Computers and Structures 277–278 (2023) 106940

Contents lists available at ScienceDirect

Computers and Structures

journal homepage: www.elsevier.com/locate/compstruc

Exploring the structure-property relations of thin-walled, 2D extruded lattices using neural networks

Junyan He [a], Shashank Kushwaha [a], Diab Abueidda [b], Iwona Jasiuk [a,*]

[a] Department of Mechanical Science and Engineering, University of Illinois at Urbana-Champaign, Champaign, IL, USA
[b] National Center for Supercomputing Applications, University of Illinois at Urbana-Champaign, Champaign, IL, USA

ARTICLE INFO

Article history:
Received 19 March 2022
Accepted 21 November 2022
Available online 20 December 2022

Keywords:
Thin-walled lattices
Structure–property relations
Johnson–Cook model
Neural networks

ABSTRACT

This paper investigates the structure–property relations of thin-walled lattices, characterized by their cross-sections and heights, under dynamic longitudinal compression. These relations elucidate the interactions of different geometric features of a design on mechanical response, including energy absorption. We proposed a combinatorial, key-based design system to generate different lattice designs and used the finite element method to simulate their response with the Johnson–Cook material model. Using an autoencoder, we encoded the cross-sectional images of the lattices into latent design feature vectors, which were supplied to the neural network model to generate predictions. The trained models can accurately predict lattice energy absorption curves in the key-based design system and can be extended to new designs outside of the system via transfer learning.

1. Introduction

Lattice-filled sandwich panels see increasingly widespread use as energy-absorbing structures in different engineering applications such as energy absorbers [1,2], sacrificial cladding [3,4], and armor plates [5,6]. Previous studies have established that the core design plays a significant role in the energy absorption capabilities of the sandwich panel [7]. The honeycomb [8] lattice structure, along with many of its variants, such as the square honeycomb [9], bio-inspired honeycombs [10], and hierarchical honeycombs [11], have been studied extensively in the literature as cores of sandwich panels. They are examples of thin-walled 2D extruded lattices, whose wall thickness is significantly smaller than their in-plane dimensions, and the unit cells can be described by their cross-sectional designs and heights. To generate optimized designs that maximize specific energy absorption, parameter optimizations have been performed on design variables such as thickness [12], unit cell size [13], wave periodicity, and amplitude (for some bio-inspired honeycombs) [10].

The aforementioned sizing optimization efforts and the construction of response surfaces for the cores [13,14] provide insights into the structure–property relations relating the lattice structure to key performance metrics like stress–strain curve and energy absorption during deformation for a limited set of honeycomb-like lattice core designs. It is of great research and industrial interest to further extend such structure–property relations to a diverse set of lattice core designs with different cross-sectional geometries, and in particular, to elucidate how different geometric features such as the addition of wavy unit-cell walls, hierarchical designs, auxetic designs, and fractal designs combine and interact to affect the specific energy absorption of the lattice core design. The exploration of structure–property relations inherently involves surveying many different lattice core designs. Most current thin-walled lattice core designs in the literature were generated based on engineering intuition, experiments, and/or bio-inspiration [15–17]. Other studies leveraged optimization-based methods like the hybrid cellular automata [18,19], ground structure approach [20], and the ant colony optimization method [21] to systematically generate new, optimized cross sections for lattices under longitudinal compression. Nonetheless, there is a lack of a systematic compilation of the mechanical response and energy absorption characteristics for these new designs and an attempt to reveal potential structure–property relations for various lattice cross-sectional designs.

Our current work aims to develop a systematic framework to generate distinct lattice cross-section designs for longitudinal compression. To explore the structure–property relations, the considered structures do not need to have optimized energy absorption characteristics. Thus, we chose to generate new lattice cross-section designs via a simple combinatorial framework. Combinatorial methods have been applied to generate trussed-based lattices

[*] Corresponding author.
E-mail address: ijasiuk@illinois.edu (I. Jasiuk).

https://doi.org/10.1016/j.compstruc.2022.106940
0045-7949/© 2022 Elsevier Ltd. All rights reserved.

[22–24], triply periodic minimal surface lattices [25,26], and thin-walled structures [27] to create new designs by combining different pre-selected geometric descriptors. It is a novel attempt to leverage a combinatorial framework to generate 2D extruded lattice core designs for longitudinal compression systematically. The second objective of this work is to approximate the underlying structure–property relations via a neural network (NN) model. NN models have seen increasing use in solid mechanics, such as to predict stress–strain curve and toughness of composites [28–32] and to predict properties of lattices and rapidly evaluate performance [23,33–36]. However, NN models have not been widely applied to approximate the structure–property relations of thin-walled lattice cores.

This paper is organized as follows: Section 2 presents an overview of the combinatorial generation framework, numerical simulation, preprocessing of the design images, and the architecture of the NN model. Section 3 presents the results, and Section 4 discusses the quality of the image preprocessing and NN predictions. Section 5 summarizes the outcomes and highlights possible future works.

2. Methods

2.1. A framework to create lattices using geometric descriptors

A diverse design space of lattice cross-sections is needed such that the subsequent NN can extract trends from various lattice architectures. Drawing inspiration from the truss descriptors used to construct truss-based lattices in Bastek et al. [23] and Zok et al. [37], we define a set of geometric descriptors for generating lattice cross-sections. In this work, we focus on 2D extruded lattices with constant cross-sections. Such lattices can be uniquely defined by their cross-sections and heights. We further require that the unit cell of the periodic lattice fits within a square bounding box. In this case, a 2D tessellation can be generated by repeating the unit cell along the X and Y directions. With this simplification, one can formulate three types of geometric descriptors:

- Vertex. The vertices of the unit cell bounding box can take on different styles: it can be a vertex or the vertex can be replaced by a straight edge or an arc, see Fig. 1.
- Edge segment. Each edge of the unit cell bounding box can be divided into different edge segments, and each can take on a different style: it can be a straight edge or two arcs (inspired by the bio-inspired honeycomb studied in the work of Ha et al. [10]), see Fig. 2.
- Interior support. Supports can be added to the interior of the unit cell: it can be no support, a +-shape, or an *X*-shape support. A circle can be added to the center of the unit cell, see Fig. 3.

It is important to point out that the lattice geometric descriptors are not unique; other choices are possible to generate a different design space. The descriptors and their sub-options chosen here aim to generate lattices that bear similar design cues to some traditional or bio-inspired lattices seen in the literature [10,38,39].

A unique lattice design can then be generated in a combinatorial manner by forming different combinations of geometric descriptors. Some geometric descriptors also allow for sub-design options. In the straight edge vertex case (Fig. 1b), the edge can take on either edge style depicted in Fig. 2. In the arc vertex case (Fig. 1c), the arc can point inward or outward to the unit cell. Vertical and horizontal edges of the unit cell bounding box can be divided into a different number of edge segments. Still, we limit to a minimum of 2 and a maximum of 4 divisions for all edges. The lattice design created by a combination of the design options and their sub-options can be conveniently denoted by an 8-digit key. Each digit represents a choice on each option/sub-option. A list of all possible options is presented in Table 1. This key-based design system contains 660 unique lattice design keys. Besides the discrete design variables, the framework also features a continuous design variable, namely the thickness of the lattice walls. In the analysis, we fixed the number of unit cells to be 4 in the cross-section, forming a 2-by-2 arrangement. Note that different numbers of unit cells and their arrangements can be used to construct the lattice structure and will affect the mechanical response of the lattice. Studying the effects of the number of unit cells and unit cell arrangement is a subject of our future works. A survey of some periodic lattices generated from the system is presented in Fig. 4.

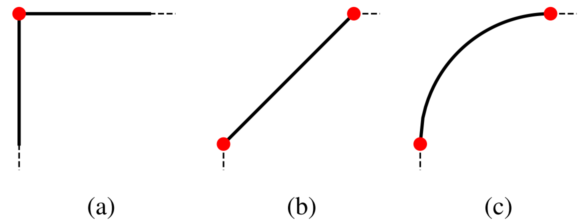


Fig. 1. (a) The vertex of the unit cell remains as-is. (b) The vertex is replaced by a straight edge. (c) The vertex is replaced by an arc.

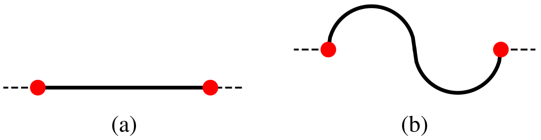


Fig. 2. (a) The edge segment of the unit cell remains as-is. (b) The edge segment is replaced by two arcs.

2.2. Finite element simulations

A Python script was developed to generate cross-sectional sketches in the finite element (FE) analysis package Abaqus [40] based on given design keys. The sketches were scaled so that the 2-by-2 lattice touches a square bounding box with a side length of 20 mm. Three-dimensional shell parts were created in Abaqus via extrusion to a fixed height of 10 mm. All lattice designs were discretized with 4-node shell elements with reduced integration (S4R) and uniform mesh sizes of 0.25 mm (in the X-Y plane) and 0.8 mm (along the height of the lattice structure) were used.

