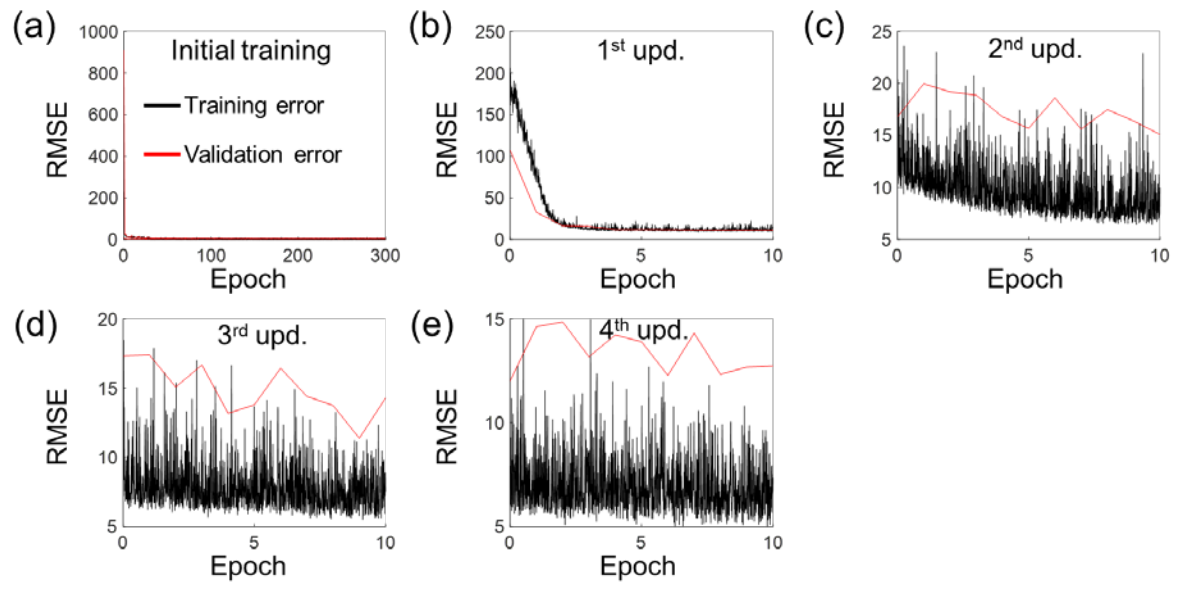
Supplementary Figure 6. The progresses of stiffness optimization process are compared with (a) particle swarm optimization, (b) genetic optimization, and (c) hyper-heuristic genetic optimization. The number of data chosen at each iteration is 30. The converged values can be varied due to the randomness in the optimization.



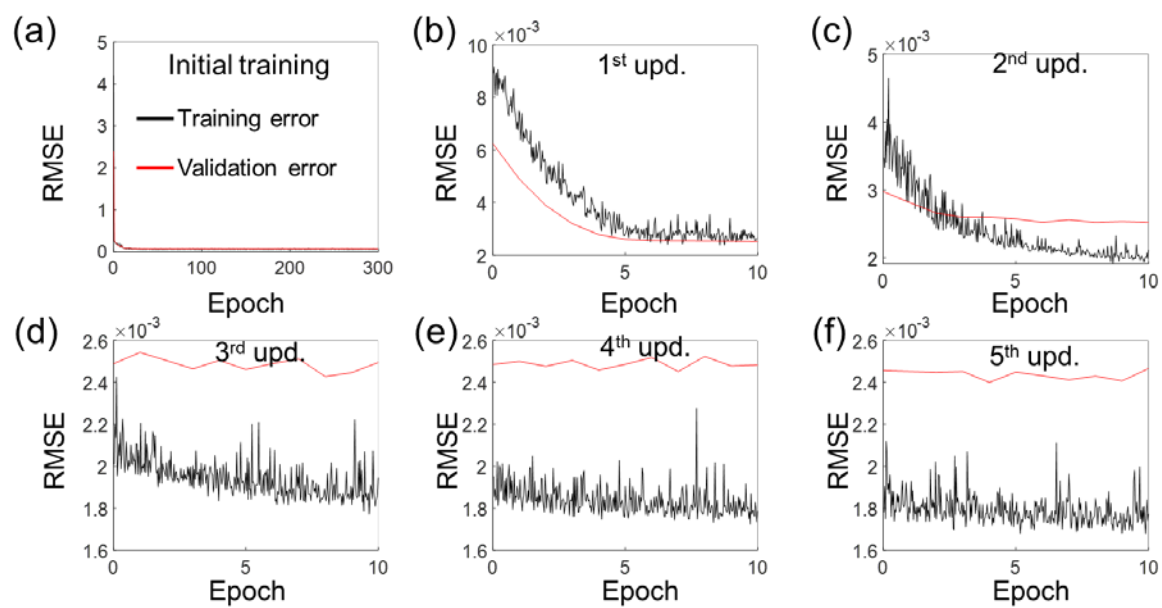
Supplementary Figure 7. The best composite structures designed by (a) particle swarm optimization, (b) genetic optimization, and (c) hyper-heuristics genetic optimization, respectively. The structures are selected at the final iteration of each optimization algorithm.

