Deep learning-based reconstruction of the structure of heterogeneous composites from their temperature fields

Haiyi Wu, Hongwei Zhang, Guoqing Hu, and Rui Qiao

AFFILIATIONS
1 Department of Mechanical Engineering, Virginia Tech, Blacksburg, Virginia 24061, USA
2 Department of Engineering Mechanics, Zhejiang University, Hangzhou 310027, China

ABSTRACT
Inverse problems involving transport phenomena are ubiquitous in engineering practice, but their solution is often challenging. In this work, we build a data-driven deep learning model to predict the heterogeneous distribution of circle-shaped fillers in two-dimensional thermal composites using the temperature field in the composite as an input. The deep learning model is based on convolutional neural networks with a U-shape architecture and encoding–decoding processes. The temperature field is cast into images of 128 × 128 pixels. When the true temperature at each pixel is given, the trained model can predict the distribution of fillers with an average accuracy of over 0.979. When the true temperature is only available at 0.88% of the pixels inside the composite, the model can predict the distribution of fillers with an average accuracy of 0.94, if the temperature at the unknown pixels is obtained through the Laplace interpolation. Even if the true temperature is only available at pixels on the boundary of the composite, the average prediction accuracy of the deep learning model can still reach 0.80; the prediction accuracy of the model can be improved by incorporating true temperature in regions where the model has low prediction confidence.

© 2020 Author(s). All article content, except where otherwise noted, is licensed under a Creative Commons Attribution (CC BY) license (http://creativecommons.org/licenses/by/4.0/). https://doi.org/10.1063/5.0004631

I. INTRODUCTION
Inverse problems involving transport phenomena are ubiquitous in engineering design, optimization, and diagnosis. Common applications include finding the configuration of systems and the properties of systems’ materials to achieve the desired transport behavior, e.g., the structure and loading, which should be adopted for the constituents of a thermal composite so that it provides the required thermal conductivity. Monte Carlo (MC)-based techniques have traditionally been used to solve inverse problems. However, these techniques tend to be computationally expensive because a large number of realizations are often needed to obtain accurate results in large parameter space. Many new methods have been developed to lower the computational cost for solving inverse problems. For example, surrogate models have been constructed via generalized polynomial chaos expansion to perform inverse uncertainty quantification of thermal-hydraulic problems, and Gaussian process regression-based inverse models have been used to estimate the thermal properties of orthotropic materials. However, these models often have difficulty in scaling to problems in which the input fields are high-dimensional.

Machine learning-based techniques can potentially be an effective approach for tackling the above difficulties. In particular, one branch of machine learning, deep learning, is gaining popularity in many disciplines (e.g., computer vision and natural language processing) and has shown promise in solving transport phenomena in complex systems. For example, the surrogate model based on the deep convolutional encoder–decoder networks has been used for uncertainty quantification in a single-phase flow, and deep learning models with a variational auto-encoder have been applied for the uncertainty quantification in reservoir simulations. A convolutional neural network (CNN) has been used to predict the effective transport properties of porous media with a complex structure with computational cost orders of magnitude lower;
neural networks (DNN) have been used to predict the multiphase flow in heterogeneous media. Nevertheless, the applications of deep learning methods for inverse problems involving transport phenomena in heterogeneous media are still limited. There also exist some conceivable challenges when deep learning is used in such problems. To ensure accuracy, deep learning-based models often require a large amount of input data, which are difficult or expensive to obtain in practice. Furthermore, obtaining fine-grained input data also needs a large number of sensors to be deployed. As suggested by recent works on sparse sensor placements for signal reconstruction and fluid flow reconstruction from limited measurements, a large number of sensors can potentially modify the data to be measured, thereby, introducing systematic errors into the model’s prediction. Therefore, whether and to what extent coarse-grained input data can be used effectively in deep learning to solve inverse problems involving high-dimensional data (e.g., when heterogeneous media are involved) remains a critical but open question.

In this work, we develop a deep learning-based model to predict the heterogeneous structure of a thermal composite from the composite’s temperature fields. For composites, “heterogeneous” can be interpreted in three ways: (1) the spatial distribution of fillers is not homogeneous (e.g., fillers are randomly, rather than homogeneously, distributed within a matrix), and (2) the fillers are irregular in shape and size, and (3) fillers with irregular shape/size are distributed in a non-homogeneous fashion in a matrix. Here, we focus on the first kind of heterogeneity. Specifically, circle-shaped fillers with high thermal conductivity are dispersed randomly in a matrix with low thermal conductivity. We note that the circle-shaped fillers are common, e.g., the cross section of carbon nanotubes is circular, and spherical solid particles are often used in thermal composites. Because both the input and output data of this inverse problem can be high-dimensional, we adopt the end-to-end image-to-image regression by using the convolutional encoder–decoder network (U-Net). Conceptually, the encoder network extracts important spatial correlations from the temperature fields, which is then used as an input for the decoder network to predict the heterogeneous structure of the composite. The proposed model is trained and tested in two-dimensional (2D), highly heterogeneous composites using computationally generated datasets. We also investigate the model’s performance and its improvement when the sparse temperature field is used as the input. Finally, we use the uncertainty of the model’s predictions to guide the placement of additional temperature sensors, whose measurement helps improve of the predictions of the deep learning model.

The rest of the manuscript is organized as follows: In Sec. II, we introduce the definition of the inverse problem considered here and the generation of datasets for the deep learning model. In Sec. III, we present our deep learning model based on the convolutional encoder–decoder networks for solving the inverse problem. In Sec. IV, the performance of the developed model is examined and discussed. The discussion and conclusions are provided in Sec. V and Sec. VI, respectively.

II. PROBLEM DEFINITION AND DATASET GENERATION

Thermal composites are heterogeneous materials in which fillers with high thermal conductivity are dispersed in a base matrix with low thermal conductivity. In forward problems related to these materials, the temperature field (and, consequently, the effective thermal properties) of the composites is predicted using the distribution of the fillers in the base matrix and the thermal conductivity of the fillers and base matrix as the input. In inverse problems, the composite’s temperature field is used as the input to determine other observables of interest, e.g., the distribution of fillers and their thermal conductivity. Here, we consider an inverse problem in a 2D square composite with randomly distributed fillers (see Fig. 1). The blue circles and the gray background represent the fillers and the base matrix, respectively. The temperature field measured at selected points in the composites is used as the input to infer the distribution of the fillers in the matrix. We solve this inverse problem with a data-driven deep learning-based model. Below, we first define a data-driven model for solving this problem. We then will present the numerical method for generating the datasets that are needed for our deep learning model.

