**Proceedings of the ASME 2022 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference**
**IDETC/CIE 2022**
**August 14-17, 2022, St. Louis, USA**

**DETC2022/CIE-91209**

# SCALAR FIELD PREDICTION ON TOPOLOGICALLY-VARYING GRAPHS USING SPECTRAL SHAPE ENCODING

**Kevin Ferguson**
Visual Design and Engineering Lab
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

**James Hardin, Andrew Gillman**
Air Force Research Lab
WPAFB, Ohio 45433

**Levent Burak Kara**[*]
Visual Design and Engineering Lab
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

## ABSTRACT

*Scalar fields, such as stress or temperature fields, are often calculated in shape optimization and design problems in engineering. For complex problems where shapes have varying topology and cannot be parametrized, data-driven scalar field prediction can be faster than traditional finite-element methods. However, current data-driven techniques to predict scalar fields are limited to a fixed grid domain, instead of arbitrary graph/mesh structures. In this work, we propose a method to predict scalar fields on meshes of arbitrary refinement and topology. It uses features that capture shape geometry on a local and global scale as input to a multilayer perceptron to predict solutions to partial differential equations on graph data structures. The proposed set of global features is a vector that concisely represents the entire mesh as a spectral shape encoding. The model is trained on finite-element von Mises stress fields, and once trained it can estimate stress values at each node on any input mesh. Two shape datasets are investigated, and the model demonstrates decent performance on both, with a median R-squared value of 0.75. We also demonstrate the model's performance on a temperature field in a conduction problem, where its predictions have a median R-squared value of 0.98. By predicting from a simple, yet rich, set of mesh features, our method provides a potential flexible alternative to finite-element simulation in engineering design contexts. Code and datasets are available at:* `https://github.com/kevinferg/spectral-shape-encoding`.

[*]Address all correspondence to lkara@cmu.edu

## INTRODUCTION

Geometry-based optimization and design problems are prevalent in engineering. Engineers frequently must design the shape of a part, then perform an analysis on the part to determine what aspects of the design should be modified. This iterative process is well-established [1], but it is often inhibited by the slow speed of the analysis stage, which requires setting up a finite-element simulation, assigning all of the necessary material properties, boundary conditions, loads, and other parameters, and waiting for a finite element software to mesh the geometry, assemble stiffness/load matrices, and solve for the requested values. Techniques to streamline this process are often sought-after in engineering design settings [2, 3].

One tool that would expedite the design process is a fast way to locate weaknesses in candidate designs, as this would let an engineer save expensive computer analyses for a more finalized design. Data-driven scalar field prediction methods solve this problem by predicting desired quantities at every point in a particular domain. In the case of shape design, for example, this may take the form of predicting a failure probability or equivalent stress at every node in the part's mesh. The task is not trivial, as small geometric changes can result in drastic field changes. However, by training across many examples, data-driven methods can learn physical phenomena, forming surrogate models that can predict finite-element results [4, 5, 6, 7, 8].

Nie et al. [6] and Jiang et al. [7] propose deep-learning methods that solve for stress fields under varying geometries, boundary conditions, and loads. However, these approaches are image-based, so all inputs, including geometry, are encoded as binary image representations. This approach lacks the flexibility of a mesh or graph data structure, significantly limiting the method's application, as the resolution of every shape is fixed to a single grid of pixels.

Qi et al. [9, 10] present methods for point cloud classification and segmentation. Point cloud segmentation is similar to scalar field prediction, and the method has been used by Kashefi et al. [11] for the same purpose. One requirement of this strategy is that each point cloud must have the same number of points – a more desirable solution for scalar field prediction can take any mesh as input, whether or not it shares topology or number of points with any other mesh. Mesh segmentation techniques have shown good results on segmenting 3-D datasets [12, 13], providing evidence that predicting scalar fields on structures should be a feasible task.

A popular approach to solve similar problems is to use a graph neural network (GNN) to make predictions on nodes of a graph [14, 15]. These often make use of "message-passing" by iteratively aggregating graph neighborhood information at each node along graph edges, to learn representations of local graph structure – these are then used for predicting node embeddings (such as scalar fields). GNNs for physics predictions are used more often for computing updates of dynamic simulations than for predicting static scalar values [16]. More recently, Meyer et al. [17] perform direct-time GNN predictions for Computational Fluid Dynamics (CFD), which is more comparable to the scalar field prediction we wish to investigate. GNNs are powerful in that input graph topology can be arbitrary – however, with a large graph or fine mesh, the depth of these networks needs to be very high, leading to substantial computational expenses [18,19]. Furthermore, GNNs tend to over-smooth solutions, and therefore may be less apt for predicting fields with steep spatial gradients [20].

Some other data-driven scalar field prediction methods combine finite element methods with machine learning methods [21,22], but these models requires performing a modified finite-element simulation as part of the prediction, when a pure surrogate model is preferred.

In this paper we also explore dimensionality reduction for 2-D geometries. Lower-dimensional representations of 3-D surface meshes are commonly used in computer graphics applications [23, 24]. Guo et al. [25] represent 2-D designs in a latent space using a Variational Autoencoder (VAE) for topology optimization. Liang et al. [26] use Principal Component Analysis (PCA) to represent human aorta meshes as vectors to predict stress fields. These representations are learned across a dataset; the representation we propose can be computed more generally for any shape.

In this work, a model that predicts the Von Mises stress field in a static structural problem for a mesh of arbitrary topology by training on the results of many finite-element simulations is described. We propose a set of features that can capture local geometry at each node; next we define a unique method, spectral shape encoding, for representing an arbitrary geometry on a domain as a vector of fixed length. The model takes both local and global features, as well as nodal coordinates and a signed distance field, to make a prediction of stress at each node using a multilayer perceptron.

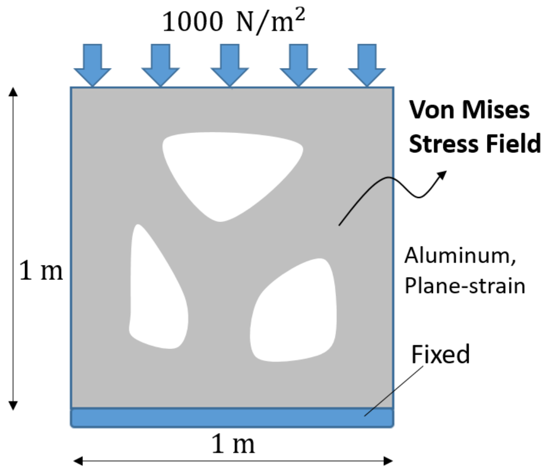


**FIGURE 1**. THE 2-D PROBLEM OF INTEREST: COMPRESSION OF ALUMINUM PART WITH VARIABLE INTERNAL GEOMETRY

We demonstrate the model's performance on two large shape datasets. The $R^2$ values of predictions for meshes in the training set, as well as for previously unseen meshes, are shown and the predicted fields are visualized. We then validate the method on a heat transfer problem and investigate possible weaknesses in the model.

