Spectrum of Engineering Sciences
ISSN (e) 3007-3138 (p) 3007-312X
Volume 3, Issue 8, 2025

AN AI-POWERED HIERARCHICAL DEEP NEURAL NETWORK (HIDENN) APPROACH FOR COMPUTATIONAL SCIENCE AND ENGINEERING"

Kinza Urooj[*1], Sumayya Bibi[2], Urooj Fatima[3], Waqas Arif[4], Larib Fatima[5], Asad Riaz[6], Saad Khan Baloch[7], Umm e Habiba[8]

[*1]*Department of Mathematics, Air University Islamabad, Pakistan*
[2]*Department of Communication Engineering , Faculty of Electrical Engineering, Universiti Teknologi Malaysia 81310 UTM, Johar Bahru, Johar, Malaysia.*
[3]*Department of Software Engineering, Faculty of Computing,The Islamia University of Bahawalpur,63100, Pakistan*
[4]*Department of Electrical Engineering Technology, National Skills University Islamabad, Pakistan*
[5]*Department of Application Development, Stewart Title Pakistan*
[6]*Department of Mechanical, Energy, Management and Transportation Engineering, School of Polytechnic, University of Genova, 16145, Itlay*
[7] *Department of Electrical Engineering, Isra University Hyderabad, Sindh, Pakistan*
[8]*Department of Mathematics, The Islamia University of Bahawalpur,63100, Pakistan*

[*1]kinzaurooj789@gmail.com

DOI: https://doi.org/10.5281/zenodo.16886331

Keywords
Deep learning, Machine Learning, Reduced order model, Data-driven discovery, Multiscale simulation, Artificial Intelligence

Article History
Received: 15 May, 2025
Accepted: 02 July, 2025
Published: 16 August, 2025

Corresponding Author: *
Kinza Urooj

Abstract
In this work, a unified AI-framework named Hierarchical Deep Learning Neural Network (HiDeNN) is proposed to solve challenging computational science and engineering problems with little or no available physics as well as with extreme computational demand. The detailed construction and mathematical elements of HiDeNN are introduced and discussed to show the flexibility of the framework for diverse problems from disparate fields. Three example problems are solved to demonstrate the accuracy, efficiency, and versatility of the framework. The first example is designed to show that HiDeNN is capable of achieving better accuracy than conventional finite element method by learning the optimal nodal positions and capturing the stress concentration with a coarse mesh. The second example applies HiDeNN for multiscale analysis with sub-neural networks at each material point of macroscale. The final example demonstrates how HiDeNN can discover governing dimensionless parameters from experimental data so that a reduced set of input can be used to increase the learning efficiency. We further present a discussion and demonstration of the solution for advanced engineering problems that require state-of-the-art AI approaches and how a general and flexible system, such as HiDeNN-AI framework, can be applied to solve these problems..

1. INTRODUCTION

With the great development of modern computer and computational algorithms, computational science and engineering have achieved enormous success in almost all fields, such as physics, chemistry, biology, mechanical, civil,and materials science and engineering. However, many problems in computational science across the disciplines are still challenging. We propose that there are three major classes, or *types*, of problems puzzling the community of computational science and engineering.

These three types are:

1. **Type 1 or purely data-driven problems**: The class of analyses with unknown or still developing governing physics but abundant data. For these problems, the lack

of knowledge of physics can be compensated by the presence of considerable data from carefully designed experiments regarding the system response.

2. **Type 2 or mechanistically insufficient problems with limited data:** The term *mechanistic* refers to the theories which explain a phenomenon in purely physical or deterministic terms [1]. Type 2 problems are characterized by physical equations that require complementary data to provide a complete solution.