In this work, we considered the lattices to be made from Ti-6Al-4V. To accurately capture the response of Ti-6Al-4V during high strain-rate impact loading, which is common in lattice-filled sandwich panels applications, the Johnson–Cook plasticity and damage models developed by Johnson and Cook [41] were used in the Abaqus/Explicit dynamic simulations. The Johnson–Cook plasticity model defines a rate-dependent material hardening behavior and the damage model defines an equivalent plastic strain at which material damage initiates [40]. Beyond damage initiation, a damage evolution law was used in Abaqus to define how the material damage evolves until the eventual failure of the finite element, which necessitates the definition of fracture energy. The relevant material properties follow those found in the literature [42–44]. The element deletion technique [40] was used to model the material damage and failure, which removes elements that undergo significant damage from the FE simulation to prevent non-convergence due to excessive element distortion. As a simple approximation to high strain-rate impact loading, we sandwiched

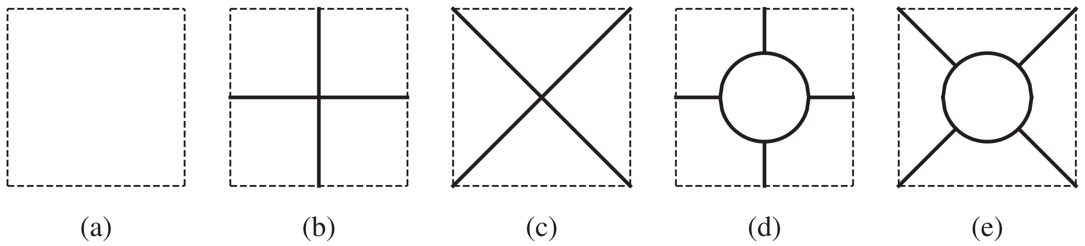
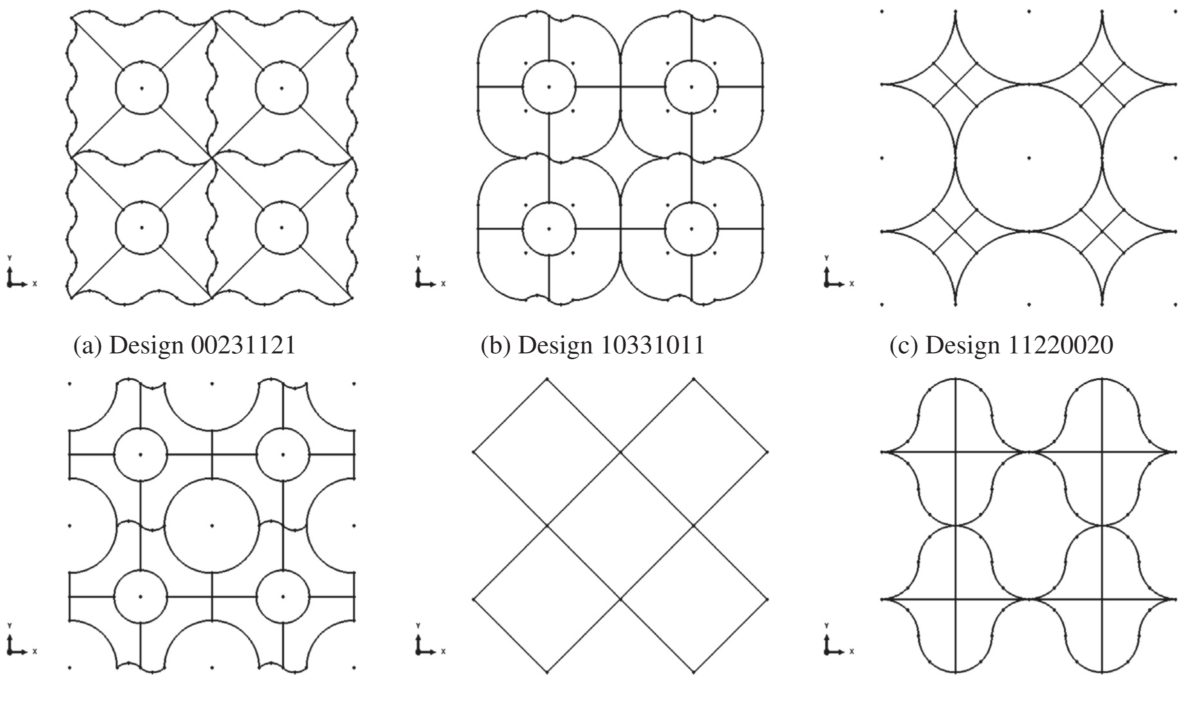


Fig. 3. (a) No interior support is added. (b) A +-shape support is added. (c) A *X*-shape support is added. (d) A +-shape support is added along with a circle. (e) A *X*-shape support is added along with a circle.

Table 1
Explanation of the 8-digit design key system.

Digit	1	2	3	4	5	6	7	8
Option	Vertex	Arc sub-opt	# segments horizontal	# segments vertical	Edge sub-opt horizontal	Edge sub-opt vertical	Interior	Interior sub-opt
Possible values	0 (2a)		2	2			0 (3a)	
	1 (1b)	0 (in)	3	3	0 (2a)	0 (2a)	1 (3b)	0 (3d)
	2 (1c)	1 (out)	4	4	1 (2b)	1 (2b)	2 (3c)	1 (3e)



(a) Design 00231121
(b) Design 10331011
(c) Design 11220020
(d) Design 11331011
(e) Design 20220000
(f) Design 21221110

Fig. 4. A survey of some lattice designs generated by the key-based system.

the lattice between two rigid plates, and the lattices were subjected to dynamic longitudinal compression. The bottom plate, where the reaction force was measured, was held fixed, and the top plate traveled downward with a constant velocity determined by the user-defined strain rate. All sidewalls were traction-free and were free to deform. All simulations shared a constant final displacement of 2 mm, corresponding to 20% nominal compressive strain along the Z-axis. Mass scaling was applied to the lattice to shorten simulation run time so that a minimum stable time increment of 2×10^{-8} s was achieved. It is important to note that the FE simulations were meant to provide an example data set for the neural network model training, and the simulated responses were not validated by lattice-level compression experiments. The reaction force (RF) at the bottom plate, displacement of the top plate, plastic dissipation (PD), damage dissipation (DMD), and elastic strain energy (ELSE) of the lattice structure were outputs of the FE simulations. The energy quantities (ELSE, PD, and DMD) were extracted by summing the corresponding element contributions for all finite elements in the model. The energy quantities were stored in Abaqus's history outputs for the simulations and were extracted by an in–house Python script. Fig. 5 depicts the FE model assembly and a typical deformed lattice at the end of dynamic compression.

A total of 15000 simulations were conducted on an Intel i7-11800H processor with 8 cores. Each simulation took about one to two minutes to complete. The design key, nominal strain rate,

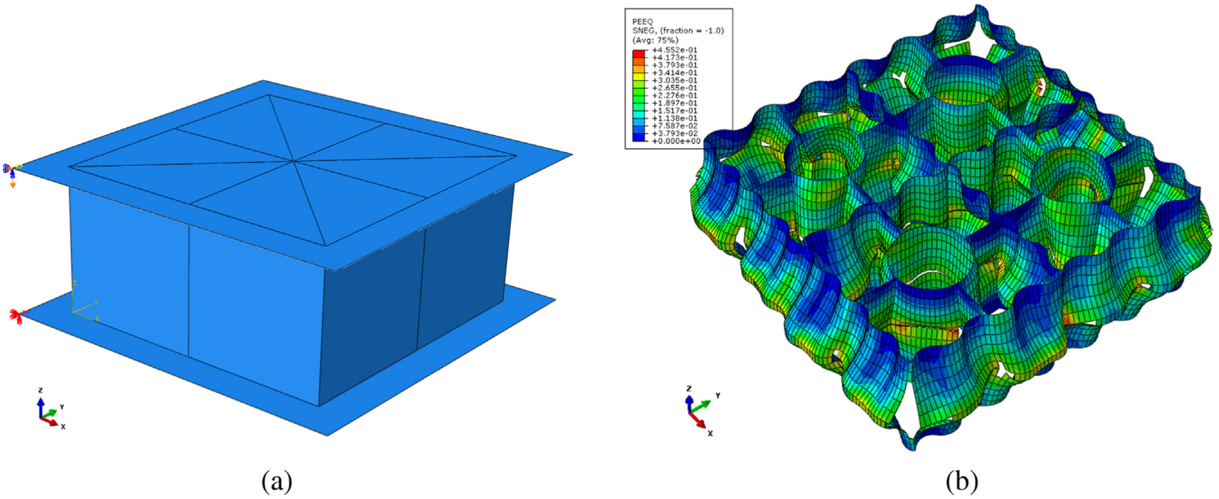


Fig. 5. FE model setup and results: (a) Lattice structure and two rigid plates set up for dynamic longitudinal compression. (b) A typical deformed lattice at the end of the simulation, colored by the equivalent plastic strain. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

and thickness of the shell elements were sampled using a random number generator. The nominal strain rates were uniformly sampled on the log scale from the range $[10^2, 10^5]$ s^{-1}, representing strain rates during blast loading [45]. Wall thicknesses of the lattice designs were uniformly sampled from the range [0.25,0.75] mm.