The paper's main contributions are:

1. A public dataset of 2-D shapes with complex internal geometries, which cannot be easily defined by a small set of parameters
2. A unique dimensionality reduction technique for 2-D geometries using spectral decomposition of a signed distance field
3. A method for making node-wise scalar field predictions on meshes of any resolution using a set of local features and global features

## TECHNICAL APPROACH

### Problem

As a test problem, we investigate at a 2-D plane-strain scenario where a distributed load is applied to the top of an aluminum square domain that has a fixed bottom boundary, in which the goal is to predict von Mises stress. The details and dimensions of this problem are depicted in Fig.1. The material and dimensions chosen are arbitrary; we assume that predicting the von Mises stress field for this 2-D problem will yield results of similar quality to other physical fields without loss of generality. Stress fields exhibit steeper spatial gradients than, for example, temperature fields, and should therefore be more challenging to predict. (To back up this claim, we will also show our model's performance on a heat transfer problem in the Results section.) Additionally, the von Mises stress field has immediate practical relevance to engineering design problems, as comparing von Mises stress with material strength is commonly used to determine whether yielding will occur [27]. Note that the geometry on the interior of the square domain varies, as predictions will be made on different meshes.

## Datasets

Scalar field prediction models were trained for two 2-D shape datasets, which will be referred to as the "Voronoi Set" and the "Lattice Set." Figure 2 depicts an example mesh from each, along with their finite-element von Mises stress field solutions. The datasets were designed such that interior geometries would have varying topology and drastically different qualitative features. In an engineering design setting, freedom to laterally explore large design spaces like those in these datasets is desirable. For both datasets, a small set of parameters is not sufficient to fully describe the variation across meshes. Furthermore, the scalar fields that can exist on these complex geometries are not trivial to predict, making these datasets a good testbed for examining the strength of the proposed features. There is not a one-to-one correspondence between any pair of meshes in either dataset – the graph structures can be fully arbitrary.

Kou and Tan [28] describe a method for generating porous structures by computing B-spline curves whose control points are vertices of Voronoi cells in a 2-D domain, and then randomly merging these curves to become pores. The Voronoi Set is inspired by this method; first, a set of points are randomly sampled from a square domain and Voronoi cell boundaries are computed. A buffer around each cell boundary is generated in order to control the wall thickness between the pores, after which Laplacian smoothing [29] can be performed to round the corners. The method is illustrated in Fig.3. In addition to the random point coordinates, three parameters control this dataset: the number of holes, wall thickness, and degree of smoothing. In the Voronoi Set, the parameter ranges are: 3-4 holes, wall thickness from 0.10-0.18, and smoothing degree from 1-20.

The Lattice Set has a more straightforward creation process. Points are randomly selected from a square lattice, and polygonal pores are generated at these points. The polygons have between 3 and 6 sides. Here, the three parameters are number of holes, lattice grid size, and degree of smoothing; all other factors – hole placement, size, type, and orientation – are random. In the Lattice Set, parameter ranges are: 4-8 holes, lattice size from $3 \times 3$ to $4 \times 4$, and smoothing degree from 1-15.

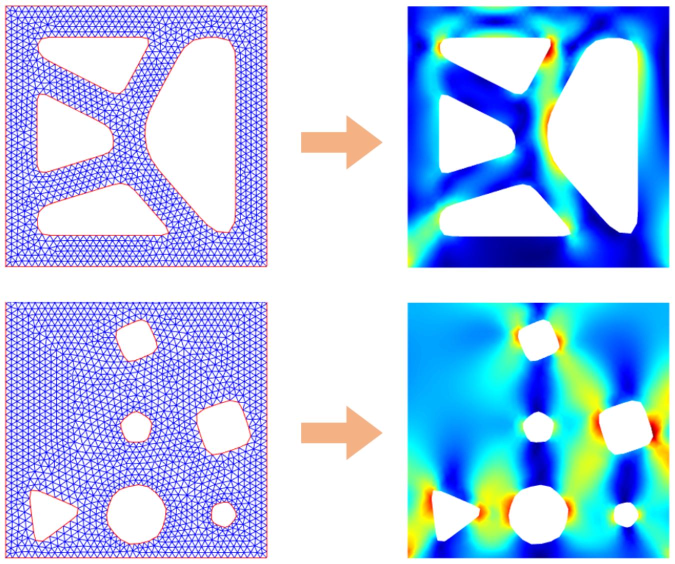


**FIGURE 2**. EXAMPLES OF MESHES AND STRESS FIELDS IN THE VORONOI SET (TOP) AND LATTICE SET (BOTTOM)

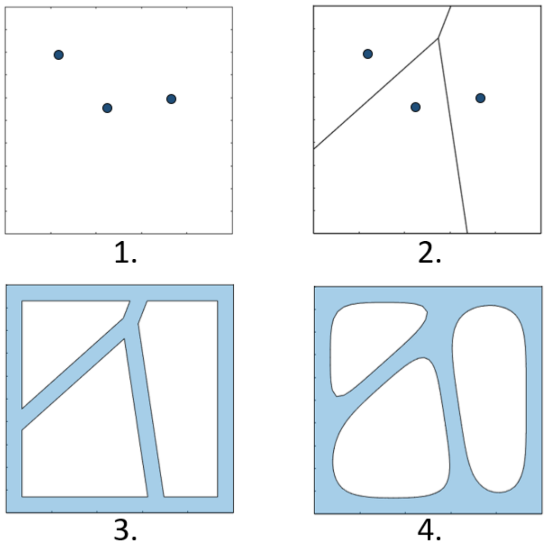
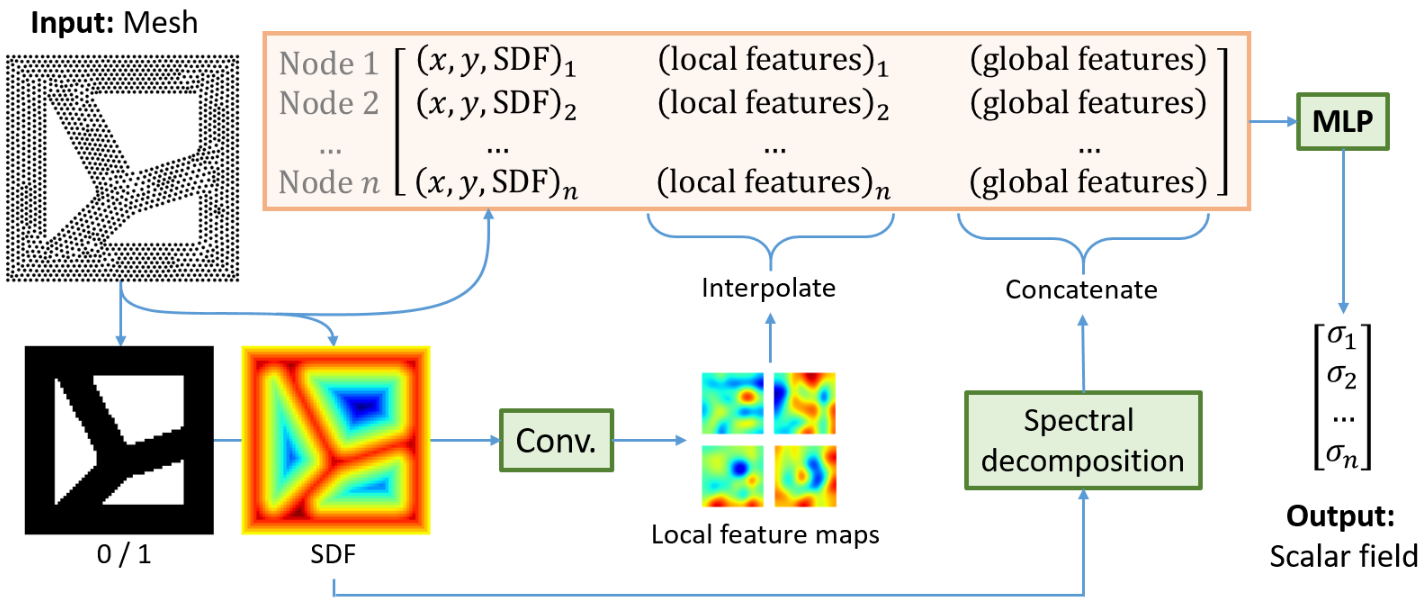