3. **Type 3 or computationally expensive problems:** The problems for which the governing equations are known but too computationally burdensome to solve. We will attempt to show that artificial intelligence (AI), particularly a subset of AI, deep learning, is a promising way to solve these challenging problems. An AI system is identified by its capability to perform tasks which currently humans perform in a better way [2]. This is famously judged by the Turing test, proposed to measure the intelligence of a machine by its capability to imitate human behavior [3]. An AI system can be classified into three classes, a) "weak" or narrow AI, b) general AI, and c) super AI [4]. A narrow or "weak" AI is designed to perform a specific task and outperform any human in doing that. General AI refers to an AI system that may exhibit intelligent behavior in different areas and might outperform humans [5]. Super AI is a conceptual version of the technology that is the supreme point where machine achieves superhuman intelligence and can perform abstract thinking [6]. Almost all of the AI systems we see around us fall in the category of narrow AI. Super and general AI are still futuristic ideas. Machine learning (ML) is a form of narrow AI [7] and defined as the process by which computers, when given data, create their own knowledge (hence the term learning) by identifying patterns in data [8, 9]. Deep neural network is a subset of machine learning tools by which computers "understand" challenging and complex concepts by building the deep hierarchy of simpler concepts [9]. A generic deep neural network consists of input layer, hidden layers, and output layer where the input (layer) is connected (nonlinear information processing) through an activation function (hidden layer) to the output (layer) [8].

There is a growing tendency across the scientific communities to engage narrow AI (machine learning or deep learning) to solve problems in disciplines such as mechanics [10, 11, 12], biology and bio-medicine [13, 14, 15], materials science and engineering [16, 17, 18], manufacturing process monitoring [19, 20, 21], topology optimization [22, 23, 24], design under uncertainty [25], and miscellaneous engineering disciplines [26, 27, 28]. The scope of machine learning tools to aid or solve computational science problems goes beyond merely regressing non-linear data. Deep neural network and transfer learning are now being applied to discover hidden governing physical laws from data [29, 30, 31], speed up the computation in multiscale and multiphysics problems [32, 33, 34, 35, 36], characterize and reconstruct complex microstructures [37], design of heterogeneous materials and metamaterials [38, 39], discover new materials [40, 41], and to model path- and history-dependent problems [42, 43, 44, 45]. Figure 1 shows AI tools currently in use to solve state-of-art computational science problems. The AI tools include data generation and collection techniques, feature extraction techniques (wavelet and Fourier transform [46], principal component analysis [46]), dimension reduction techniques (clustering, self-organizing map [21, 46]), regression (neural network, random forest) [46], reduced order models (can be something similar to regression techniques or more advanced technique like self-consistent clustering analysis (SCA) [47, 48] or Proper Orthogonal Decomposition (POD) [46]) and classification (convolutional neural networks or CNN [46]).

There are several practical challenges in directly applying current AI frameworks to solve aforementioned types of problems: 1) it is often difficult to decide on the criteria to identify the type of the problem and on the set of toolsto use; 2) for a computational materials scientist or practicing engineer, it might become a challenging task to go backand forth among the different machine learning tools; 3) a design engineer needs to have a closed form relationshipamong different parameters controlling the desired property of the system. Moreover, the bridge connecting seemingly disparate fields of data-science and computational methods has to be a general one so that a common framework can be used to solve problems of different nature and originating from different physics. One other problem for applying AI frameworks in science and engineering is the paucity of data. Often experiments are too expensive to be useful to generate a large amount of data. Computational and theoretical predictions are limited by inherent assumptions. Considering these current difficulties and constraints, we propose a unified deep learning framework named Hierarchical Deep Learning Neural Network

(HiDeNN). An advantage of using the HiDeNN structure is that such a neural network has a universal approximation capability enabling it to correctly interpolate among the data points generated by extremely non-linear relationships. HiDeNN can identify the governing physics from an experimental dataset without any prior knowledge and therefore can fill the missing link between data and mechanistic knowledge. As will be explained, HiDeNN also has the capability to incorporate mechanistic knowledge in training through proper definition of the loss function. This will reduce the necessity of a large amount of data to get an accurate prediction. A practical example is provided in a companion paper by Tajdari et al. [49], submitted to the same issue, to demonstrate in detail how a small amount of medical data available for adolescent idiopathic scoliosis can be used with mechanistic knowledge and deep learning to predict spine curvature. All the machine learning tools and computational methods mentioned earlier can be built into HiDeNN, eliminating the need for the user to decide on specific tools. This article is organized as follows: section 2 introduces and describes the components of HiDeNN, section.