A. Deep learning-based model

Without losing generality, we consider a square-shaped composite (see Fig. 1). The base matrix and fillers have uniform thermal conductivity within them. Thus, the composite is characterized by a binary thermal conductivity field $k(x)$. The 2D heat conduction within the composite is governed by

$$\nabla \cdot (k(x) \nabla T(x)) = 0, \quad 0 < x, \quad y < 1,$$

(1)

with boundary conditions

$$(q'') = 0$$

$$(q'') = 0$$

$$(q''_0) = 0$$

$$(q''_0) = 0$$

$$(T_1) = 0$$

$$(T_1) = 0$$

FIG. 1. A two-dimensional composite with a binary thermal conductivity field $k(x)$ (blue circles: fillers with high thermal conductivity; gray background: base matrix with low conductivity). The goal of the inverse model is to predict $k(x)$ from the temperature field measured in the composite under the labeled boundary conditions.
\[ \frac{dT}{dy}|_{y=0} = \frac{dT}{dy}|_{y=1} = 0, \]  
\[ -k \frac{d^2 T}{dx^2} |_{x=0} = q_0; \quad T|_{x=1} = T_1. \]  
Here, \( T(x) \) is the temperature field, which serves as the input of our inverse problem. \( k(x) \) is the binary thermal conductivity field in the composite, which is the desired output of our inverse model.

Mathematically, the above problem is a special case of a more general problem governed by

\[ \mathcal{L}(k(x); T(x)) = 0, \quad x \in \Omega, \]  
\[ B(T(x)) = b(x), \quad x \in \partial \Omega, \]

where \( \mathcal{L} \) is a differential operator, and \( k(x) \in \mathbb{R}^{|\Omega|} \) is the output feature of interest at a spatial location \( x \in \Omega \subset \mathbb{R}^d \) (\( d \) is the dimension of the space, and it can be 1, 2, or 3). \( T(x) \in \mathbb{R}^{|\Omega|} \) is the input field, whose dimension \( d_I \) may differ from the spatial dimension (e.g., input data measured by sensors are available only on a 2D surface). \( B \) defines the boundary conditions of the physical processes occurring on the domain boundary \( \partial \Omega \). Both the input and output fields of this type of inverse problem are high-dimensional.

For the problem shown in Fig. 1, if the binary thermal conductivity \( k(x) \) at \( k \times k \) points within the square domain is to be predicted from the temperature \( T(x) \) measured at \( m \times m \) points in the domain, then, \( T(x) \) and \( k(x) \) are vectors of length \( m^2 \) and \( k^2 \), respectively, i.e., \( T(x) \in \mathbb{R}^{m^2} \) and \( k(x) \in \mathbb{R}^{k^2} \). The inverse problem can be treated as a mathematical mapping,

\[ h : T(x) \rightarrow k(x) \in \mathbb{R}^{m^2} \rightarrow \mathbb{R}^{k^2}. \]  

Usually, obtaining the above mapping is computationally expensive. To tackle this limitation, we adopt an inverse model \( \hat{k}(x) = f(T(x), \theta) \) to approximate the exact mapping of the physical systems \( k(x) = h(T(x), x), \) where \( \theta \) represents some model parameters.

The model \( f \) is trained using \( N \) sets of data \( D = \{T(x), k(x)\}_{i=1}^N \) through the minimization of a loss function \( \text{Loss} \)

\[ f = \arg \min_\theta \text{Loss}(f(T(x), \theta); k(x))). \]  

### III. DEEP CONVOLUTIONAL ENCODER-DECODER NETWORKS (U-NET)

We adopt the deep convolutional encoder–decoder network (U-Net) model to solve the inverse problem defined in Sec. II. Conceptually, this model consists of an encoder network and a decoder network. The encoder network is used to extract important spatial correlations and features from the temperature fields and is constructed from deep CNN. The decoder network takes the extracted features as input and projects the feature maps to high-dimensional space to predict the heterogeneous structure of the composite (i.e., the binary thermal conductivity field). The decoder network is constructed with the transposed convolutional layers to up-sample the feature maps to high-dimensional space. Below, we first briefly introduce the basic operations in CNN. We next present the transposed convolutional operation and its difference from the convolutional operation. We, then, present the architecture of our U-Net model and summarize the computational framework of using U-Net to predict the heterogeneous structure of composites.

#### A. Deep convolutional neural networks

Detailed descriptions of the classical and convolutional neural networks and their data flow, written in a way that is easily accessible for researchers with computational transport phenomena background, can be found in our recent work. Here, we only briefly outline the key idea of these networks. A neural network approximates an input–output relation \( h : \mathcal{X} \rightarrow \mathcal{Y} \) via a series of layers with an affine transformation followed by a nonlinear activation function. For example, the \( k \)th layer of a neural network processes the output of its preceding layer \( \mathcal{Z}^{k-1} \) by first assigning weights and biases to this output and then passing the results to an activation function to generate its output \( \mathcal{Z}^k \),

\[ \mathcal{Z}^k = g^k(W^k \mathcal{Z}^{k-1} + b^k), \]  

where \( W^k \) and \( b^k \) are the weight matrix and bias vector (called the learnable parameters) for this layer, respectively. \( g^k(\cdot) \) is an element-wise nonlinear activation function, such as the tanh function or the rectified linear unit (ReLU) function. For an \( N \)-layer neural networks, its input and output are \( \mathcal{X} = \mathcal{Z}^0, \mathcal{Y} = \mathcal{Z}^N \), respectively. Classical neural networks are fully connected (i.e., a neuron in an arbitrary layer \( k \) is connected to all neurons in its previous layer) and \( W^k \) and \( b^k \) are different for each layer. Therefore, as the...
number of layers and the dimension of the input increase, the number of learnable parameters increases drastically, resulting in enormous computational cost.