**FIGURE 3**. GENERATING THE VORONOI SET: 1. RANDOM POINT SELECTION; 2. VORONOI TESSELLATION; 3. CREATING PORES; 4. SMOOTHING

Each dataset is split into three groups: training, testing, and out-of-sample. We train each model on its respective training set, and test the models on the testing sets to examine overfitting and generalization beyond the training set. The out-of-sample sets consist of geometries where one parameter has values outside of the range it had in the training/testing sets. For the Voronoi



**Input:** Mesh
Node 1 $(x, y, \text{SDF})_1$ $(\text{local features})_1$ (global features)
Node 2 $(x, y, \text{SDF})_2$ $(\text{local features})_2$ (global features)
...
Node $n$ $(x, y, \text{SDF})_n$ $(\text{local features})_n$ (global features)
MLP
Interpolate
Concatenate
Conv.
Spectral decomposition
Local feature maps
0 / 1
SDF
$\sigma_1$ $\sigma_2$ ... $\sigma_n$
**Output:** Scalar field

**FIGURE 4**. THE PROPOSED MODEL, AN MLP THAT PERFORMS A SCALAR PREDICTION AT EACH MESH NODE, USING THE NODE COORDINATES, A SIGNED DISTANCE FIELD, A SET OF LOCAL FEATURES, AND A SET OF GLOBAL FEATURES

Set, this parameter was wall thickness, and for the Lattice Set, this parameter was hole count. Testing each model on out-of-sample meshes will help determine how significantly the proposed method is able to extrapolate beyond the geometries seen during training. The Voronoi Set contains 800 training, 200 testing, and 200 out-of-sample meshes, while the Lattice Set has 800, 200, and 160 respectively. We train one model on each dataset, and another model on the Combined Set, which contains all Lattice and Voronoi geometries.

Finite-element solutions were computed using MATLAB's PDE toolbox [30] to solve for Von Mises stress for the posed problem on each geometry.

## Model

To perform a scalar field prediction for the problem posed in Fig.1, the model must take in a mesh as input and output a set of node-wise predictions. In particular, we propose a multi-layer perceptron (MLP) that is evaluated on each node of the mesh separately. As input, the MLP takes in the coordinates of the given node, its signed distance field value, a set of local features describing the geometry near the node, and a set of global features describing the complete geometry. The local features are convolutional feature maps interpolated to each node. The global features, a "spectral shape encoding" are a set of coefficients representing the geometry. Figure 4 gives an illustration of the full predictive model; details on local and global features are in the sections to follow.

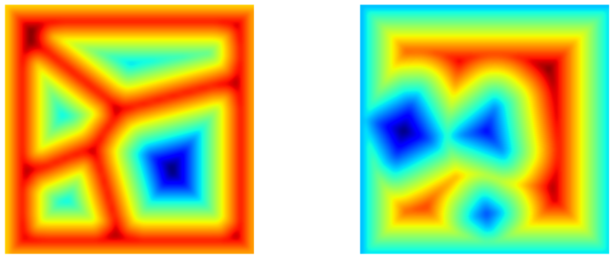


**FIGURE 5**. SIGNED DISTANCE FIELD (SDF) FOR TWO GEOMETRIES; SMALLER VALUES ARE BLUE, LARGER VALUES ARE RED

**Signed Distance Field** We compute several quantities at each node to serve as local features. First, x- and y-coordinates of a node are fed into the model, as these identify a node's location within a mesh. Another relevant quantity we input is the nodal Signed Distance Field (SDF) value. The SDF of a shape has a magnitude equal to the distance to the nearest boundary, and is positive for points on the interior of the geometry, negative for external/void regions, and 0 on the boundary [31]. On an arbitrary domain, the SDF can be represented as a solution to the Eikonal equation [32], which can be computed efficiently through the fast marching method [33]. Figure 5 shows a visualization of the SDF for one shape in the Voronoi Set and one in the Lattice Set. The "distance to a boundary" is a useful value for representing geometry locally, and as such, similar metrics are frequently used in image processing applications [34]. Hence,

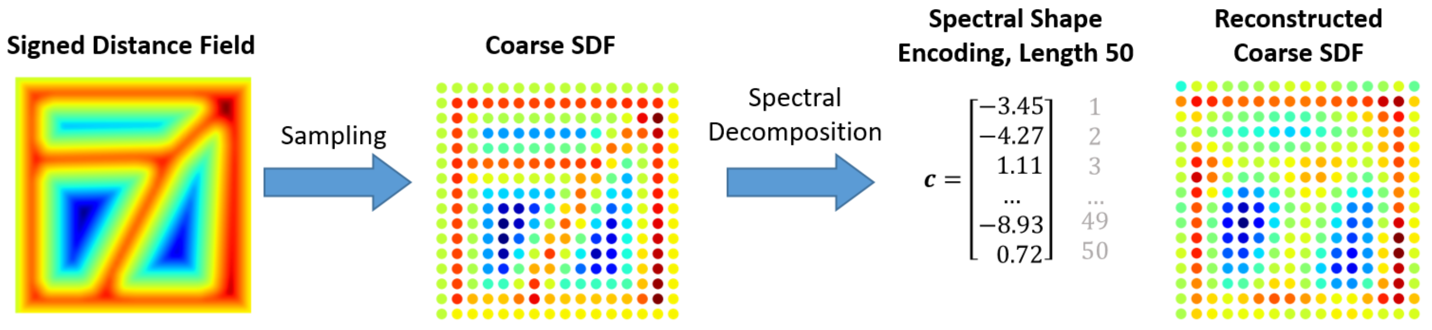

Signed Distance Field
Sampling
Coarse SDF
Spectral Decomposition
Spectral Shape Encoding, Length 50
$c = \begin{bmatrix} -3.45 \\ -4.27 \\ 1.11 \\ \dots \\ -8.93 \\ 0.72 \end{bmatrix}$ 1 2 3 … 49 50
Reconstructed Coarse SDF

**FIGURE 6**. COMPUTING GLOBAL FEATURES – A SPECTRAL SHAPE ENCODING – FROM THE SIGNED DISTANCE FIELD OF A GRAPH

we use SDF as a distinct nodal feature for scalar field prediction, along with $x$ and $y$ coordinates.

**Local Features** In addition to the node coordinates and SDF, both of which carry information about a node with respect to the part, we propose local features that contain information about the region surrounding each node. We consider these separately from the coordinates and SDF because instead of being pre-computed, they will be produced during model evaluation by performing convolution on image representations of the geometry.

In image classification and segmentation problems, machine learning solutions typically employ convolution to learn filters that produce useful local image features [35, 36]. However, unlike these applications, we want to predict scalar values at each node, not on a grid of pixels. Computation of local features therefore requires first performing convolution on input geometric images, and then interpolating the resulting feature maps to each node.

To start, we define two geometric images: the SDF image, found by sampling the SDF on grid coordinates, and a binary image, which is the same size as the SDF image, but has the value 0 in void regions and 1 on the geometry. In theory, the SDF image alone contains enough information to reproduce the binary image; however, to aid in learning high-quality feature maps, we provide both as inputs with slight redundancy. An example of each is included in Fig.4.