2.3. Encoding lattice cross-sectional images

As a general way to contain the geometry information in the lattice cross-section, we converted the 2D cross-sectional sketch into a 128-by-128 binary image revealing the skeleton of the cross-section. Note that the image did not include the shell thickness, so the resulting skeleton is always a single pixel wide. However, this means that the image matrices are highly sparse, and a naïve attempt to directly feed the image matrix to the NN as input is likely wasteful on computer memory.

In this work, we employed a simple autoencoder to extract information from the lattice cross-sectional image. The encoder consists of two dense layers, with each having 100 neurons with the rectified linear unit as an activation function. It transforms a 128-by-128 input image into a 100-by-1 latent feature vector. The decoder consists of a single dense layer with 128^2 neurons and uses the sigmoid function as activation. The autoencoder was developed and tested in Keras [46] with a TensorFlow [47] backend. To train and test the autoencoder, 600 design images were randomly chosen from a total of 660 designs, and an 80–20 split was adopted for training and testing. We used the mean squared error (MSE) as the loss function and the Adam optimizer [48] to minimize it. The autoencoder was trained for 80 epochs with a batch size of 50. To gauge the ability of the autoencoder to encode lattice structures never seen in a training set, an additional testing set consisting only of the 60 lattice designs not included in the training data set, was formed.

2.4. Neural network for sequence prediction

2.4.1. Input data, data augmentation, and loss function

A sampling of the input space was described in Section 2.2. The output arrays were extracted from Abaqus and were downsampled from 100 time steps down to 50-time steps to reduce size of the training data set. Two groups of inputs were generated for the NN model. The first group contains information on the lattice cross-sections in the form of 100-by-1 encoded latent feature vectors, which remain constant in time. The second group of inputs has six physics-informed temporal information arrays, which are:

1. Lattice shell thickness[1].
2. Final nominal compression strain[1].
3. $\log_{10}\dot{\bar{\epsilon}}$ [1]. The logarithm of the strain rate is used instead of the strain rate since the strain rate dependence in the Johnson–Cook model is logarithmic.
4. Nominal compression strain value at each output time point.
5. Current time value at each output time point.
6. A binary elastic wave indicator is defined as:

$$I_e = \begin{cases} 0, & t \leqslant t_e \\ 1, & t > t_e \end{cases}, \tag{1}$$

where $t_e = \frac{H}{\sqrt{E/\rho}}$ is an estimated time for the elastic stress wave to travel through the height of the lattice H. This indicates that the reaction force at the rigid bottom plate should remain 0 before the impact stress wave arrives.

Both groups of inputs were normalized by a standard scaler in Scikit-Learn [49] prior to training. The scaler was fitted to the training data points to avoid information leakage [28].

To increase the amount of training data, data augmentation was applied. For each FE simulation (compressed to a constant 20% strain), twelve final nominal strains in the range [5%,20%] were randomly sampled, and all inputs and outputs were linearly interpolated up to the selected final strain level. This effectively generated data points corresponding to the same strain rate but different final strain levels and increased the total number of input data points to 180000. These data points were divided into training (600 designs keys, approximately 61.8%), validation (10.9%), testing on lattice designs seen in the training set (denoted Test1, 18.2%), and testing on lattice designs unseen by the training set (denoted Test2, 60 design keys, approximately 9.1%).

We employed the mean absolute error (MAE) [50] as the loss function, defined as:

[1] All scalars are expressed in the form of a constant array with a length of 50, same as the number of time steps.

$$\mathrm{MAE} = \frac{\sum_{i=1}^{N} |\mathbf{Y_i} - \hat{\mathbf{Y}}_\mathbf{i}|}{N}, \tag{2}$$

where $N, \mathbf{Y_i}, \hat{\mathbf{Y}}_\mathbf{i}$ denote the number of training data points, ground-truth outputs, and the NN predictions, respectively. The MSE was chosen as a metric.

2.4.2. Neural network model

In solid mechanics, a recurrent NN model known as the gated recurrent unit (GRU) model has been widely used to predict sequences [51,52]. In the work of Abueidda et al. [51], three recurrent neural network models, including the GRU, long short-term memory (LSTM) model, and the temporal convolutional network (TCN) model, were used to predict the effective stress of an elasto-plastic material undergoing a randomly generated load path. They found that the GRU model yielded the most accurate predictions, but took the longest time to train compared to the other two models (see the comparison in Fig. 2 therein). This result highlights the ability of the GRU model to capture structure response under complex deformation and material constitutive models. For this reason, we utilized a GRU model to predict the energy absorption of lattices under dynamic compression. However, we did not compare the performance of the LSTM and TCN models using the current data set, as it is outside the scope of this current research.

The developed GRU model consists of three stacked layers, each of 300 GRU units with hyperbolic tangent (tanh) activation, leading to a model with 1.45 million trainable parameters. The architecture of the GRU model is depicted in the schematic in Fig. 6. The loss function was minimized using an Adam optimizer [48] with an inverse time decay learning rate schedule and an initial learning rate of 1×10^{-3}. The model was trained for 150 epochs with a batch size of 600, and training was repeated 10 times to obtain average training time and model accuracy. The complete data set was reshuffled and partitioned in each training repetition, as described in Section 2.4.1. The GRU model was implemented and tested in Keras [46] with a TensorFlow [47] backend. All training was conducted on an Intel i7-11800H processor with GPU acceleration on an Nvidia GeForce RTX 3050 GPU.

3. Results

3.1. Lattice design encoding and reconstruction

Using a latent space dimension of 100, we compared the original lattice designs to the reconstructed ones generated by the trained autoencoder for six randomly selected designs, see Fig. 7. To quantitatively describe the reconstruction quality, the dice similarity coefficient (DSC) [17] is used:

$$\mathrm{DSC} = \frac{2|\boldsymbol{I} \cap \boldsymbol{I}_r|}{|\boldsymbol{I}| + |\boldsymbol{I}_r|}, \tag{3}$$

where $\boldsymbol{I}$ and $\boldsymbol{I}_r$ are the ground-truth and reconstructed binary skeleton images (1 denotes presence of material and 0 indicates void), respectively.

3.2. Predicting energy outputs

To access the number of input data points required in training to obtain accurate results, different percentages of the total input data were used to train the model independently and were tested on two identical testing sets; the results are shown in Fig. 8a. To obtain a measure of the average performance of the GRU model, training was repeated ten times using the partition described in Section 2.4.1. A typical convergence curve showing the loss and metric during training is shown in Fig. 8b. As described in Section 2.4.1, we tested our model on two testing sets: (1) Test1, which was chosen from the 600 lattice designs that were seen by training, and (2) Test2, which contains 60 lattice designs that were completely unseen by training. Mean and standard deviation of training time, relative MAE for each component in each test set, are reported in Table 2. Relative MAE is presented here since the four output arrays considered have vastly different scales; thus, we normalize the MAE by the range over time steps of the respective ground truth[2].

A comparison between ground truths and NN predictions, ranked by the percentile of MAE for each output array, is presented in Figs. 9 and 10. The median model (one that gives the median overall MAE among the 10 training repetitions) was used to generate the plots. The 25th percentile plots for the damage dissipation in Figs. 9 and 10 depict a poor visual comparison despite having a low MAE. This apparent disagreement is due to the small damage dissipation magnitude in the FE ground truth, leading to a small absolute error compared to other cases, where the FE ground truths show damage dissipation that is two orders of magnitude larger.

Once trained, the GRU model can infer lattice mechanical response and energy absorption at a rate much faster than FE simulations. To test the inference speed, we randomly selected 50 shell thickness values. We used the model to infer the response of all 660 designs in the key-based design system under different shell thicknesses, which led to a total of 33000 evaluations. The inference finished in 6.15s, yielding an average inference speed of 1.866×10^{-4}s per design. In comparison, a FE simulation of the lattice design typically takes around one to two minutes to complete.

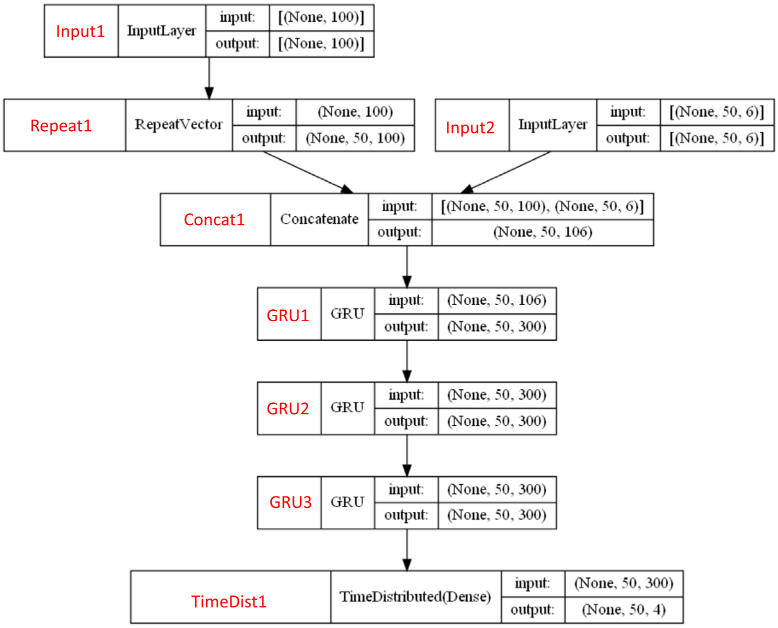
3.3. Extending to new lattice designs through transfer learning

Comparison of model performance between sets Test1 and Test2 shows how well the trained GRU model can generalize to previously unseen but similar designs (all were generated by the key-based system detailed in Section 2.1). It is of interest to further investigate the performance of the as-trained model on lattice designs that are generated outside of the key-based design system.