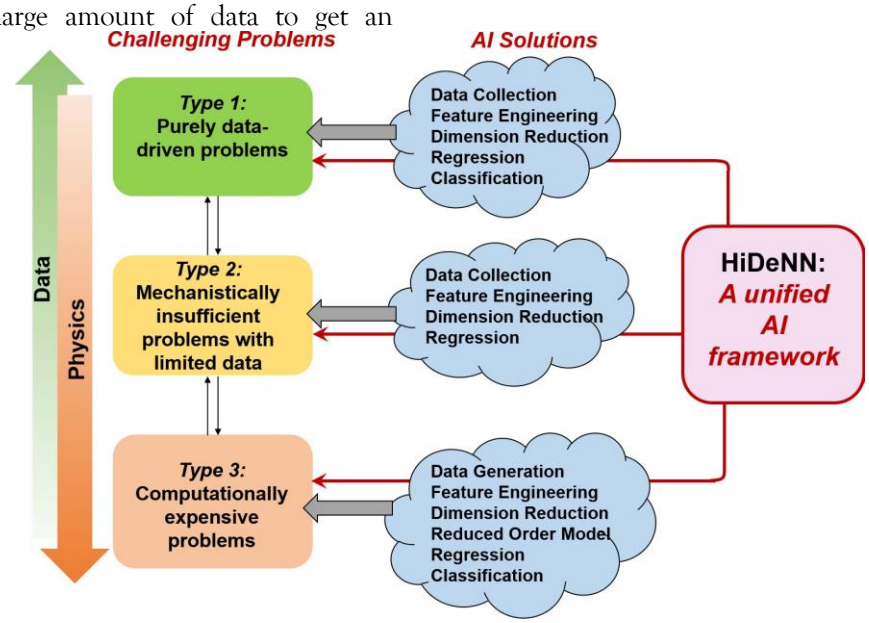


Figure 1: A comparative picture of state-of-the-art AI tools in computational science and engineering and the proposed Hierarchical Deep Neural Network (HiDeNN) framework. HiDeNN offers the advantage of being a unified framework to solve the problem in computational science and engineering without resorting to different set of tools for different types of problem.

3 presents the application of HiDeNN framework by solving three illustrative problems, section
4 discusses three examples from each *type* of challenging problems, how those are solved using state-of-the-art methods, and recast the solution of the problems using HiDeNN, section
5 proposes possible extensions of HiDeNN for general problems.

2. Hierarchical Deep Learning Neural Network (HiDeNN)

An example structure of HiDeNN for a general computational science and engineering problem is shown in Figure 2. Construction of HiDeNN framework is discussed in following points:

- The **input layer** of HiDeNN consists of inputs from spatial (Ω), temporal (t), and parameter (D) spaces. The neurons of this layer serve as independent variables of any physical system.
- The input layer of HiDeNN is connected to a set of neurons that represents a set of **pre-processing functions** $f(\mathbf{x}, t, \mathbf{p})$ where $\mathbf{x}$, t, and $\mathbf{p}$ are position, time, and parameter vector, respectively. These functions can be thought of as tools for feature engineering. For example, the pre-processing functions can convert dimensional parameters into dimensionless inputs. Such conversion can be necessary for fluid mechanics problems where,

for example, the Reynolds (Re) number is important.

• The layer of the pre-processing functions is connected to **structured hierarchical deep learning neural networks** (**DNN**). Hierarchical DNN layers consist of parameter layers, customized physics-based neural networks **PHY-NN**) or experimental-data based neural network (**EXP-NN**). In Figure 2, the indices i and j indicate similar neural networks layers can be appended for both PHY-NN and EXP-NN, respectively. The **PHY-NN** refers to a neural network formulated from physics-based data and **EXP-NN** neural network is designed from experimental data.