We perform image convolution using these two images as input channels. For our implementation, all convolutions use $5 \times 5$ kernels, zero-padding 2, and stride 1, starting with $32 \times 32$ input images. The output of the convolution is a set of feature maps, which serve as local features, because the convolutions operate on the local neighborhood of each pixel. Therefore, we found that we can use these feature maps directly, without applying non-linearities using multiple layers of convolution.

To transform these local feature maps into a set of local features defined at each node, we interpolate each of them to the coordinates of each node. Any differentiable image interpolation scheme can be used for this [37], but we achieved good results with third-order Hermite interpolation.

**Global Features** As the model is simply an MLP evaluated at each node of a mesh, the local features of the node are not enough to make a prediction. A predictive model will need a description of the global geometry in addition to local features, in order to put a given node in full context. The set of global features proposed here is a *spectral shape encoding*. The goal is to have a fixed-length vector description of the entire geometry that can be inputted to the neural network as a set of additional features at each node.

We propose a two-step method to accomplish this: 1. Sample the SDF on a common graph, and 2. Perform a spectral decomposition of the graph. The method begins with the SDF because this field encodes the location of the boundary by default (its value is zero on boundary points, and it takes on a positive value at nodes within the geometry). By sampling points on the SDF corresponding to locations of nodes on a common graph, in this case a grid of size $16 \times 16$ with connections to nearest neighbors and along diagonals, an essentially image-based thumbnail representation of the geometry is obtained. However, the prediction is not performed in this coarsely-sampled space, since the resolution of the original mesh still must be maintained – Instead, the sampled SDF serves to generate a global shape descriptor that has consistent size across meshes with disparate node counts. Generation of a spectral shape encoding using this two-step process is illustrated for one geometry in Fig.6.

Next, we perform a spectral decomposition of this field on the common graph. A spectral decomposition is a method for representing a scalar field not as a set of nodal values, but as a set of coefficients. First, a Laplacian matrix of the graph is computed

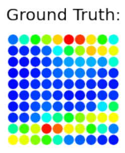


**FIGURE 7**. RECONSTRUCTION OF FIELD ON A $10 \times 10$ GRID USING SPECTRAL COEFFICIENT VECTORS OF LENGTH $k$

according to:

$$L_{ij} = \begin{cases} \frac{-1}{\text{edge length}} & \text{if there exists edge } ij \\ 0 & \text{if there is no edge } ij \\ -\sum_{i \neq j} L_{ij} & \text{if } i = j \end{cases}, \tag{1}$$

which describes the connectivity within the graph in matrix form [38]. Laplacian matrices like this are often used for dimensionality reduction of graph structures.

By computing the eigenvectors $E = [\boldsymbol{e}_1\ \boldsymbol{e}_2\ \ \ldots\ \ \boldsymbol{e}_N]$ of this matrix, any scalar field can be represented as a weighted sum of the eigenvectors $\boldsymbol{e}_i$ [39]. As a matrix equation, this means for a scalar field $\boldsymbol{y}$,

$$\boldsymbol{y} = E\ \boldsymbol{c} \tag{2}$$

for some vector of weight coefficients $\boldsymbol{c}$. In this case, $\boldsymbol{c}$ can be computed directly by inverting $E$ and multiplying by $y$. However, the eigenvectors corresponding to the smallest eigenvalues contain the most useful low-frequency information, an idea often applied in spectral clustering [40, 41], so we can truncate $E$ into the matrix $\tilde{E}$, containing only the $k$ smallest eigenvectors, i.e. $\tilde{E} = [\boldsymbol{e}_1\ \boldsymbol{e}_2\ \ \ldots\ \ \boldsymbol{e}_k]$, where $\boldsymbol{y} \approx \tilde{E}\ \boldsymbol{c}$. Now, the unknown vector $\boldsymbol{c}$ has length $k$, yet it still can be used to reconstruct the original scalar field $\boldsymbol{y}$ in a least-squares sense by solving:

$$\boldsymbol{c}^* = \underset{\boldsymbol{c}}{\operatorname{argmin}} \sum_{i=1}^{k} \left( (\tilde{E}\boldsymbol{c})_i - \boldsymbol{y}_i \right)^2. \tag{3}$$

.

This solution can be obtained using gradient descent, matrix factorization, or, for small $\boldsymbol{c}$ vectors, pseudo-inverse ($\boldsymbol{c} = \left(E^\top E\right)^{-1} E^\top\ \boldsymbol{y}$). Even without the full-length $\boldsymbol{c}$ vector, reconstruction using Eqn.2 gives a qualitatively good reconstruction of the target field, which we demonstrate for an arbitrary field on a $10 \times 10$ grid in Fig.7.

The spectral shape encoding vector $\boldsymbol{c}$ is computed once for a graph, and the whole vector is appended to the set of features at each node, to serve as a global shape description, before it is plugged into the model.

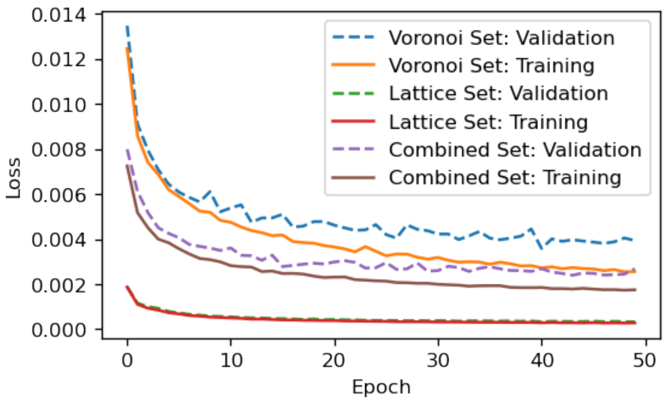


**FIGURE 8**. LOSS DURING EACH EPOCH OF TRAINING FOR ALL MODELS

## Training Details

The MLP contained 3 layers of 128, 128, and 96 hidden neurons, respectively, and used the ReLU activation function at each hidden layer. For the local features, 16 feature maps were computed from an SDF resolution of $32 \times 32$. A spectral shape encoding of length 50 is used for the global features at each node, from a common grid of size $16 \times 16$.

Each model was trained for 50 epochs using the Adam optimizer [42] in PyTorch [43]. Technically, the final MLP evaluates on each node individually, but we pass in each mesh during training as a mini-batch, which should induce more efficient training than pure stochastic gradient descent [44]. Each epoch is therefore a randomly-ordered pass through all training set meshes, inputted batch-wise into the model, one by one. Loss is the mean-squared error (MSE) in von Mises stress prediction across the entire mesh, given by

$$L(\boldsymbol{\sigma}, \boldsymbol{f}) = \frac{1}{n} \sum_{i=1}^{n} (\sigma_i - f_i)^2 \tag{4}$$

where $\sigma_i$ is the $i$th element of ground truth field values $\boldsymbol{\sigma}$, $f_i$ is the $i$th element of predicted field values $\boldsymbol{f}$, and $n$ is the number of nodes in the mesh. Because this loss function operates across all nodes, it may be preferred over a loss function that averages across pixel/voxels. It penalizes incorrect predictions without compromising resolution of fine details, and it allows areas of low node density to be considered less important to the final prediction.