Three new geometries were generated manually, and the comparison of the lattice designs with their autoencoder reconstructions is shown in Fig. 11. The mean and standard deviation of relative MAE for each output array in each test set using the as-trained model are listed in Table 3. For easier comparison, results from Test2 in Table 2 are repeated here.

Besides the as-trained model, it is worth investigating how the model can be extended to unseen geometries by exposing it to a small amount of new input data via transfer learning. Transfer learning can be used to extend the trained model to a new field by transferring its knowledge learned previously from a similar field [53]. The trained GRU model was retrained for 20 epochs on a new training set of data points corresponding to new geometries. To prevent catastrophic forgetting during transfer learning, we add 5000 original training data points corresponding to geometries in the key-based design system to the training set, although other more involved methods exist [54,55]. To gauge the presence and amount of catastrophic forgetting, we test our data on two testing sets: one consists of 60 lattice design keys unseen during the training of both the original GRU model and the transferred model, while the other one contains data points for the three new geometries (that are unseen by the transfer training). Fifty strain rates

[2] To avoid erroneous relative MAE calculations when the range of the ground truth is very close to 0 (i.e., for cases where the stress wave has not arrived yet), a minimum range of 0.25 N is enforced for reaction force and 1×10^{-2} J for energies.


Input1 | InputLayer | input: [(None, 100)] | output: [(None, 100)]
Repeat1 | RepeatVector | input: (None, 100) | output: (None, 50, 100)
Input2 | InputLayer | input: [(None, 50, 6)] | output: [(None, 50, 6)]
Concat1 | Concatenate | input: [(None, 50, 100), (None, 50, 6)] | output: (None, 50, 106)
GRU1 | GRU | input: (None, 50, 106) | output: (None, 50, 300)
GRU2 | GRU | input: (None, 50, 300) | output: (None, 50, 300)
GRU3 | GRU | input: (None, 50, 300) | output: (None, 50, 300)
TimeDist1 | TimeDistributed(Dense) | input: (None, 50, 300) | output: (None, 50, 4)

Layer	Usage
Input1	Takes in latent image information vector
Input2	Takes in other input data (Sect 2.4.1.)
Repeat1	Repeats image by the number of time steps
Concat1	Concatenates two inputs together
GRU1	First GRU layer
GRU2	Second GRU layer
GRU3	Third GRU layer
TimeDist1	Generates predictions for all time steps

Fig. 6. Architecture of the developed GRU model.

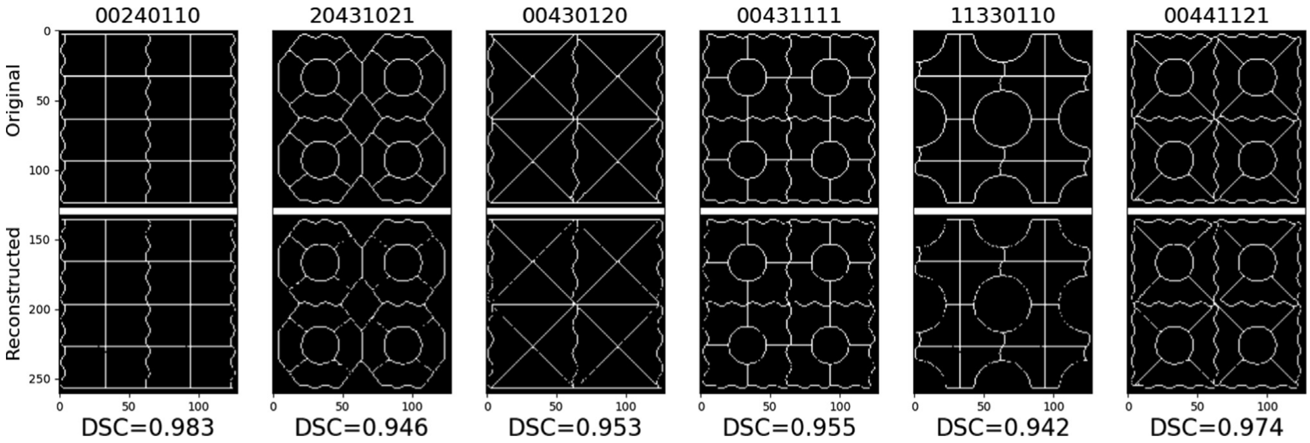


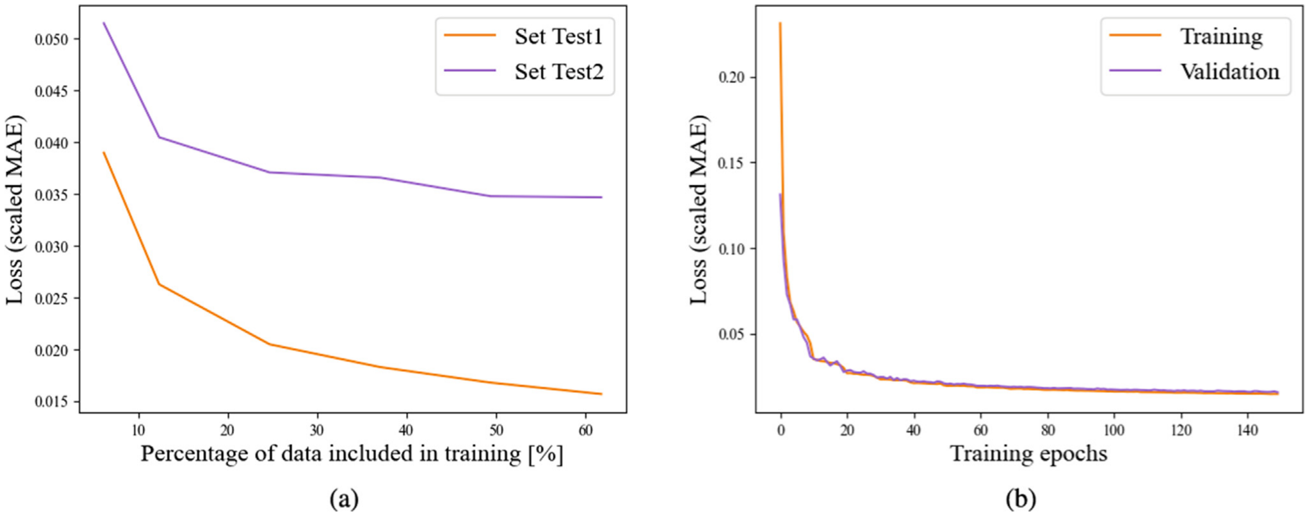
Fig. 7. Comparison of original and reconstructed lattice design skeletons for six randomly chosen designs.

and thicknesses were randomly sampled from the same ranges as described in Section 2.2 for each of the three new geometries, leading to 150 FE simulations. A similar data augmentation technique detailed in Section 2.4.1 was used to generate 7500 input data points. Plots comparing the relative MAE in both testing sets are shown in Fig. 12. A comparison between ground truths and transferred GRU model predictions, ranked by the percentile of MAE for each output array, is presented in Fig. 13. The model retrained with 40% of the new data points was used to generate the plot.

4. Discussion

Fig. 7 demonstrates the ground truth images and their autoencoder reconstructions. For six randomly chosen cases, we observe a close resemblance of the two, which indicates that a 100-by-1 latent vector is sufficient to encode the input images from within the key-based design system. This confirms the autoencoder as an effective tool to encode image inputs [56,57]. However, as Fig. 11 shows, the reconstruction quality drops significantly on images from outside of the key-based design system. Especially for the third new geometries shown in Fig. 11, its autoencoder reconstruction resembles the first geometry, although the ground truths are vastly different. This confusion in the autoencoder indicates that it is specialized in encoding images within the key-based system, which all bear some design feature resemblance due to the combinatoric nature of the framework. One possible way to enhance the encoding capability of the autoencoder is to expose it to a more diverse design space. As training for the autoencoder is separated from the training of the GRU model, this additional training can be done efficiently without the need for any expensive FE simulations. It is also hypothesized that the prediction accuracy of the subsequent GRU model will increase if the autoencoder effectively and accurately extracts important latent features from the given design.