CNN is designed to reduce the learnable parameters of neural networks greatly.\textsuperscript{23,24} This is achieved by connecting neurons in a layer \( k \) only to some neurons in layer \( k - 1 \) and sharing learnable parameters for all neurons in the same convolutional layer. When CNN is used in image processing, a series of convolution kernels are applied to compute the output volume, which is termed \textit{feature maps} (as the purpose of a convolutional layer is to extract important features from the input volume). Typically, the number of channels of the feature map is the depth of the output volume of a convolutional layer. For example, consider a convolution layer with an input volume \( X_i \in \mathbb{R}^{h \times w \times d_i} \), where \( h \times w \) is the size of each feature map and \( d_i \) is the depth of the volume. To obtain the \( \beta \)-th feature map \( (F \cdot \text{map}_{\text{conv}}^\beta) \), the \( \beta \)-th filter \( W^\beta \in \mathbb{R}^{h \times w \times d_i} \) (\( g \): kernel size; \( d_i \): the number of filters) is made to slide over the input volume, and the results are combined with the \( \beta \)-th bias \( B \) before passing to the nonlinear activation function \( g(\cdot) \),

\[
F \cdot \text{map}_{\text{conv}}^\beta = g \left( W^\beta \otimes X_i + B(\beta) \cdot J \right),
\]

where \( J \) is a tensor with all elements equal to 1. \( W^\beta \otimes X_i \) denotes the 2D-convolution of the filter kernel with the input. Its detailed expression depends on the padding \( (p) \) and strides \( (s) \), and can be found in many textbooks.\textsuperscript{25} Take, for example, the convolution of a \( 2 \times 2 \) kernel \( (w_j) \) with a \( 3 \times 3 \) input matrix \( (x_{ij}) \), stride \( s_t = 1 \) and padding \( p = 0 \), the output is a \( 2 \times 2 \) matrix \( (y_{ij}) \). If we unroll the input and output into vectors from left to right and top to bottom, the computation of convolution can then be represented as the matrix operation as follows:

\[
\mathcal{M} x = y,
\]

\[
\mathcal{M} = \begin{pmatrix}
 w_{11} & w_{12} & 0 & w_{21} & w_{22} \\
 0 & w_{11} & w_{12} & 0 & w_{21} \\
 0 & 0 & w_{11} & w_{12} & 0 \\
 0 & 0 & 0 & w_{11} & w_{12} \\
 0 & 0 & 0 & 0 & w_{21} & w_{22}
\end{pmatrix}, 
\]

\[
x = (x_{11} \ x_{12} \ x_{13} \ x_{21} \ x_{22} \ x_{23} \ x_{31} \ x_{32} \ x_{33})^T,
\]

\[
y = (y_{11} \ y_{12} \ y_{21} \ y_{22})^T.
\]

Using this formulation, the forward passes during the training step are computed by multiplying the input data with \( \mathcal{M} \), and the error is back-propagated by multiplying the loss with \( \mathcal{M}^T \).

**B. Transposed convolution**

Transposed convolution is widely used in computer vision problems as a decoding layer of a convolution auto-encoder to project the feature maps to a higher dimensional space.\textsuperscript{26-28} For example, people used stride transposed convolution layer to decode the feature maps from convolutional encoding layers and up-sample the feature maps to acquire the image semantic segmentation.\textsuperscript{29} Transposed convolution can be considered as a transformation in the opposite direction of the regular convolution. It works by swapping the forward and backward passes of a regular convolution. As seen in Eq. (8), the forward and backward passes for a regular convolution are computed by multiplying the input \( x \) by \( \mathcal{M} \) and \( \mathcal{M}^T \), respectively; the forward and backward passes for transposed convolutions are calculated by multiplying input \( x \) with \( \mathcal{M}^T \) and \( \mathcal{M} \), respectively. Transposed convolutions are also known as the \textit{fractionally strided convolutions} because it is always possible to emulate a transposed convolution with a direct convolution by adding many columns and rows of zeros to the input.\textsuperscript{30} It should be noted that transposed convolution does not recover the input of the convolution itself, but rather returns the feature maps that have the same size as that of the input.

**C. Deep convolutional encoder–decoder networks (U-Net)**

The convolutional encoder–decoder (U-Net) architecture is an extension of CNN for problems with pixel-wise prediction. It has shown high efficiency in handling the mapping between the high-dimensional input and output. For example, U-Net has been used successfully in biomedical image segmentation\textsuperscript{31-33} and conditional shape generation.\textsuperscript{34} Inspired by these works, we adopt the U-Net architecture for solving the inverse model defined in Sec. II, i.e., to predict the binary conductivity field of a 2D composite from its temperature field.

The architecture of our U-Net model is shown in Fig. 2. The measured temperature field, represented as a 2D image, is the input for the U-Net and the binary conductivity field, in which a pixel value of 1 (−1) corresponding to a local conductivity \( k = 200(4) \) \( W/(mK) \) is the output prediction. To reduce the computation costs, the input temperature field images are down-sampled to 64 × 64 pixels using kernel\textsuperscript{35} before feeding to the U-Net model. Below, we outline the different layers in the U-Net model.

As illustrated in Fig. 2, our U-Net model consists of an encoder with a contracting path (left side), a decoder with an expansive path (right side), and a bottleneck that connects the encoder and the decoder part. The encoder part follows the typical structure of CNN, with four repeating units and each unit contains three layers: (1) the input data of each layer is applied with a \( 3 \times 3 \) convolutional layer (all convolution layers in this study have the same padding and unit stride) and followed by an ReLU activation function; (2) the output feature maps of the first layer are down-sampled with a \( 2 \times 2 \) max pooling layer with stride \( s_t = 2 \); (3) to avoid overfitting, a dropout layer with keep ratio \( kp = 0.5 \) is applied before feeding the data to the convolution layer of the next unit.

The decoder part consists of four repeating units, and each unit contains four layers: (1) the input feature maps are up-sampled via transposed convolutions (all the transposed convolutions in this study are applied with the same padding and a stride of \( s_t = 2 \)) and followed by the ReLU activation function; (2) the output features are then concatenated with its corresponding feature maps from the encoder part by the contracting paths, which are referred to as skip connections; (3) the dropout layer is applied to the concatenated features to avoid overfitting; (4) the output results from the dropout layer are fed to a regular convolution layer and followed by the ReLU activation function.