Figure 8 shows the convergence of loss during training for both datasets. Note that although there is some difference between training and validation loss, both exhibit convergence. Between datasets, however, the model trained on the Lattice Set converged to a smaller loss than the Voronoi Set model. This may indicate that our method performs better on certain types

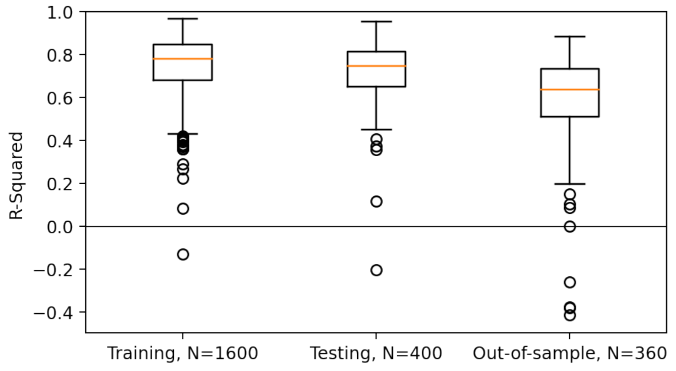


**FIGURE 9**. DISTRIBUTIONS OF $R^2$ FOR THE MODEL TRAINED ON THE COMBINED SET

of shapes than others. However, the typical magnitude of von Mises stress fields was lower for meshes in the Lattice Set than for the Voronoi Set. Hence, a comparison of MSE values may be misleading, so we instead look at $R^2$ for comparable model evaluation.

Training a single model on the Combined Set took 42 minutes on an Intel Core i7-11700 CPU. Generating a 2000-mesh finite-element stress dataset in MATLAB took approximately 13 minutes. Using the model to predict all 2000 stress fields takes 12 seconds. Thus, once trained, using the model as a finite-element method surrogate results in about 65× speedup, a figure which will likely be more drastic for a field that requires a more intensive simulation to compute.

## RESULTS AND DISCUSSION

Here we present our results for stress field predictions, before investigating the contributions of local and global features, demonstrating the model's upsampling ability, and finally verifying that it can make predictions of another field (steady-state temperature).

### Stress Field Prediction

Because some structures result in much higher peak stresses than others, in this section we compare models using the $R^2$ goodness-of-fit measure [45],

$$R^2 = 1 - \frac{\sum_{i=1}^{n} (\sigma_i - f_i)^2}{\sum_{i=1}^{n} (\sigma_i - \overline{\sigma})^2}, \tag{5}$$

where $\sigma_i$ is the ground truth field value at node $i$, $\overline{\sigma}$ is the mean of all ground truth field values, $f_i$ is the predicted field value at node $i$, and $n$ is the number of nodes in the mesh. To

| Dataset | Median $R^2$ | | |
|---|---|---|---|
| | Training | Testing | Out-of-sample |
| Voronoi Set | 0.850 | 0.784 | 0.635 |
| Lattice Set | 0.887 | 0.866 | 0.843 |
| Combined Set | 0.780 | 0.748 | 0.637 |

**TABLE 1**. MEDIAN $R^2$ FOR STRESS PREDICTIONS ON VORONOI, LATTICE, AND COMBINED DATASETS

supplement an $R^2$ value, we will also plot predicted-vs-actual stress for several examples. A good model will have an $R^2$ value close to 1 and a predicted-vs-actual plot that is approximately linear with a slope 1 and y-intercept 0.

Figure 9 is a box-and-whisker plot showing the spread of $R^2$ values for the model trained on the Combined Set. The plot has several outliers (shown as circles), but the typical $R^2$ values are close between the training and testing sets. This indicates that a trained model can be expected to perform roughly the same on geometries in the training set as it can on a random geometry generated with the same parameters. Out-of-sample evaluation reveals that our model may struggle to capture geometries that significantly differ from those in the training set. For reference, the median values of $R^2$ on all three partitions of both datasets are tabulated in Tab.1.

A testing set median $R^2$ of 0.748 for the model trained on the Combined Set demonstrates that each model has good overall performance, although there is room for further improvement to be made. Interestingly, there is not a significant decrease in prediction quality for the model trained on *both* Voronoi and Lattice sets, compared to the models trained on the individual datasets. This reveals that a single model can indeed make predictions for multiple types of shapes, and it will be limited mainly by the diversity of shapes in its training dataset.

Figure 10 shows the predicted field, ground truth field, error, and predicted-vs-actual plots for the best-case, median case, and the worst-case graphs in the testing sets for both shape datasets– these graphs have not been seen during training, but are generated using the same parameters.

A typical prediction captures the qualitative behavior of the stress field, accurately predicting the locations of the peak stress. For example, all four predictions shown in Fig.10 have high-stress locations that closely match their corresponding locations on the ground truth mesh, although their values may not match perfectly.

### Individual contributions of local and global features

To examine the contributions of local and global features, we trained additional models on the Combined Set: models whose