Two validation sets, Test1 and Test2, were always employed for examining the model's ability to extend the structure–property relations learned from a limited input space to new design geometries. Fig. 8 shows the accuracy of the two sets when a different number of data points were used in training. Although loss keeps decreasing on set Test1, model performance plateaus on set Test2 after about 50% of points are used. For best performance in both



(a)

(b)

Fig. 8. Convergence plot for GRU model training process: (a) Scaled mean squared error when a different percentage of the total data is used in training. (b) Scaled mean absolute error evolution during training. Note that the MAE shown here is the MAE computed on the variables scaled by the standard scaler.

Table 2
Mean and standard deviation of training time and relative MAE.

	Train time [*s*]	rMAE,RF[1], [%]	rMAE, PD[2], [%]	rMAE,DMD[3], [%]	rMAE,ELSE[4], [%]
Test1	3597.44 (23.15)	6.26 (0.19)	0.30 (0.01)	4.15 (0.16)	0.95 (0.01)
Test2		9.32 (1.08)	1.04 (0.18)	6.93 (0.61)	2.38 (0.26)
Percent difference	\	48.75%	242.58%	67.09%	151.83%

[1] Reaction force.
[2] Plastic dissipation.
[3] Damage dissipation.
[4] Elastic strain energy.

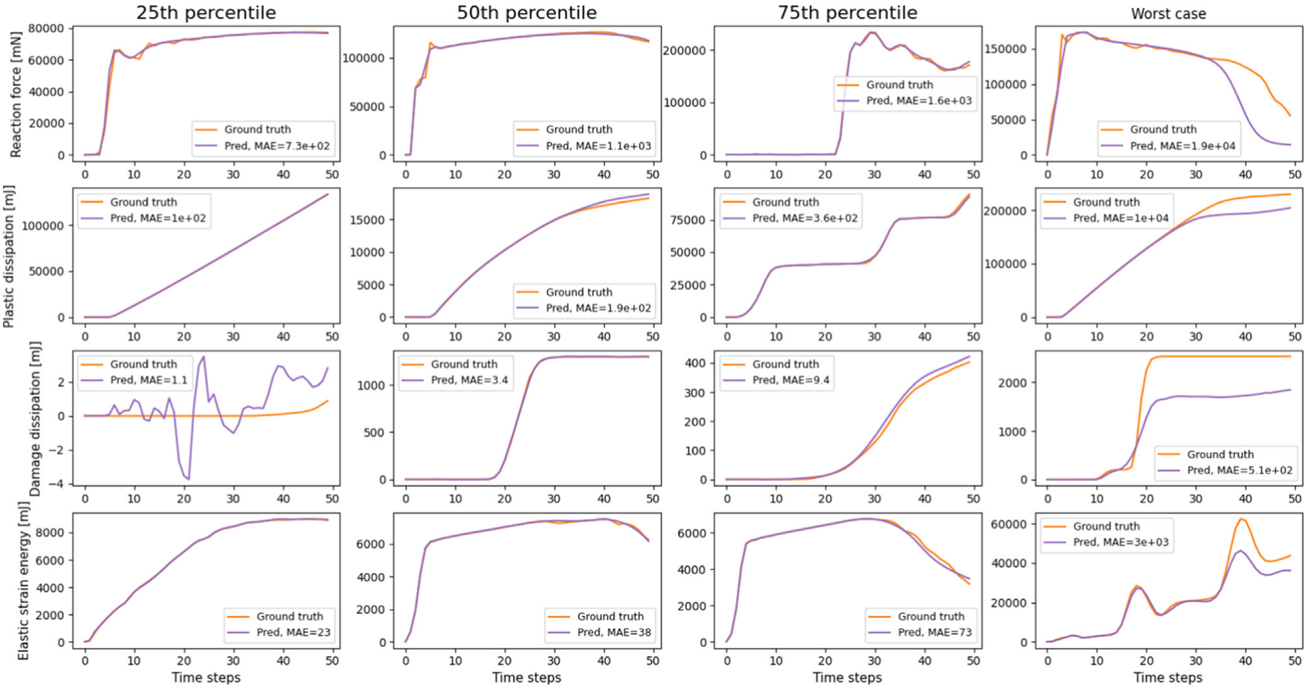


Fig. 9. Comparison of ground truths and GRU predictions on set Test1 (contains designs seen in training), ranked by percentile of MAE to provide a representative sampling. Note that the MAE is ranked independently for each of the four output arrays.

sets, all subsequent models in this work were trained using 61.8% of total data points, as described in Section 2.4.1. It can be inferred from Fig. 8b that no severe overfitting occurred during training.

Comparing the results in Table 2, we see that the prediction accuracy on set Test2 has deteriorated by about 50 to 240 percent compared to set Test1, but we note that all relative MAEs in Test2

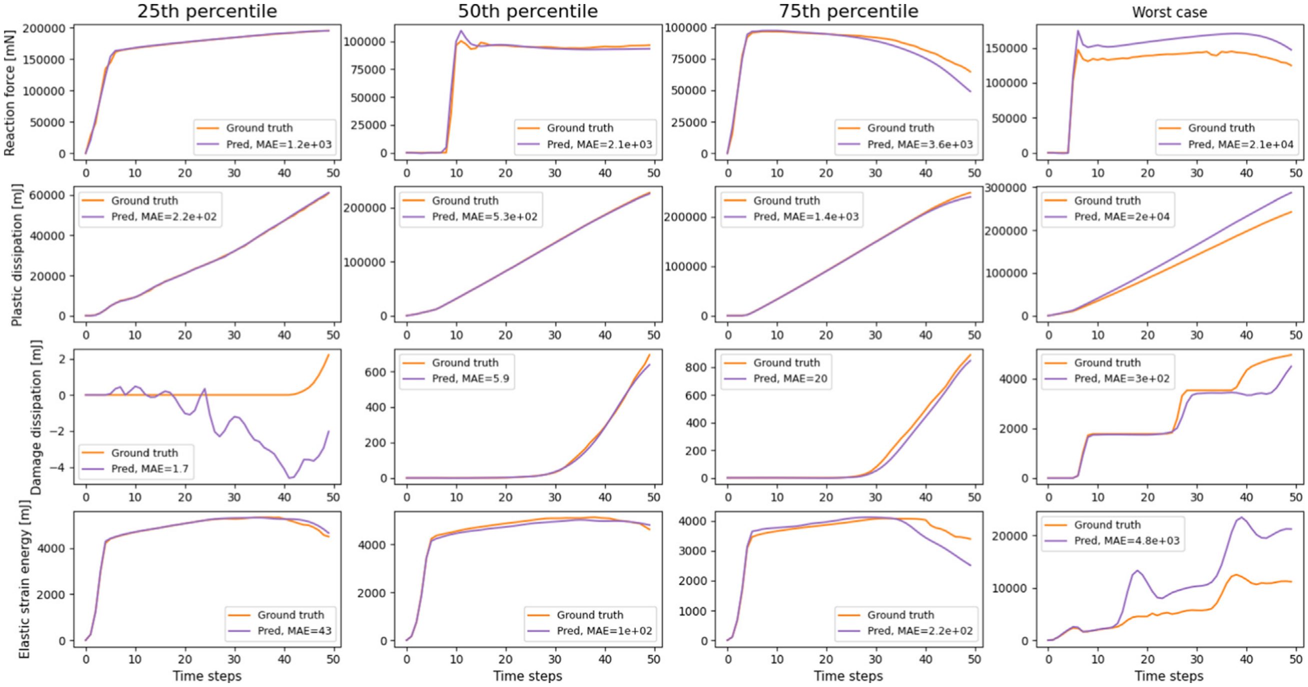


Fig. 10. Comparison of ground truths and GRU predictions on set Test2 (contains designs unseen by training), ranked by percentile of MAE to provide a representative sampling. Note that the MAE is ranked independently for each of the four output arrays.