We apply a convolutional layer at the bottleneck to connect the encoder and decoder part. In the last layer of the U-Net, a \( 1 \times 1 \) convolution with tanh activation function is used to map features to the desired output \( k_{\text{final}} \). The final prediction for binary conductivity...
The U-Net model includes hyperparameters such as the size and volume of the feature map produced by each convolution/pooling layer, the choice of the activation function after each convolution layer, the number of total layers, and training epochs. In this work, these hyperparameters are selected based on the performance of the model on the validation dataset. The optimized hyperparameters are marked in Fig. 2 and presented in Sec. III.

The above U-Net model is implemented using Tensorflow. The L2 loss is adapted to quantify how well the model reproduces the training dataset,

$$L_2 = \frac{1}{2} \sum \sum (k_{ij,tanh} - k_{ij})^2,$$

where \(N\) is the number of samples, and \(N_x (N_y)\) is the number of pixels in the \(x\)- (\(y\)-) direction of the image of the 2D composite, \(k_{ij,tanh}\) and \(k_{ij}\) are the predictions by the model after the \(1 \times 1\) convolution with tanh activation function and the ground truth for pixel \(ij\) of a sample \(l\).

The U-Net model is trained using the training dataset. During training, all weights of the kernels are initialized using a truncated normal distribution with a standard deviation of 0.1, and all biases are set to 0.1. Adam Optimizer with a decay learning rate (initialized with \(8 \times 10^{-4}\) and decreased by 40% every 200 epochs) is applied to minimize \(L_2\) in Eq. (10). The training stops at 3000 epochs when the validation error is minimized. We save the optimized weights and biases for all learnable filters every 1000 epochs during the training process. The saved parameters at the final step are then restored for evaluating the testing dataset.

### IV. PERFORMANCE OF DEEP LEARNING MODELS

Here, we examine the performance of our deep learning model for predicting the heterogeneous structure of 2D composites (i.e., their binary thermal conductivity fields) from their temperature fields. The prediction error for a sample \(l\) is defined as

$$E_l = \frac{1}{N_x \times N_y} \sum \sum (\hat{k}_{ij} - k_{ij})^2,$$

where \(\hat{k}_{ij}\) is the model’s prediction of the binary conductivity field. The average accuracy for the testing dataset is defined as

$$\text{Acc} = 1.0 - \frac{1}{N} \sum E_l.$$

The U-Net model is trained using the training dataset. During training, all weights of the kernels are initialized using a truncated normal distribution with a standard deviation of 0.1, and all biases are set to 0.1. Adam Optimizer with a decay learning rate (initialized with \(8 \times 10^{-4}\) and decreased by 40% every 200 epochs) is applied to minimize \(L_2\) in Eq. (10). The training stops at 3000 epochs when the validation error is minimized. We save the optimized weights and biases for all learnable filters every 1000 epochs during the training process. The saved parameters at the final step are then restored for evaluating the testing dataset.

#### A. Deep learning model with full-scale inputs

Figure 3 shows the performance of the deep learning model when its input is taken as the temperature field measured at all pixels in the composite (hereafter, referred to as full temperature field).
FIG. 3. Boxplot of the prediction accuracy and the required number of temperature measurements for deep learning models in this work. Case1: model trained with full temperature measurement (i.e., the temperature is available at each pixel). Case2: model trained with temperature field obtained through the Laplace interpolation of the sparse temperature input. Case3: model trained with temperature field obtained through the Laplace interpolation of the sparse temperature data on composite’s boundaries (<0.6% of the full measurement). Case4: model trained with temperature field obtained through the Laplace interpolation of the sparse temperature data measured at positions suggested by the deep learning model. The average accuracy of the predicted binary conductivity field is $\bar{Acc} = 0.979$. Figure 4(a) shows the temperature field in a representative example. Figures 4(b) and 4(c) show the true and predicted binary thermal conductivity fields in the example shown in Fig. 4(a), respectively. We observe that the conductivity field predicted by the U-Net model agrees well with the true field. The histogram of the prediction error for the testing dataset is presented in Fig. 4(d), and we observe that >70% of the test data have prediction error less than 2%. Interestingly, from the input temperature field shown in Fig. 4(a), we can loosely discern the contours of the high-conductivity fillers. Apparently, the spatial correlations within the temperature field discerned by humans are extracted well by the encoder model [in the form of feature maps, see Eq. (7)]. Otherwise, the binary conductivity field would not be predicted accurately by the decoder model, whose predictive power relies significantly on the effective extraction of features by the encoder model.

When the distribution of fillers becomes more complicated, e.g., when some fillers overlap with each other, it becomes increasingly harder to discern the contours of the fillers in the temperature field by eyes. Nevertheless, the U-Net model can still predict the binary conductivity fields well under these conditions (see Fig. S1 in the supplementary material). This suggests that deep learning can perform better than humans under some conditions.

In the above example, the contrast between the thermal conductivity of the filler [200 W/(mK)] and the matrix [4 W/(mK)] is quite high. To clarify how the contrast between the conductivity of the filler and matrix affects the performance of our model, we performed four sets of studies by keeping the matrix conductivity at $k_m = 4$ W/(mK) but varying the filler conductivity as $k_f = 4.4$ W/(mK), 8 W/(mK), 20 W/(mK), and 200 W/(mK). Our studies, thus, cover a conductivity contrast of $\gamma = k_f/k_m = 1.1$, 2, 5, and 50. The structure of the composite is identical to that used above. Figure 5 summarizes the model performance for these datasets. The model performs quite well for conductivity contrast $\gamma \geq 5$. For example, Fig. 5(a) shows the model’s average accuracy $\bar{Acc}$, and we observe that $\bar{Acc}$ is better than 0.979 for $\gamma \geq 5$. Figure 5(b) shows the cumulative probability of the prediction error $E$ for all structures in each dataset. We observe that, at $\gamma = 5$, the model has a prediction error $E < 0.02$ for more than 70% of the structures in the dataset. Because a conductivity contrast of at least 5 is widely encountered [e.g., matrix materials such as plastics have a conductivity of $\sim 0.1$ W/(mK) and most fillers have a conductivity of at least a few W/(mK)], the performance of the model is adequate under practical conditions.