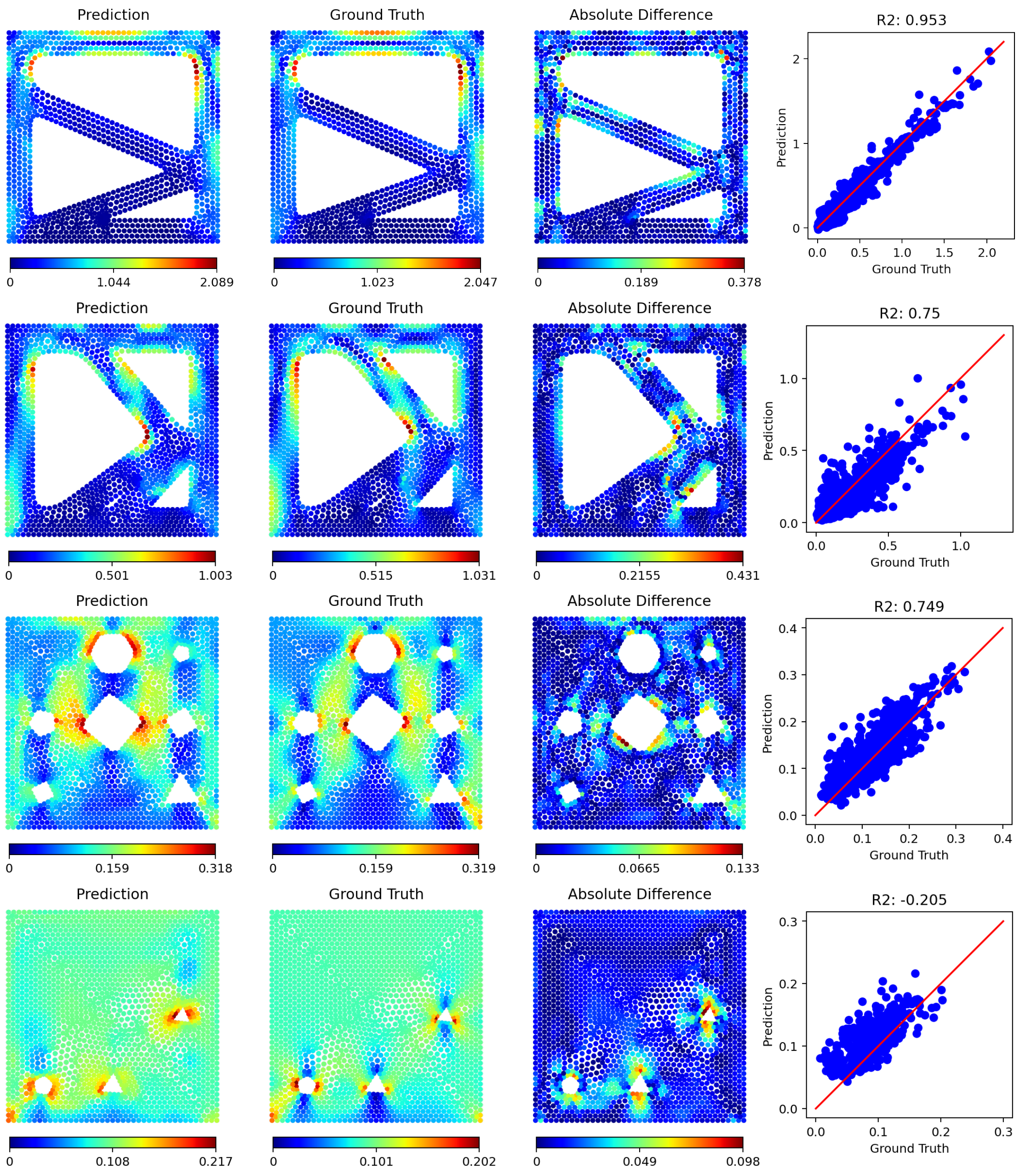


**FIGURE 10**. STRESS PREDICTION, GROUND TRUTH, ERROR, AND PREDICTED-VS-ACTUAL PLOTS FOR TESTING SET MESHES ON WHICH THE MODEL PERFORMED (FROM TOP TO BOTTOM): 1. BEST; 2. NEAR-MEDIAN, VORONOI; 3. NEAR-MEDIAN, LATTICE; 4. WORST. STRESS VALUES ARE IN $10^4 \times$ PASCALS.

inputs only contain a subset of the inputs to the MLP. That is, we trained one model without any local features, one model without global features, and one model without node coordinates or SDF. By comparing the $R^2$ values, we can intuit how much each set of features impacts model predictions. This process lets us verify the efficacy of the spectral shape encoding as a global feature vector for the two datasets. Table 2 shows the median $R^2$ values across both testing datasets for the models described.

Overall, models with *either* local features *or* global features, perform markedly worse than a model trained with both. Clearly, both features are essential to achieving maximum performance.

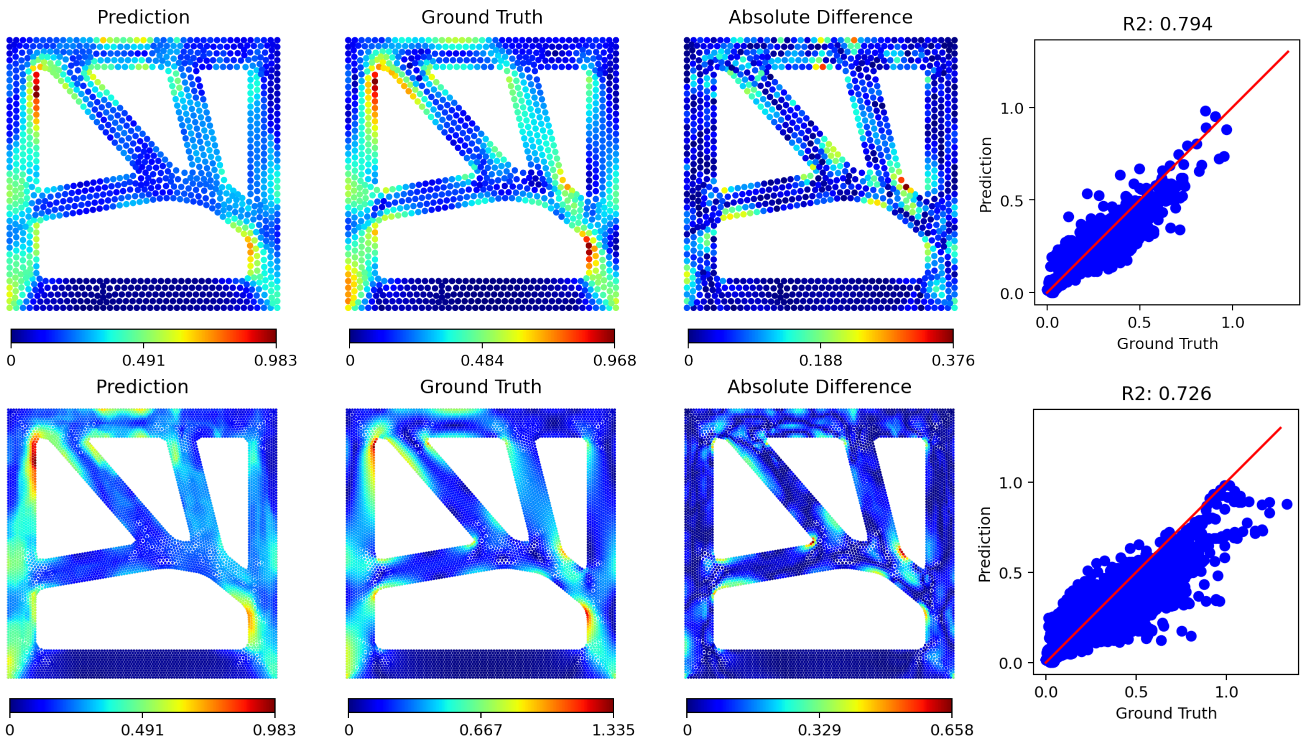


**FIGURE 11**. PREDICTION, GROUND TRUTH, ERROR, AND PREDICTED-VS-ACTUAL PLOTS FOR COARSE (TOP) AND FINE (BOTTOM) MESHES. STRESS VALUES ARE IN $10^4 \times$ PASCALS.

| Input Features | | | Median $R^2$ | | |
|---|---|---|---|---|---|
| (x,y, SDF) | Local | Global | Test | Train | Out |
| ✓ | ✓ | - | 0.650 | 0.638 | 0.540 |
| ✓ | - | ✓ | 0.544 | 0.525 | 0.353 |
| - | ✓ | ✓ | 0.684 | 0.663 | 0.529 |
| ✓ | ✓ | ✓ | 0.780 | 0.748 | 0.637 |

**TABLE 2**. MEDIAN $R^2$ FOR TRAINING, TESTING, AND OUT-OF-SAMPLE SETS, FOR MODELS TRAINED WITH DIFFERENT INPUT COMBINATIONS ON THE COMBINED SET

## Upsampling to a higher mesh resolution

We claim that the mesh-independence of our method enables a model trained on a coarse-resolution mesh to make a prediction on a fine-resolution mesh without any additional training. After all, the model performs separate predictions for each node, and each node has a set of features, the spectral shape encoding, that describes the entire geometry regardless of mesh refinement. Th local features similarly have no dependence on mesh resolution. On the original datasets, the maximum edge length on each mesh was set to 0.025m. Generating a mesh on the Voronoi Set with a maximum edge length of 0.01m (roughly 6 times as many nodes per mesh), we can make a stress field prediction on the finer mesh because interpolation is built into the network for computation of local features. Figure 11 shows a visualization of a coarse and fine mesh for the same Voronoi geometry, evaluated by the model trained on the Combined Set. Note that the field generated is similar for both, demonstrating the interpolation capabilities of our method.

While we cannot expect that training only on coarse meshes will enable the model to make predictions on fine meshes *accurately* (this is not comparable to a mesh convergence study), this analysis demonstrates the flexibility that the method offers. Once trained for a physical problem, an engineer can use any mesh generation technique and any mesh resolution, and the model will make a prediction according to the physical problem on which it was trained.