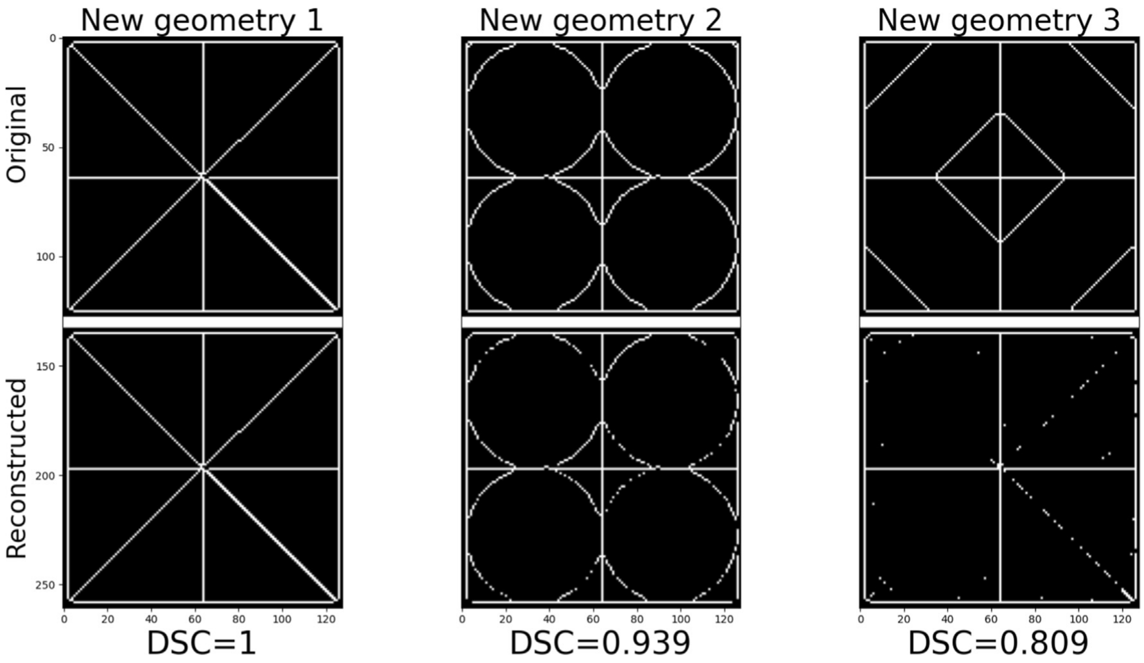


Fig. 11. Comparison of original and reconstructed lattice design skeletons for new lattice geometries.

remain smaller than 10% of the ground truth range. The statistical distributions of MAEs are shown in Fig. 9 and Fig. 10. Comparing the first three columns (up to 75th percentile), we observe that the GRU predictions closely follow the FE simulation results, except when the ground truth values are very small (e.g., row 3 column 1 in Fig. 9). Deviations from ground truth grow in the later stages of loading, after about 30 time steps. However, even in the highest MAE case, the GRU model can predict response curves that bear similar trends as the ground truth in both testing sets. It is worth noting that a positivity constraint was not enforced on the energy outputs; thus, some non-physical negative energy values arise when the ground truths are small. In most cases, the model can accurately predict the reaction curve and energy absorption history for various lattice designs, with an observed accuracy similar to that reported by Yang et al. [28], which employs a convolutional NN architecture as opposed to the recurrent NN architecture in this work. As is apparent in Table 3, the trained GRU model has a certain generalization ability on set Test2, but falls short on new geometries outside of the key-based design system. It is hypothesized that this limited generalization ability is due to both geome-

Table 3
Mean and standard deviation of relative MAE on new geometries.

	rMAE, RF[1], [%]	rMAE, PD[2], [%]	rMAE, DMD[3], [%]	rMAE, ELSE[4], [%]
Test2	9.32 (1.08)	1.04 (0.18)	6.93 (0.61)	2.38 (0.26)
New geometry 1	132.45 (70.19)	7.05 (0.86)	27.59 (5.38)	13.47 (1.03)
New geometry 2	338.09 (85.60)	26.35 (2.05)	38.95 (7.86)	35.82 (1.98)
New geometry 3	186.81 (63.80)	17.88 (1.62)	28.52 (4.99)	26.00 (1.30)

[1] Reaction force.
[2] Plastic dissipation.
[3] Damage dissipation.
[4] Elastic strain energy.

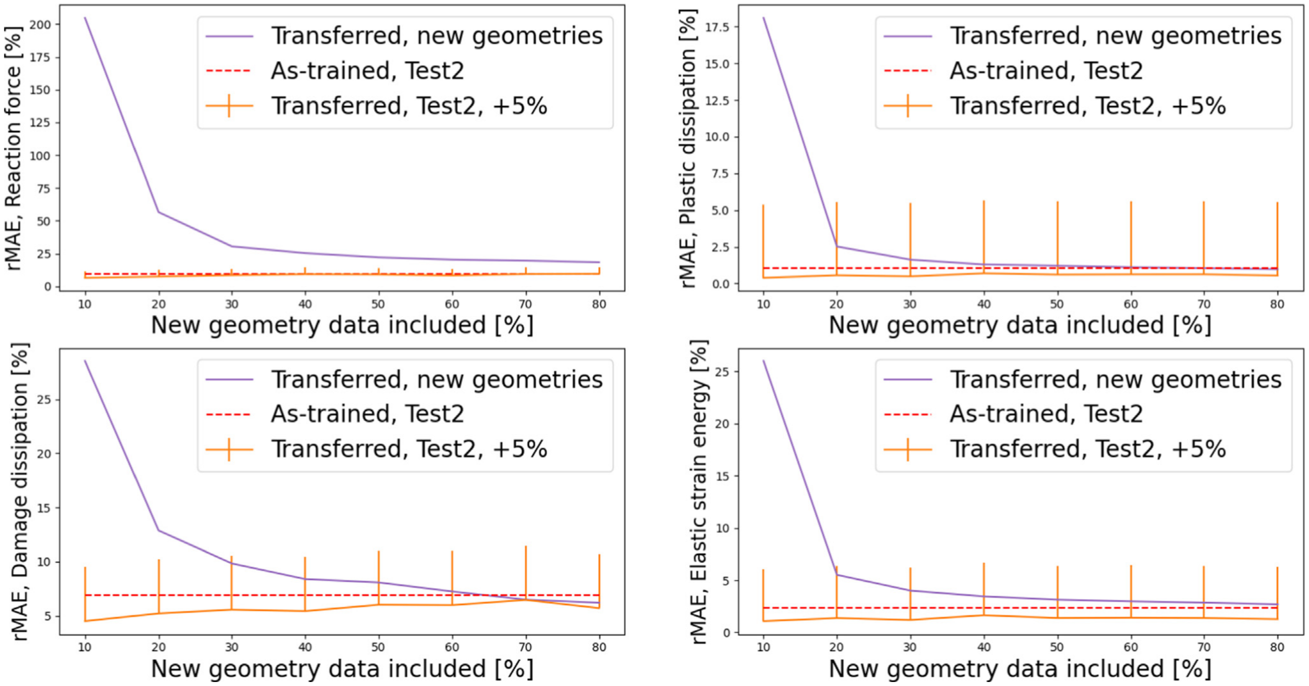


Fig. 12. Comparison of relative MAE on different output arrays using different percentages of the total data points corresponding to the new geometries. The relative MAE of the as-trained GRU model is plotted in red as a reference. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

tries in sets Test1 and Test2 coming from the same key-based design system and can be remedied by having a more diverse training set that covers a wider range of input designs. Previous studies also suggest alternative ways to improve the generalization ability of trained models, such as altering the input format [58] or using a neural network ensemble [59].

It is unlikely that the model needs to predict the mechanical response and energy absorption of a new lattice geometry given only its cross-section. It is often possible to obtain additional input data from limited testing. The model would then extend predictions to the new lattice design with different wall thicknesses and/or loading strain rates. Fig. 12 demonstrates the results of transfer learning with different percentages of new input data. We see that transfer learning based on the GRU model trained in Section 3.2 is highly effective. With as few as 20 training epochs using only 30% of the 150 FE simulations on the new lattice designs, the prediction accuracy on the energy absorption curves is within 5% of the model performance on the set Test2. Relative MAE of the reaction force prediction remains relatively high at 30.5% with the inclusion of 30% new data points, but it is still a dramatic decrease from over 300% in the as-trained model. Also, note the transferred model retains similar accuracy on set Test2 to the original GRU model, indicating that catastrophic forgetting has not occurred. Fig. 13 depicts the MAE distribution of the transferred model. Predictions up to the 75th percentile show relatively close agreement with FE simulation results, and large deviations only emerge after about halfway through the loading. This shows that transfer learning effectively extends a trained GRU model to new lattice designs outside of the original, finite training design space with limited new data.

5. Conclusions and future work

In this work, a GRU model was trained to predict the reaction force curve and energy absorption curves of thin-walled lattices during dynamic longitudinal compression. A novel combinatorial, key-based system was developed to generate lattice geometries that share some of the design cues with bio-inspired honeycomb lattices, which the authors believe is the first time that a combinatorial framework is used to systematically generate thin-walled lattice designs. An autoencoder was used to effectively encode the lattice cross-sectional image to reduce input dimensions. The GRU model is able to accurately predict the energy responses in unseen geometries from the key-based design system but falls short of directly extending the predicting capability to lattice designs outside of the key-based design system. Nonetheless, when enhanced with data points on the new geometries via transfer learning, the transferred GRU model reclaims much of the accu-