Supplementary Note 4. Details of DNN training histories.

It took approximately 8 hours for the initial training of the DNN with NVIDIA GeForce RTX 2080 Ti. The initial training were proceeded for 300 epochs, until the validation error converged. In case of the update procedures, the DNN is trained with reduced epoch of 10 and reduced learning rate of 10^{-6} . The stiffness and strength training histories for initial training and updates are depicted in Supplementary Figure 8-9. In the training histories for 1st and 2nd updates of stiffness and strength, the training error and validation error dramatically decreased because the DNN parameters were tuned significantly to expand the reliable prediction domain due to the relatively large number of additional data. However, as the updates proceed, the reduction of error looks insignificant because the number of additional data decreased and the training process converged. In the updates history of the strength optimization problem, the validation error is relatively large compared to training error. However, because the validation error converged, we believe that the training is well-performed.



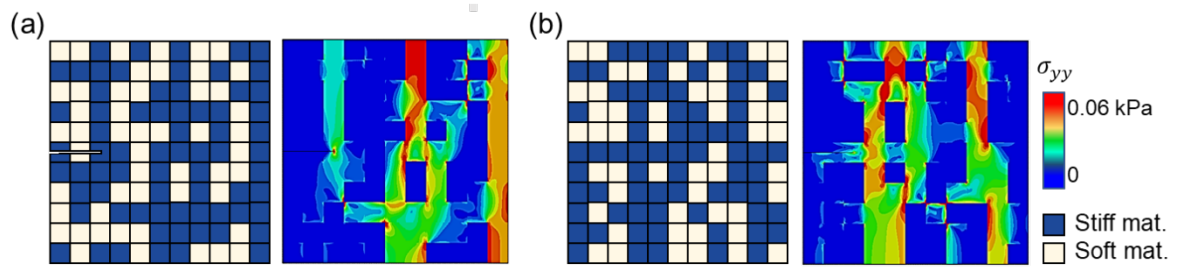
Supplementary Figure 8. Training histories for stiffness prediction at (a) initial DNN training, and (b-e) subsequent updates.