## Alternate scalar field: Predicting temperature

We have posited that by demonstrating on a von Mises stress field, results of predicting scalar fields in other engineering disciplines will be similar. As an example, we will also show that our method is effective at predicting a temperature field in a steady-state heat transfer scenario. We therefore pose another problem

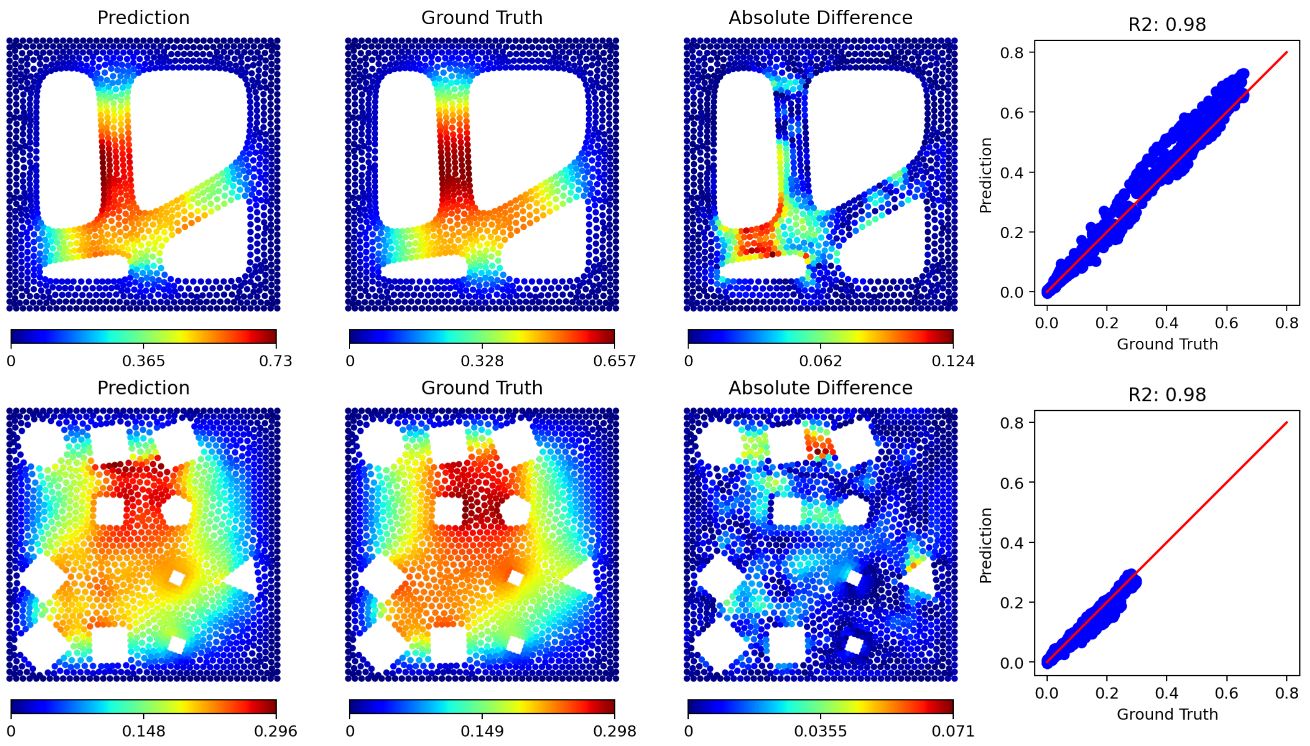


**FIGURE 12**. TEMPERATURE FIELD PREDICTION, GROUND TRUTH, ERROR, AND PREDICTED-VS-ACTUAL PLOTS FOR NEAR-MEDIAN PREDICTIONS ON VORONOI (TOP) AND LATTICE (BOTTOM) TESTING SETS. TEMPERATURE VALUES ARE IN °C.

| Dataset | Median $R^2$ | | |
|---|---|---|---|
| | Training | Testing | Out-of-sample |
| Voronoi Set | 0.987 | 0.980 | 0.941 |
| Lattice Set | 0.988 | 0.986 | 0.976 |
| Combined Set | 0.983 | 0.980 | 0.958 |

**TABLE 3**. MEDIAN $R^2$ FOR TEMPERATURE PREDICTIONS ON VORONOI, LATTICE, AND COMBINED DATASETS

on the same 2-D domain: The part has the thermal conductivity of aluminum, and all four outer boundary walls have a fixed temperature constraint of 0 °C. Each pore now acts as a heat source with constant heat flux 100 W/m$^2$. We assume unit part thickness. These quantities are once again chosen arbitrarily, and although they result in very low temperature values across the mesh – from 0 °C to about 0.6 °C – the distribution has enough variation that we can still judge whether our method is capable of predicting a temperature field.

Table 3 contains the temperature prediction $R^2$ values for all three partitions of models trained on the Voronoi, Lattice, and Combined Sets. Clearly, this field is less demanding to predict, with testing set median $R^2$ values of 0.980 and 0.986 for the individual datasets' models and 0.980 for a model trained on both. We believe the smoothness of the temperature field (i.e. the lack of steep spatial gradients) makes capturing its behavior easier for the network. This can be observed in Fig.12, which depicts two typical predictions in the testing set for the "combined" model.

## LIMITATIONS AND FUTURE WORK

The current model has several limitations we wish to address in future work. One drawback is that each trained model can make predictions of *one* field for *one* material in *one* loading case. Future work will look at parametric variation of loads and boundary conditions. Material properties, including those for nonhomogeneous materials, can also be inputted as features. Prediction of other fields, including vector fields or even dynamic fields, may be investigated as well.

Since 3-D geometries are more common throughout engineering, we are interested in adapting the model to make predictions on 3-D meshes. This will add complexity to the computation of the SDF and may require reformulation of the spectral shape encoding to better operate on a 3-D structure.

Restriction of the geometry to a $1\text{m} \times 1\text{m}$ square envelope is also a severe geometric constraint, but this type of constraint is reasonable in that it mimics many real design decisions in engineering. In spite of this, we are still interested in expanding the spectral shape encoding framework to operate on shapes with fewer restrictions.

## CONCLUSION

In this work, we proposed a model that makes predictions for scalar fields by computing a set of local and global features at every node on a mesh. Both sets of features are shown to be important to making field predictions, and the set of global features – a spectral shape encoding – provides a succinct description of the geometry that efficiently represents the global shape at every node.

A model was trained, on a large dataset of complex shapes, to predict the von Mises stress field for a part undergoing compression. Our model achieved an overall median $R^2$ of approximately 0.75 for stress field prediction on the testing set. Validating on a steady-state temperature problem, the model has even stronger performance, yielding $R^2$ values above 0.98. Thus, the proposed method demonstrates the potential to replace finite-element computations with less expensive surrogate model evaluations in an engineering design setting.

Future work will explore variation of boundary conditions and expansion to 3-D geometries, for a more robust field prediction model.

## ACKNOWLEDGMENT

This research was funded by Air Force Research Laboratory S111068002.

## REFERENCES

[1] Rankin, J. J., and Ott, D. A., 1992. "The open approach to fea integration in the design process". *Mechanical Engineering,* ***114***(9), p. 70.

[2] Torczon, V., and Trosset, M., 1998. "Using approximations to accelerate engineering design optimization". In 7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, p. 4800.

[3] Eriksson, M., Burman, Å., et al., 2005. "Improving the design process by integrating design analysis". In DS 35: Proceedings ICED 05, the 15th International Conference on Engineering Design, Melbourne, Australia, 15.-18.08. 2005, pp. 555–556.

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

[5] Barmada, S., Fontana, N., Formisano, A., Thomopulos, D., and Tucci, M., 2021. "A deep learning surrogate model for topology optimization". *IEEE Transactions on Magnetics,* ***57***(6), pp. 1–4.