When $\gamma$ decreases from 5 to 2, $\bar{Acc}$ decreases from 0.979 to 0.965. Such a reduction of $\bar{Acc}$ by 0.014, which corresponds to an increase in the error by 68%, is often considered a significant decline for deep learning models. When $\gamma$ decreases further to 1.1, the model’s predictive power decreases sharply: its $\bar{Acc}$ drops to 0.87; its prediction error is larger than 0.08 for virtually all structures in the dataset and is between 0.12 and 0.24 for more than 50% of the structures.

FIG. 4. [(a)–(c)] The temperature field (a), true binary thermal conductivity field (b), and the predicted binary thermal conductivity field (c) in a representative composite. [(b) and (c)] The black (white) color marks the positions with a thermal conductivity of 4 (200) W/m K. (d) The histogram of the prediction error of the U-Net model.
The decreased model performance as \( \gamma \) is lowered can be understood as follows. If the conductivity is homogeneous in a composite, its temperature field obeys the Laplace equation (i.e., \( \nabla^2 T = 0 \)) everywhere. Essentially, the deep learning model is trained to approximate the correlations between the non-Laplacian distribution of the temperature field and the binary conductivity fields in the composite. We postulate that if the non-Laplacian nature of the temperature field is strong, then the model should be able to learn to approximate these correlations and perform well; otherwise, the model will perform poorly. To test this postulation, we characterize the non-Laplacian nature of the temperature field by defining a parameter \( \omega \),

\[
\omega = \frac{1}{N} \sum_{i=1}^{N} \iint |\nabla^2 T| dA,
\]

where the integration is performed over each composite \( i \) and \( N \) is the number of composites in the dataset. As shown in Fig. 5(a), \( \omega \) decreases as \( \gamma \) is reduced, especially when \( \gamma \) reduced from 2 to 1.1. This trend corresponds very well to the evolution of the average accuracy with \( \gamma \) shown in the same figure, thus, supporting our postulation. In the following sections, simulations are all performed under fixed conductivity contrast \( \gamma = 50.0 \).

### B. Deep learning model with coarse-grained inputs

The above deep learning model uses a high-resolution temperature field in the composite as input (128 \( \times \) 128 pixels in the composite considered here). In practice, measuring high-resolution temperature fields can be time-consuming and costly, both when preparing the training dataset and when applying the trained model. Here, we test the performance of the U-Net model when coarse-grained temperature fields are used as input. Specifically, for each training dataset prepared in Sec. II, which features the temperature at 128 \( \times \) 128 pixels [see, e.g., Fig. 6(a)], the temperature at 12 \( \times \) 12 pixels uniformly distributed in the 2D composite [see Fig. 6(c)] is extracted. The measurement effort is, thus, reduced to 12\(^2\)/128\(^2\) (i.e., <1\%) of the original models. Each training, validation, and testing dataset now contains the coarse-grained temperature field and the original binary thermal conductivity field given at 128 \( \times \) 128 pixels.

Since we seek to predict the binary thermal conductivity field with high resolution, we first up-sample the coarse-grained temperature field in all datasets to 128 \( \times \) 128 pixels uniformly distributed in the composites. Next, a U-Net model with the same architecture, hyperparameters, and training protocol as those in Sec. IV A is adopted and trained. Finally, using the coarse-grained temperature field in the testing dataset as input, we evaluate the performance of the trained model. Note that because the true temperature field is available only at 12 \( \times \) 12 pixels both during the training and deployment of our U-Net model, the above scheme does allow us to evaluate the performance of deep learning models using coarse-grained input.

We use the Laplace interpolation for the upsampling of the input temperature field. The Laplace interpolation method has been widely used to recover damaged images. For an arbitrary variable \( y \) defined in a 2D space, whose value \( y \) is known at points \( x_i \) (\( i = 1 \) to \( N \)), its values at other points are obtained by solving the Laplace equation \( \nabla^2 y = 0 \) under the condition \( y(x_i) = y_i \). The Laplace interpolation method is selected for two reasons. On the one hand, because the temperature to be interpolated is governed by an equation similar to the Laplace equation [see Eq. (1)], such an interpolation method may perform well in up-sampling the temperature field. On the other hand, the jump of thermal conductivity at base matrix–filler interfaces means that the Laplace interpolation necessarily introduces errors into the temperature field, and it is interesting to investigate whether these errors will compromise the predictions of the deep learning model.

Figure 6(e) shows the binary thermal conductivity field predicted by our U-Net model when the temperature field shown in Fig. 6(d) is used as the input. The predicted binary field closely resembles the true field shown in Fig. 6(b). The average prediction accuracy of the binary thermal conductivity field based on the temperature fields up-sampled using the Laplace interpolation method is 0.94 (see Fig. 3) over the whole testing data, slightly lower compared with the case in which the temperature field is available at

**FIG. 5.** Performance of the deep learning model with dataset featuring different conductivity contrasts between the filler and the matrix (\( \gamma = k_f/k_m = 1.1, 2, 5, \text{ and } 50 \)) to test the performance of the U-Net model when coarse-grained temperature fields are used as input. Specifically, for each training dataset prepared in Sec. II, which features the temperature at 128 \( \times \) 128 pixels [see, e.g., Fig. 6(a)], the temperature at 12 \( \times \) 12 pixels uniformly distributed in the 2D composite [see Fig. 6(c)] is extracted. The measurement effort is, thus, reduced to 12\(^2\)/128\(^2\) (i.e., <1\%) of the original models. Each training, validation, and testing dataset now contains the coarse-grained temperature field and the original binary thermal conductivity field given at 128 \( \times \) 128 pixels.
FIG. 6 [(a) and (b)] The temperature field (a) and true binary thermal conductivity field (b) in a representative composite. (c) The distribution of the points at which the true temperature is available as input to the U-Net model. [(d) and (e)] The temperature field up-sampled using Laplace interpolation (d) and the binary thermal conductivity field (e) predicted using the up-sampled temperature field for the composite shown in (d). (f) The histogram of the prediction error of the U-Net model using temperature field up-sampled from coarse-grained temperature fields by Laplace interpolation.

every pixel in the composite. Further inspection of the prediction error [see Fig. 6(f)] reveals that over 95% of the tested cases have an average error of less than 10%. Given that the true temperature is available only at less than 1% of the pixels in the composite, the performance of the deep learning model is remarkable.