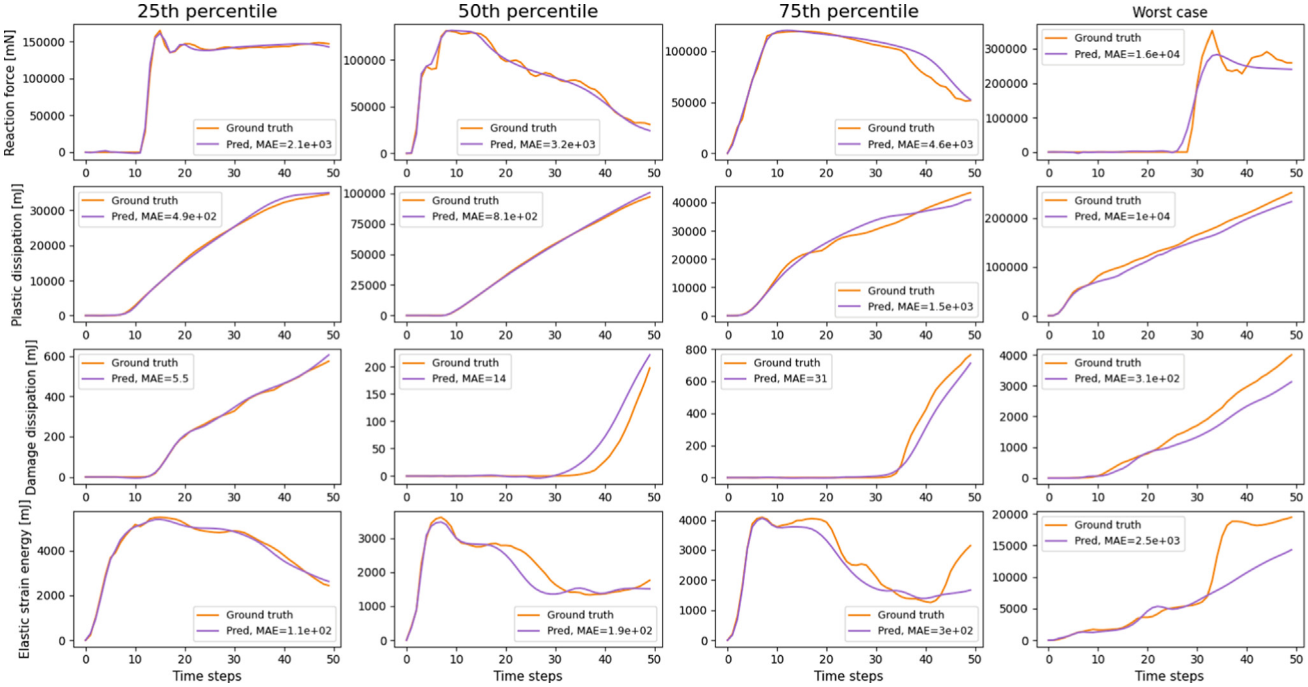


Fig. 13. Comparison of ground truths and GRU predictions on new geometries, ranked by percentile of MAE to provide a representative sampling. Note that the MAE is ranked independently for each of the four output arrays.

racy on new designs, while maintaining high accuracy over the designs in the key-based system.

We conclude that the trained GRU model can accurately approximate the relations between the lattice structure and key performance measurements such as stress–strain curve and energy absorption as a function of compression strain for a wide variety of thin-walled lattices under dynamic longitudinal compression. The ability of the trained model to quickly generate performance predictions for new lattice designs even on low-end laptop platforms renders itself a suitable guide in preliminary design stages to quickly sift out potentially performant designs for more detailed analyses. In this work, all repetitions of the unit cell in the 2-by-2 periodic arrangement share identical size and shape for simplicity. Additional unit-cell level irregularities can be introduced by distorting the unit cells and/or choosing a different unit cell in each of the 4 locations. However, that does not fundamentally alter the workflow of the current GRU framework. The autoencoder and GRU model can learn to encode and decode the input design images and to predict the mechanical response, irrespective of whether the lattice designs are regular or not, provided that a sufficient number of irregular lattices have been supplied to the training set.

In future work, we will focus on leveraging the gradients of the trained GRU model to inversely generate new designs with optimized specific energy absorption during high strain rate loading, similar to the work by Chen et al. [60] and Zheng et al. [61]. Different ways to parameterize the lattice design space, such as using the adjacency matrix [62], will be explored to enhance the generalizability and generative ability of future NN models. The effect of a number of unit cells and unit cell arrangements, as well as the role of periodic boundary conditions, will also be explored.

Data availability

The data and source code that support the findings of this study can be found at: https://github.com/Jasiuk-Research-Group/LatticeResponse_NN_Prediction.

CRediT author contributions

Junyan He: Conceptualization, Methodology, Software, Formal analysis, Investigation, Data Curation, Writing – Original Draft. **Shashank Kushwaha**: Methodology, Software, Formal analysis, Investigation, Writing – Original Draft. **Diab Abueidda**: Supervision, Writing – Review & Editing. **Iwona Jasiuk**: Supervision, Resources, Writing – Review & Editing, Funding Acquisition.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

We (I. J.) acknowledge the support of the Army Research Office contract (No. W 911NF-18–2-0067) and the National Science Foundation grant (MOMS-1926353).

References

[1] Shin Kwang Bok, Lee Jae Youl, Cho Se Hyun. An experimental study of low-velocity impact responses of sandwich panels for Korean low floor bus. Compos Struct 2008;84(3):228–40.
[2] Xie Suchao, Zhou Hui. Impact characteristics of a composite energy absorbing bearing structure for railway vehicles. Compos Part B: Eng 2014;67:455–63.
[3] Van Paepegem Wim, Palanivelu Sivakumar, Degrieck Joris, Vantomme John, Reymen Bruno, Kakogiannis Dimitrios, et al. Blast performance of a sacrificial cladding with composite tubes for protection of civil engineering structures. Compos Part B: Eng 2014;65:131–46.
[4] Codina Ramón, Ambrosini Daniel, de Borbón Fernanda. New sacrificial cladding system for the reduction of blast damage in reinforced concrete structures. Int J Protect Struct 2017;8(2):221–36.
[5] Qi Chang, Yang Shu, Yang Li-Jun, Wei Zhi-Yong, Lu Zhen-Hua. Blast resistance and multi-objective optimization of aluminum foam-cored sandwich panels. Compos Struct 2013;105:45–57.
[6] Gama Bazle A, Bogetti Travis A, Fink Bruce K, Yu Chin-Jye, Dennis Claar T, Eifert Harald H, et al. Aluminum foam integral armor: a new dimension in armor design. Compos Struct 2001;52(3–4):381–95.

[7] Tarlochan F, Hamouda AMS, Mahdi E, Sahari BB. Composite sandwich structures for crashworthiness applications. Proc Inst Mech Eng, Part L: J Mater: Des Appl 2007;221(2):121–30.

[8] Tarlochan Faris. Sandwich structures for energy absorption applications: A review. Materials 2021;14(16):4731.

[9] Xue Zhenyu, Hutchinson John W. Crush dynamics of square honeycomb sandwich cores. Int J Numer Meth Eng 2006;65(13):2221–45.

[10] Ha Ngoc San, Lu Guoxing, Xiang Xinmei. Energy absorption of a bio-inspired honeycomb sandwich panel. J Mater Sci 2019;54(8):6286–300.

[11] Qiao Jinxiu, Chen Changqing. In-plane crushing of a hierarchical honeycomb. Int J Solids Struct 2016;85:57–66.

[12] Paz J, Díaz J, Romera L, Costas M. Crushing analysis and multi-objective crashworthiness optimization of gfrp honeycomb-filled energy absorption devices. Finite Elem Anal Des 2014;91:30–9.

[13] Sun Guangyong, Li Guangyao, Stone Michael, Li Qing. A two-stage multi-fidelity optimization procedure for honeycomb-type cellular materials. Comput Mater Sci 2010;49(3):500–11.

[14] Panda Biranchi, Leite Marco, Biswal Bibhuti Bhusan, Niu Xiaodong, Garg Akhil. Experimental and numerical modelling of mechanical properties of 3d printed honeycomb structures. Measurement 2018;116:495–506.

[15] Christensen RM. Mechanics of cellular and other low-density materials. Int J Solids Struct 2000;37(1):93–104. https://doi.org/10.1016/S0020-7683(99)00080-3. ISSN 0020-7683, https://www.sciencedirect.com/science/article/pii/S0020768399000803.

[16] Gao Jie, Li Hao, Gao Liang, Xiao Mi. Topological shape optimization of 3D micro-structured materials using energy-based homogenization method. Ad. Eng Softw 2018;116:89–102.

[17] Abueidda Diab W, Koric Seid, Sobh Nahil A. Topology optimization of 2D structures with nonlinearities using deep learning. Comput Struct 2020;237:106283.

[18] Duddeck Fabian, Hunkeler Stephan, Lozano Pablo, Wehrle Erich, Zeng Duo. Topology optimization for crashworthiness of thin-walled structures under axial impact using hybrid cellular automata. Struct Multidiscip Optim 2016;54(3):415–28.

[19] Zeng Duo, Duddeck Fabian. Improved hybrid cellular automata for crashworthiness optimization of thin-walled structures. Struct Multidiscip Optim 2017;56(1):101–15.

[20] Guo Guikai, Zhao Yanfang, Weihe Su, Zuo Wenjie. Topology optimization of thin-walled cross section using moving morphable components approach. Struct Multidiscip Optim 2021;63(5):2159–76.

[21] Sharafi Pezhman, Teh Lip H, Hadi Muhammad NS. Shape optimization of thin-walled steel sections using graph theory and ACO algorithm. J Constr Steel Res 2014;101:331–41.