[6] Nie, Z., Jiang, H., and Kara, L. B., 2019. "Stress field prediction in cantilevered structures using convolutional neural networks". *Journal of Computing and Information Science in Engineering,* ***20***(1), 9.

[7] Jiang, H., Nie, Z., Yeo, R., Farimani, A. B., and Kara, L. B., 2021. "Stressgan: A generative deep learning model for two-dimensional stress distribution prediction". *Journal of Applied Mechanics,* ***88***(5), 2.

[8] Khadilkar, A., Wang, J., and Rai, R., 2019. "Deep learning–based stress prediction for bottom-up sla 3d printing process". *The International Journal of Advanced Manufacturing Technology,* ***102***(5), pp. 2555–2569.

[9] Qi, C. R., Su, H., Mo, K., and Guibas, L. J., 2017. Pointnet: Deep learning on point sets for 3d classification and segmentation.

[10] Qi, C. R., Yi, L., Su, H., and Guibas, L. J., 2017. Pointnet++: Deep hierarchical feature learning on point sets in a metric space.

[11] Kashefi, A., Rempe, D., and Guibas, L. J., 2021. "A point-cloud deep learning framework for prediction of fluid flow fields on irregular geometries". *Physics of Fluids,* ***33***(2), 2, p. 027104.

[12] Attene, M., Katz, S., Mortara, M., Patane, G., Spagnuolo, M., and Tal, A., 2006. "Mesh segmentation - a comparative study". In IEEE International Conference on Shape Modeling and Applications 2006 (SMI'06), pp. 7–7.

[13] Kalogerakis, E., Hertzmann, A., and Singh, K., 2010. "Learning 3D Mesh Segmentation and Labeling". *ACM Transactions on Graphics,* ***29***(3).

[14] Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., and Yu, P. S., 2021. "A comprehensive survey on graph neural networks". *IEEE Transactions on Neural Networks and Learning Systems,* ***32***(1), 1, p. 4–24.

[15] Wang, Y., Sun, Y., Liu, Z., Sarma, S. E., Bronstein, M. M., and Solomon, J. M., 2019. Dynamic graph cnn for learning on point clouds.

[16] Sanchez-Gonzalez, A., Godwin, J., Pfaff, T., Ying, R., Leskovec, J., and Battaglia, P. W., 2020. Learning to simulate complex physics with graph networks.

[17] Meyer, L., Pottier, L., Ribes, A., and Raffin, B., 2021. Deep surrogate for direct time fluid dynamics.

[18] Li, G., Müller, M., Thabet, A., and Ghanem, B., 2019. Deepgcns: Can gcns go as deep as cnns?

[19] Pfaff, T., Fortunato, M., Sanchez-Gonzalez, A., and Battaglia, P. W., 2021. Learning mesh-based simulation with graph networks.

[20] Zhao, L., and Akoglu, L., 2020. Pairnorm: Tackling oversmoothing in gnns.

[21] Javadi, A., and Tan, T., 2003. "Neural network for constitutive modelling in flnite element analysis". *Computer*

*Assisted Mechanics and Engineering Sciences,* ***10***, 01.

[22] de Avila Belbute-Peres, F., Economon, T. D., and Kolter, J. Z., 2020. Combining differentiable pde solvers and graph neural networks for fluid flow prediction.

[23] Mardhiyah, I., Madenda, S., Salim, R. A., and Wiryana, I. M., 2016. "Dimensionality reduction of laplacian embedding for 3d mesh reconstruction". *Journal of Physics: Conference Series,* ***725***, jun, p. 012007.

[24] Belkin, M., and Niyogi, P., 2003. "Laplacian eigenmaps for dimensionality reduction and data representation". *Neural Computation,* ***15***(6), pp. 1373–1396.

[25] Guo, T., Lohan, D., Allison, J., Cang, R., and Ren, Y., 2018. "An indirect design representation for topology optimization using variational autoencoder and style transfer". In AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials, no. 210049 in AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, 2018, American Institute of Aeronautics and Astronautics Inc, AIAA. Publisher Copyright: © 2017 Tinghao Guo, Danny J. Lohan, Ruijin Cang, Max Yi Ren and James T. Allison.; AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, 2018 ; Conference date: 08-01-2018 Through 12-01-2018.

[26] Liang, L., Liu, M., Martin, C., and Sun, W., 2018. "A deep learning approach to estimate stress distribution: a fast and accurate surrogate of finite-element analysis". *Journal of The Royal Society Interface,* ***15***(138), p. 20170844.

[27] Brnic, J., 2018. *Analysis of Engineering Structures and Material Behavior*. John Wiley & Sons.

[28] Kou, X., and Tan, S., 2010. "A simple and effective geometric representation for irregular porous structure modeling". *Computer-Aided Design,* ***42***(10), pp. 930–941.

[29] Sorkine, O., Cohen-Or, D., Lipman, Y., Alexa, M., Rössl, C., and Seidel, H.-P., 2004. "Laplacian surface editing". SGP '04, Association for Computing Machinery, p. 175–184.

[30] The MathWorks, I., 2021. *Partial Differential Equation Toolbox*. Natick, Massachusetts, United States.

[31] Osher, S., and Fedkiw, R., 2003. "Signed distance functions". In *Level set methods and dynamic implicit surfaces*. Springer, pp. 17–22.

[32] Mauch, S., 2000. "A fast algorithm for computing the closest point and distance transform". *Go online to http://www. acm. caltech. edu/seanm/software/cpt/cpt. pdf*.

[33] Sethian, J. A., 1996. "A fast marching level set method for monotonically advancing fronts". *Proceedings of the National Academy of Sciences of the United States of America,* ***93***(4), pp. 1591–1595.

[34] Strutz, T., 2021. The distance transform and its computation.

[35] Sultana, F., Sufian, A., and Dutta, P., 2018. "Advancements in image classification using convolutional neural network". In 2018 Fourth International Conference on Research in Computational Intelligence and Communication Networks (ICRCICN), IEEE.

[36] Minaee, S., Boykov, Y., Porikli, F., Plaza, A., Kehtarnavaz, N., and Terzopoulos, D., 2020. Image segmentation using deep learning: A survey.

[37] Thévenaz, P., Blu, T., and Unser, M., 2000. "Image interpolation and resampling". *Handbook of medical imaging, processing and analysis,* ***1***(1), pp. 393–420.

[38] Chung, F., Graham, F., on Recent Advances in Spectral Graph Theory, C. C., (U.S.), N. S. F., Society, A. M., and of the Mathematical Sciences, C. B., 1997. *Spectral Graph Theory*. CBMS Regional Conference Series. Conference Board of the mathematical sciences.

[39] Bruna, J., Zaremba, W., Szlam, A., and LeCun, Y., 2014. Spectral networks and locally connected networks on graphs.

[40] von Luxburg, U., 2007. A tutorial on spectral clustering.

[41] Filippone, M., Camastra, F., Masulli, F., and Rovetta, S., 2008. "A survey of kernel and spectral methods for clustering". *Pattern recognition,* ***41***(1), pp. 176–190.

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

[43] Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., and Lerer, A., 2017. "Automatic differentiation in pytorch".

[44] Cotter, A., Shamir, O., Srebro, N., and Sridharan, K., 2011. "Better mini-batch algorithms via accelerated gradient methods". *Advances in neural information processing systems,* ***24***.

[45] Miles, J., 2005. *R-Squared, Adjusted R-Squared*. John Wiley and Sons, Ltd.