The comparison of the predicted and true binary thermal conductivity field [see Figs. 6(b) and 6(e)] shows that the deep learning model can capture the curvature of fillers with a radius of ∼10 pixels, despite the fact that the true temperature field is only known at coarsely distributed points with a spacing of 12 pixels. This success is likely related to the way the geometrical features are extracted and represented in deep learning models. In the deep learning model, the geometrical features of the fillers are implicitly encapsulated in the spatial correlations in the temperature fields. The several layers of filters in the encoder model can extract the spatial correlations in the temperature field at many length scales, which allows the geometrical features of the binary thermal conductivity field (and, thus, the curvature of the fillers) to be resolved at length scales comparable to the spacing between the points at which the true temperature is known. Therefore, the deep learning model has some potential advantages over methods based on finite difference and finite element methods. In those methods, because the differential operator in Eq. (1) is discretized locally, the mesh size must be considerably smaller than the radius of curvature of the geometrical features in the computational domain to capture the temperature fields well. The ability of deep learning models to capture global (large scale) features without requiring fine-grained input should be explored more systematically in the future.

C. Deep learning model using boundary-only inputs

While the interpolation-based up-sampling allows the binary thermal conductivity fields in a composite to be predicted from coarse-grained temperature data, the deep learning model requires data measured inside the composite. This kind of measurement is often more difficult to implement than measuring the temperature on boundaries, especially in three-dimensional composites. To study if this limitation can be alleviated, here, we explore the situation when the temperature is only measured on the boundaries of the 2D composite. Specifically, the temperature is only measured at 96 points distributed evenly on the four boundaries of the composite shown in Fig. 1.

With the true temperature measured on boundaries, we first obtain the temperature inside the composite using the Laplace interpolation. The interpolated temperature field for a representative case is shown in Fig. 7(a) and the corresponding true temperature field is shown Fig. 6(a). The interpolated temperature fields deviate from the true temperature fields with an average relative error of $AE_{i} = 43\%$. The interpolated field does not reproduce the true field well and contours of fillers are hardly discernible in the interpolated temperature field. Figure 7(c) shows the binary thermal conductivity field predicted by the U-Net model. We observe that the model
can reveal the regions where the high-conductivity fillers are located. This suggests that the encoder model can capture large-scale spatial correlations in the imperfect temperature field, and the decoder model can use these correlations to reconstruct the location of the fillers to a large extent. Nevertheless, the deep learning model has difficulties delineating the filler–matrix interfaces, particularly, in the space where several fillers come close to each other [highlighted using blue ovals in Fig. 7(c)] and in the composite’s interior. This is consistent with the fact that the interpolated temperature field cannot capture the temperature variation at small scales and away from the boundaries. Over the entire testing dataset, the average prediction accuracy of the binary conductivity field is $Acc = 0.797$ (see Fig. 3), with over 95% of the cases having an average error of less than 25% [see Fig. 7(d)]. Given that the true temperature is available only on the boundaries and at ~0.5% of all pixels in the composite, the accuracy of the deep learning model seems to be reasonable.

D. Improve deep learning prediction by model-informed temperature measurement

For the composite shown in Fig. 7(b), Fig. 7(c) shows that the model’s prediction of the binary thermal conductivity field is relatively poor in the composite’s interior. This poor performance can be inferred from the model itself. The last layer of our model is a tanh layer, which outputs a $k_{tanh}$ at each pixel in the composite [see Fig. 7(e), for example]. Thresholding $k_{tanh}$ using 0 as the decision threshold gives the predicted binary thermal conductivity field. Therefore, $k_{tanh}$ can be used as a surrogate for the confidence of the predicted binary thermal conductivity; a large (small) deviation of $k_{tanh}$ from the decision threshold 0 corresponds to a prediction with high (low) confidence. In the grayscale representation of $k_{tanh}$ in Fig. 7(e), regions with color close to the perfect black and white are places where the deep learning model is “certain” about the local thermal conductivity; while the regions with color more intermediate to black and white (mostly in the composite’s interior) correspond to places where the deep learning model is “uncertain” about the local thermal conductivity. The latter regions in which the deep learning model has less confidence (or more uncertainty), indeed, correspond to the regions in which the predicted binary thermal conductivity has larger errors [cf. Figs. 7(b) and 7(c)].

The relatively poor performance of the deep learning model in the composite’s interior when the true temperature is available on its boundary and the strong correlation between the poor performance and the low model confidence revealed by the tanh layer’s output are generally true for all composites studied here. Specifically, we compute the average prediction error for pixel $i,j$ inside all composites in the testing dataset by $\bar{E}_{i,j} = \sum_{l=1}^{N} |k_{l} - \hat{k}_{l}|/N$. Figure 7(f) shows that $E_{ij}$ in the composite’s interior is, indeed, larger than that near its boundary. We also define model confidence for each pixel $i,j$ inside all composites in the testing dataset by $CF_{ij} = \sum_{l=1}^{N} |k_{tanh,l}|/N$, where $CF_{ij} = 1$ and 0 correspond to a confidence level of 100% and 0%, respectively. Figure 7(g) shows that the regions with low model confidence roughly coincide with the regions with a high prediction error [cf. Figure 7(f)].
The strong correlation between the large prediction error and the low model confidence at a pixel can be used to guide the placement of temperature sensors for improving the model performance and reducing the number of temperature measurements. For example, because the model confidence in the composite’s interior is low when the true temperature is only available at the boundary, we can introduce additional sensors in this region to potentially reduce the prediction error in this region. To explore this idea, in addition to the 96 boundary points used in Sec. IV C, we introduce 56 measurement points into the center region of the composite [see Fig. 8(a)]. We then compute the temperature field in the composite using the Laplace interpolation method. Figure 8(b) shows the interpolated temperature field, which deviates from the true temperature field [see Fig. 6(a)] with an average relative error $AE = 30\%$. The binary thermal conductivity field predicted based on this temperature field is presented in Fig. 8(c). When compared with the result in Fig. 7(c), the prediction is markedly improved, especially in the central part of the composite. For the whole testing dataset, we find that the average prediction accuracy is $Acc = 0.885$ (see Fig. 3), and over 90% of the predictions have an error of less than 15% [see Fig. 8(d)]. Apart from this, the model confidence and the average prediction accuracy in the central region of the composite are improved from $\sim 55\%$ to $\sim 80\%$ and from $\sim 0.65$ to $\sim 0.85$, respectively [see Figs. 8(e) and 8(f)]. Therefore, measuring the temperature at additional points in the composite based on feedback from the deep learning model (i.e., the distribution of the model confidence across the composite) helps in the improvement of the model’s performance. The accuracy of the model is slightly lower than the model in case 2 shown in Fig. 3, where the true temperature is measured at 144 points inside the composite. However, the number of temperature measurements inside the composite is about half of that in case 2.