[22] Verma Chaman Singh, Rankouhi Behzad, Suresh Krishnan. A combinatorial approach for constructing lattice structures. J Mech Des 2020;142(4).

[23] Bastek Jan-Hendrik, Kumar Siddhant, Telgen Bastian, Glaesener Raphaël N, Kochmann Dennis M. Inverting the structure–property map of truss metamaterials by deep learning. Proc Natl Acad Sci 2022;119(1).

[24] Baykasoğlu Adil, Baykasoğlu Cengiz, Cetin Erhan. Multi-objective crashworthiness optimization of lattice structure filled thin-walled tubes. Thin-Walled Struct 2020;149:106630.

[25] Wang Jun, Callanan Jesse, Ogunbodede Oladapo, Rai Rahul. Hierarchical combinatorial design and optimization of non-periodic metamaterial structures. Additive Manuf 2021;37:101710.

[26] Callanan Jesse, Ogunbodede Oladapo, Dhameliya Maulikkumar, Wang Jun, Rai Rahul. Hierarchical combinatorial design and optimization of quasi-periodic metamaterial structures. International design engineering technical conferences and computers and information in engineering conference, 51760. American Society of Mechanical Engineers; 2018. page V02BT03A011.

[27] Wang ZhiJin, HuaZhi Zhou VI, Zhou HuaZhi, Shabalov VI, Shabalov AV. Six-ray folded configurations as the geometric basis of thin-walled elements in engineering structures. Thin-Walled Struct 2018;130:435–48.

[28] Yang Charles, Kim Youngsoo, Ryu Seunghwa, Gu Grace X. Prediction of composite microstructure stress-strain curves using convolutional neural networks. Mater Des 2020;189:108509.

[29] Gu Grace X, Chen Chun-Teh, Buehler Markus J. De novo composite design based on machine learning algorithm. Extreme Mech Lett 2018;18:19–28.

[30] Chen Chun-Teh, Gu Grace X. Machine learning for composite materials. MRS Commun 2019;9(2):556–66.

[31] Yang Charles, Kim Youngsoo, Ryu Seunghwa, Gu Grace X. Using convolutional neural networks to predict composite properties beyond the elastic limit. MRS Commun 2019;9(2):609–17.

[32] Abueidda Diab W, Almasri Mohammad, Ammourah Rami, Ravaioli Umberto, Jasiuk Iwona M, Sobh Nahil A. Prediction and optimization of mechanical properties of composites using convolutional neural networks. Compos Struct 2019;227:111264. https://doi.org/10.1016/j.compstruct.2019.111264. ISSN 0263-8223, https://www.sciencedirect.com/science/article/pii/S0263822319312383.

[33] Laban Othman, Gowid Samer, Mahdi Elsadig. Experimental investigation and uncertainty prediction of the load-carrying capacity of composite double hat for lattice core sandwich panels using artificial neural network. In: 2020 IEEE International Conference on Informatics, IoT, and Enabling Technologies (ICIoT). IEEE; 2020. p. 67–72.

[34] Messner Mark C. Convolutional neural network surrogate models for the mechanical properties of periodic structures. J. Mech. Des. 2019;142(2):024503. https://doi.org/10.1115/1.4045040. ISSN 1050-0472.

[35] Garland Anthony P, White Benjamin C, Jared Bradley H, Heiden Michael, Donahue Emily, Boyce Brad L. Deep convolutional neural networks as a rapid screening tool for complex additively manufactured structures. Additive Manuf 2020;35:101217.

[36] Hassanin Hany, Alkendi Yusra, Elsayed Mahmoud, Essa Khamis, Zweiri Yahya. Controlling the properties of additively manufactured cellular structures using machine learning approaches. Adv Eng Mater 2020;22(3):1901338.

[37] Zok Frank W, Latture Ryan M, Begley Matthew R. Periodic truss structures. J Mech Phys Solids 2016;96:184–203.

[38] Yang Xianfeng, Sun Yuxin, Yang Jialing, Pan Qifan. Out-of-plane crashworthiness analysis of bio-inspired aluminum honeycomb patterned with horseshoe mesostructure. Thin-Walled Struct 2018;125:1–11.

[39] San Ha Ngoc, Lu Guoxing. A review of recent research on bio-inspired structures and materials for energy absorption applications. Compos Part B: Eng 2020;181:107496.

[40] SIMULIA. Abaqus; 2020.

[41] Johnson Gordon R, Cook William H. Fracture characteristics of three metals subjected to various strains, strain rates, temperatures and pressures. Eng Fract Mech 1985;21(1):31–48.

[42] Kay Gregory. Failure modeling of titanium 6AI-4V and aluminum 2024–T3 with the Johnson-Cook material model. Office of Aviation Research, Federal Aviation Administration; 2003.

[43] Wang Yu-Fei, Yang Zhen-Guo. Finite element model of erosive wear on ductile and brittle materials. Wear 2008;265(5–6):871–8.

[44] D Leseur. Experimental investigations of material models for Ti-6A1-4V and 2024-T3. https://doi.org/10.2172/11977. https://www.osti.gov/biblio/11977.

[45] Othman Hesham, Marzouk H. Strain rate sensitivity of fiber-reinforced cementitious composites. Am Concr Inst Mater J 2016;113(2):143–50.

[46] Francois Chollet et al. Keras, 2015. https://github.com/fchollet/keras.

[47] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, et al. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. https://www.tensorflow.org/. Software available from tensorflow.org.

[48] Kingma Diederik P, Ba Jimmy. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980; 2014.

[49] Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, et al. Scikit-learn: Machine learning in Python. J Machine Learn Res 2011;12:2825–30.

[50] Willmott Cort J, Matsuura Kenji. Advantages of the mean absolute error (mae) over the root mean square error (rmse) in assessing average model performance. Climate Res 2005;30(1):79–82.

[51] Abueidda Diab W, Koric Seid, Sobh Nahil A, Sehitoglu Huseyin. Deep learning for plasticity and thermo-viscoplasticity. Int J Plast 2021;136:102852.

[52] Frankel Ari L, Jones Reese E, Alleman Coleman, Templeton Jeremy A. Predicting the mechanical response of oligocrystals with deep learning. Comput Mater Sci 2019;169:109099.

[53] Weiss Karl, Khoshgoftaar Taghi M, Wang DingDing. A survey of transfer learning. J Big Data 2016;3(1):1–40.

[54] Lee Sang-Woo, Kim Jin-Hwa, Jun Jaehyun, Ha Jung-Woo, Zhang Byoung-Tak. Overcoming catastrophic forgetting by incremental moment matching. Adv Neural Informat Process Syst 2017;30.

[55] Chen Xinyang, Wang Sinan, Fu Bo, Long Mingsheng, Wang Jianmin. Catastrophic forgetting meets negative transfer: Batch spectral shrinkage for safe transfer learning. Adv Neural Informat Process Syst 2019;32.

[56] Kim Yongju, Park Hyung Keun, Jung Jaimyun, Asghari-Rad Peyman, Lee Seungchul, Kim Jin You, et al. Exploration of optimal microstructure and mechanical properties in continuous microstructure space using a variational autoencoder. Mater Des 2021;202:109544. https://doi.org/10.1016/j.matdes.2021.109544. ISSN 0264-1275, https://www.sciencedirect.com/science/article/pii/S0264127521000976.

[57] Jung Jaimyun, Yoon Jae Ik, Park Hyung Keun, Jo Hyeontae, Kim Hyoung Seop. Microstructure design using machine learning generated low dimensional and continuous design space. Materialia 2020;11:100690. https://doi.org/10.1016/j.mtla.2020.100690. ISSN 2589-1529. https://www.sciencedirect.com/science/article/pii/S2589152920301071.

[58] Yiquan Zhang, Bo Peng, Xiaoyi Zhou, Cheng Xiang, Dalei Wang. A deep Convolutional Neural Network for topology optimization with strong generalization ability. arXiv e-prints, art. arXiv:1901.07761, January 2019.

[59] Yang Jing, Zeng Xiaoqin, Zhong Shuiming, Wu Shengli. Effective neural network ensemble approach for improving generalization performance. IEEE Trans Neural Networks Learn Syst 2013;24(6):878–87. https://doi.org/10.1109/TNNLS.2013.2246578.

[60] Chen Chun-Teh, Gu Grace X. Generative deep neural networks for inverse materials design using backpropagation and active learning. Adv Sci 2020.;7(5):1902607.

[61] Zheng Li, Kumar Siddhant, Kochmann Dennis M. Data-driven topology optimization of spinodoid metamaterials with seamlessly tunable anisotropy. Comput Methods Appl Mech Eng 2021;383:113894. https://doi.org/10.1016/j.cma.2021.113894. ISSN 0045-7825, https://www.sciencedirect.com/science/article/pii/S0045782521002310.

[62] Li Shunning, Liu Yuanji, Chen Dong, Jiang Yi, Nie Zhiwei, Pan Feng. Encoding the atomic structure for machine learning in materials science. Wiley Interdiscip Rev: Comput Mol Sci 2022;12(1):e1558.