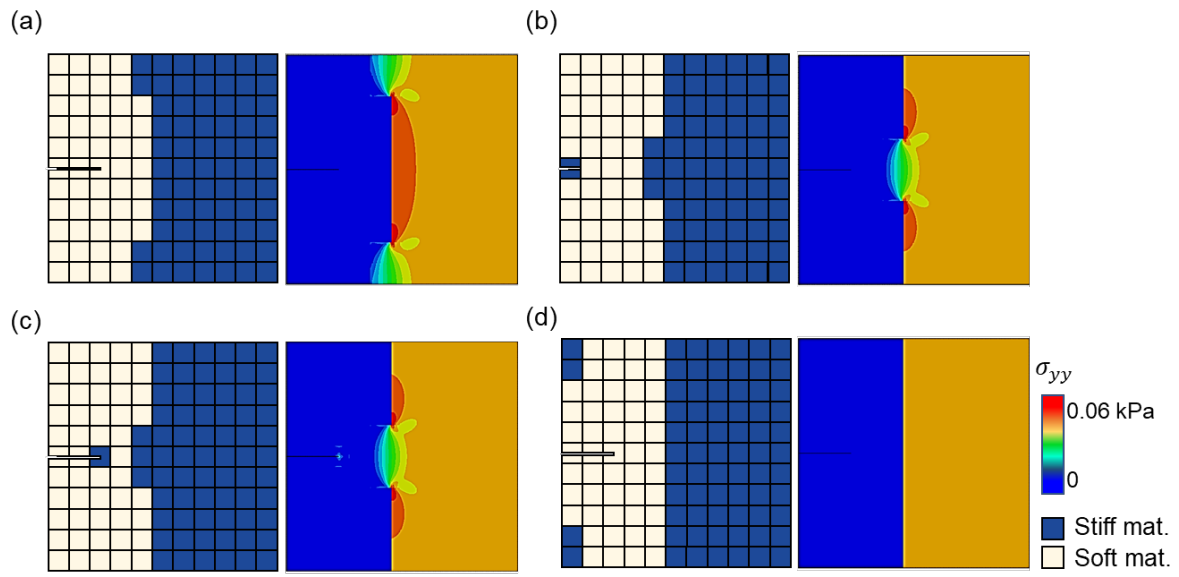
Supplementary Figure 9. Training histories for strength prediction at (a) initial DNN training, and (b-e) subsequent updates.

Supplementary Note 5. Stress distribution of each composite microstructure.

The optimized structures have less degree of stress concentration compared to the randomly chosen structures in training data set represented in Supplementary Figure 10. Because the severe stress concentration promotes early fracture at lower stress level, it is important to relieve the stress concentration to achieve high strength. To validate our results, the composite structures, which are expected to have weak stress concentration are manually designed, as represented in Supplementary Figure 11. We found that manually designed structures in Supplementary Figure 11 (a), (b), and (c) show relatively severe stress concentration compared to deep learning based optimized structures. Interestingly, although Supplementary Figure 11 (d) shows almost uniform stress distribution (which is much better than optimal design from the DNN-based search in Figure 6(b)), it has lower strength value of 0.15728 MPa due to its relatively low stiffness of 1109.7 MPa. Our manual design examples imply that the appropriate combination of high stiffness and small stress concentration is necessary to achieve the high strength. Our deep-learning-based forward design frameworks enables the search of the optimal design with the optimal compensation of each.



Supplementary Figure 10. (a-b) Randomly chosen composite structure and corresponding axial stress (σ_{yy}) are colored at $\varepsilon_{yy} = 2.2727 \times 10^{-5}$.



Supplementary Figure 11. (a-d) Composite structures based on heuristics for maximum strength, and corresponding axial stress (σ_{yy}) are colored at $\epsilon_{yy} = 2.2727 \times 10^{-5}$. The strength of each composite structure are 0.15798 MPa, 0.15794 MPa, 0.15823 MPa and 0.15728 MPa respectively.

Supplementary References

- 1 Sonoda, S. & Murata, N. Neural network with unbounded activation functions is universal approximator. *Applied and Computational Harmonic Analysis* **43**, 233-268 (2017).
- 2 Arora, R., Basu, A., Mianjy, P. & Mukherjee, A. Understanding deep neural networks with rectified linear units. *arXiv preprint arXiv:1611.01491* (2016).
- 3 Hein, M., Andriushchenko, M. & Bitterwolf, J. in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 41-50.
- 4 He, K., Zhang, X., Ren, S. & Sun, J. in *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770-778.
- 5 He, K., Zhang, X., Ren, S. & Sun, J. in *European conference on computer vision*. 630-645 (Springer).