V. DISCUSSION

The results in Sec. IV show that, for composites featuring circle-shaped fillers randomly distributed in a 2D matrix, deep learning models can predict a composite’s structure from its temperature fields with reasonable accuracy. Therefore, although deep learning models for the quantitative solution of engineering problems are still in their infancy, they show considerable promise. The practical application of the deep learning models, however, must tackle several important problems not addressed here, e.g., they should be extended to three-dimensional (3D) systems with more realistic heterogeneity, and sufficient training data must be generated to train these models. We are cautiously optimistic that these problems can be solved.

First, the deep learning model can likely be extended to 3D systems. Recently, exciting progress has been reported in several papers on predicting the transport property of 3D porous media from their structure using CNN. These works suggest that deep learning
techniques are promising for solving 3D problems. Besides, U-Net models are good at mapping between high-dimensional data, and the feasibility of using it in 3D multi-class segmentation problems has been demonstrated. Second, a deep learning model should be able to handle more general heterogeneity, e.g., composites in which arbitrarily shaped fillers are dispersed in a base matrix. Indeed, we have tested a dataset in which circular fillers can overlap with each other, which produces partially irregular fillers [see Fig. S1(a)]. The model’s average prediction accuracy for this dataset is $\text{Acc} = 0.972$, which is similar to the situation where no overlap between fillers exists. The good prediction accuracy here suggests that the deep learning models can potentially deal with heterogeneity in a relatively general sense. However, this must be checked more thoroughly (especially in 3D systems) in the future. Extensive tests must be performed to confirm the feasibility of the U-Net model in solving inverse problems involving realistic, 3D heterogeneous structures.

Second, generating a large dataset for training a deep learning model should be feasible in many situations. Although building a large dataset experimentally can be challenging, this does not necessarily prevent the application of deep learning models. First, in many fields related to transport phenomena (e.g., petroleum engineering), massive datasets are generated experimentally on a daily basis. Some of these datasets are available to the public, e.g., the digital rock database. Second, large datasets can be built computationally. Realistic composite structures can be generated not only experimentally but also computationally. For many problems, a reliable mathematical model is available (heat conduction equation in our study). Because only forward problems (predicting temperature field from structure) must be solved when building a dataset, a massive dataset can be built by solving such problems using matured algorithms and high-performance clusters. Datasets thus built can be used to train the deep learning model. While care must be taken when using these datasets (e.g., ensuring the composite structure in the dataset is representative of the structure in real composites), they do greatly alleviate the difficulty associated with generating data by experiments.

VI. CONCLUSIONS

In summary, we develop a deep learning-based model to predict the distribution of high-conductivity fillers in a low-conductivity 2D matrix from the temperature field within their composite. By adopting the encoder–decoder architecture, our U-Net model handles the high-dimensional input and output fields effectively. When the true temperature at every pixel in the composite is used as the input, the model accurately predicts the filler distribution in the composite. Even when coarse-grained temperature fields (<1% of the full data) are used as the input, the model still performs reasonably well; the model’s performance can be improved further if the temperature data in selected regions, in which the model has low prediction confidence, are introduced as an extra input.

While the present work deals with an inverse problem involving heat conduction, conceivably similar models can be developed for other transport phenomena involving governing equations (e.g., fluid transport in heterogeneous media with non-uniform permeability). More generally, this work demonstrates the feasibility of solving inverse problems with high-dimensional input and output using deep learning models. Importantly, the geometrical features of fillers in the composite can be predicted with a spatial resolution slightly higher than the spatial resolution of the true temperature field. This suggests that the encoder sub-model can effectively capture the large-scale features of the temperature field despite the uncertainty (noise) at fine scales, and the decoder sub-model can map these features to the binary thermal conductivity field effectively. Since only coarse-grained input fields are available for solving many practical inverse problems, the results shown here are encouraging for the future development of deep learning models for these problems.

SUPPLEMENTARY MATERIAL

See the supplementary material for sample results of predicting the structure of heterogeneous media with a structure more complex than in Fig. 4 using deep learning models.

ACKNOWLEDGMENTS

We thank the ARC at Virginia Tech for the generous allocation of computer time.

The data that support the findings of this study are available from the corresponding author upon reasonable request.

REFERENCES

1. R. C. Aster, B. Borchers, and C. H. Thurber, *Parameter Estimation and Inverse Problems* (Elsevier, 2018).
2. A. Beiskos, A. Jasra, E. A. Muzaffer, and A. M. Stuart, “Sequential Monte Carlo methods for Bayesian elliptic inverse problems,” *Stat. Comput.* 25(4), 727–737 (2015).
3. F. Ballio and A. Guadagnini, “Convergence assessment of numerical Monte Carlo simulations in groundwater hydrology,” *Water Resour. Res.* 40(4), W0460, https://doi.org/10.1029/2003wr002876 (2004).
4. X. Wu and T. Kozlowski, *Inverse Uncertainty Quantification of Reactor Simulation with Polynomial Chaos Surrogate Model* (Transactions of American Nuclear Society, New Orleans, LA, USA, 2016).
5. K. S. Reddy and S. Somasundaram, “An inverse method for simultaneous estimation of thermal properties of orthotropic materials using Gaussian process regression,” *J. Phys.: Conf. Ser.* 745, 032090 (2016).
6. C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, “Rethinking the inception architecture for computer vision,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (IEEE, 2016), pp. 2818–2826.
7. Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature* 521(7553), 436 (2015).
8. T. Young, D. Hazarika, S. Poria, and E. Cambria, “Recent trends in deep learning based natural language processing,” *IEEE Comput. Intell. Mag.* 13(3), 55–75 (2018).
9. R. Collobert and J. Weston, “A unified architecture for natural language processing: Deep neural networks with multitask learning,” in *Proceedings of the 25th International Conference on Machine Learning* (ACM, 2008), pp. 160–167.
10. L. Brunton, B. R. Noack, and P. Koumoutsakos, “Machine learning for fluid mechanics,” *Annu. Rev. Fluid Mech.* 52, 477–508 (2020).
11. H. Chang and D. Zhang, “Machine learning subsurface flow equations from data,” *Comput. Geosci.* 23(5), 895–910 (2019).
12. H. Wei, S. Zhao, Q. Rong, and H. Rao, “Predicting the effective thermal conductivities of composite materials and porous media by machine learning methods,” *Int. J. Heat Mass Transfer* 127, 908–916 (2018).
15V. Zhu and N. Zabaras, “Bayesian deep convolutional encoder–decoder networks for surrogate modeling and uncertainty quantification,” J. Comput. Phys. 366, 415–447 (2018).

16K. K. Tripathy and I. Bilionis, “Deep UQ: Learning deep neural network surrogate models for high dimensional uncertainty quantification,” J. Comput. Phys. 375, 565–588 (2018).

17L. Jin, H. Lu, and G. Wen, “Fast uncertainty quantification of reservoir simulation with variational U-Net,” arXiv:1907.00718 (2019).

18J. Wu, X. Yin, and H. Xiao, "Seeing permeability from images: Fast prediction with convolutional neural networks," Sci. Bull. 63(18), 1215–1222 (2018).

19S. Kamrava, P. Tahmasbi, and M. Sahimi, “Linking morphology of porous media to their macroscopic permeability by deep learning,” Transp. Porous Media 131, 427 (2019).

20H. Wu, W.-Z. Fang, Q. Kang, W.-Q. Tao, and R. Qiao, “Predicting effective diffusivity of porous media from images by deep learning,” Sci. Rep. 9(1), 20387 (2019).

21H. H. Kang, M. Kaya, and S. Hajimirza, “A data driven artificial neural network model for predicting radiative properties of metallic packed beds,” J. Quant. Spectrosc. Radiat. Transfer 226, 66–72 (2019).

22S. Mo, Y. Zhu, N. Zabaras, X. Shi, and J. Wu, “Deep convolutional encoder-decoder networks for uncertainty quantification of dynamic multiphase flow in heterogeneous media,” Water Resour. Res. 55(1), 703–728, https://doi.org/10.1029/2018wr023528 (2019).

23K. Manohar, B. W. Brunton, J. N. Kutz, and S. L. Brunton, “Data-driven sparse sensor placement for reconstruction: Demonstrating the benefits of exploiting known patterns,” IEEE Control Syst. 38(3), 63–86 (2018).

24N. B. Ericsson, L. Mathelin, Z. Yao, S. L. Brunton, M. W. Mahoney, and J. N. Kutz, “Shallow learning for fluid flow reconstruction with limited sensors and limited data,” arXiv:1902.07359 (2019).

25J. Gu, Z. Wang, J. Kuen, L. Ma, A. Shahroudy, B. Shuai, T. Liu, X. Wang, G. Wang, and J. Cai, “Recent advances in convolutional neural networks,” Pattern Recognit. 77, 354–377 (2018).

26I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning (MIT Press, 2016).

27S. W. Smith, The Scientist and Engineer’s Guide to Digital Signal Processing (New Scientist Ltd., 2001), Vol. 171, p. 47.

28V. Badrinarayanan, A. Kendall, and R. Cipolla, “SegNet: A deep convolutional encoder–decoder architecture for image segmentation,” IEEE Trans. Pattern Anal. Mach. Intell. 39(12), 2481–2495 (2017).

29C. Doersch, “Tutorial on variational autoencoders,” arXiv:1606.05908 (2016).

30A. Kendall, V. Badrinarayanan, and R. Cipolla, “Bayesian SegNet: Model uncertainty in deep convolutional encoder–decoder architectures for scene understanding,” arXiv:1511.02860 (2015).

31J. Long, E. Shelhamer, and T. Darrell, “Fully convolutional networks for semantic segmentation,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (IEEE, 2015), pp. 3431–3440.

32V. Dumoulin and F. Visin, “A guide to convolution arithmetic for deep learning,” arXiv:1603.07285 (2016).

33O. Ronneberger, P. Fischer, and T. Brox, “U-Net: Convolutional networks for biomedical image segmentation,” in International Conference on Medical Image Computing and Computer-Assisted Intervention (Springer, 2015), pp. 234–241.

34P. Esser, E. Sutter, and B. Ommer, “A variational U-Net for conditional appearance and shape generation, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (IEEE, 2018), pp. 8857–8866.

35P. Dollar, R. Appel, S. Belongie, and P. Perona, “Fast feature pyramids for object detection,” IEEE Trans. Pattern Anal. Mach. Intell. 36(8), 1532–1545 (2014).

36M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, and M. Isard, “Tensorflow: A system for large-scale machine learning,” in 12th Symposium on Operating Systems Design and Implementation (OSDI ’16) (USENIX, 2016), pp. 265–283.

37D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” arXiv:1412.6980 (2014).

38L. Hoeltgen, S. Setzer, and J. Weickert, “An optimal control approach to find sparse data for Laplace interpolation,” in International Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition (Springer, 2013), pp. 151–164.

39Q. Rong, H. Wei, X. Huang, and H. Bao, “Predicting the effective thermal conductivity of composites from cross sections images using deep learning methods,” Compos. Sci. Technol. 184, 107861 (2019).

40Z. Yang, Y. C. Yabansu, R. Al-Bahrani, W.-K. Liao, A. N. Choudhary, S. R. Kalidindi, and A. Agrawal, “Deep learning approaches for mining structure-property linkages in high contrast composites from simulation datasets,” Comput. Mater. Sci. 151, 278–287 (2018).

41C. Wang, T. MacGillivray, G. Macnought, G. Yang, and D. Newby, “A two-stage 3D U-Net framework for multi-class segmentation on full resolution image,” arXiv:1804.04348 (2018).