• In the hierarchical DNNs portion of the HiDeNN (see Figure 2), we see multiple **sub-neural networks** connected (the red blocks). We define the sub-neural networks as stand-alone neural networks that can provide input the PHY-NN or EXP-NN. This multi-level structure is the source of the name "Hierarchical" in HiDeNN.

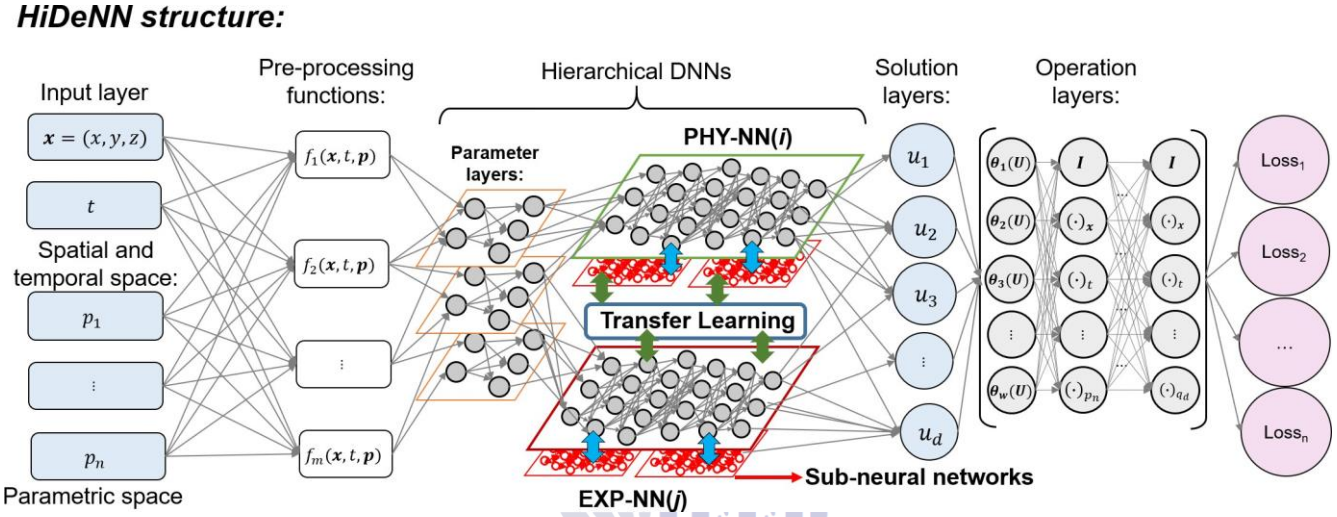


Figure 2: Detail construction of the proposed HiDeNN framework. The input layer takes in space, time, and parameter variables of a system. The input layer is connected to the pre-processing function, Hierarchical DNNs, and finally the solution layer. Governing equations can be obtained from solution layers through the operation layer and the loss function.

• The Hierarchical DNNs can be any type of neural network, including convolutional neural network (CNN), recurrent neural network (RNN), and graph neural network (GNN). In order to enhance the capability of PHY- NN or EXP-NN transfer learning technique can adopted in the proposed structure.

• Lack of data is of big concern in AI community. Available experimental data often come from dissimilar experimental or computational conditions making them hard to use directly in an AI framework. As one means of dealing with the problem, HiDeNN has provision for transfer learning in the Hierarchical DNN layer. The PHY-NNs and EXP NNs can be trained separately with the available computational and experimental data.Later, these individual neural networks can be combined through transfer learning.

• The Hierarchical DNN layer is connected to the **solution layer**. The solution layer represents the set of dependent variables of any particular problem.

• To discover unknown governing equations from data, HiDeNN has **operation layers**. In this layer, the neurons are connected through weights and biases in a way that mimics the behaviour of different spatiotemporal operators. Through proper training (i.e. minimization of the **loss** function in the HiDeNN), the operation layer can be trained to discover hidden physics from data.

• The **loss function** layer of HiDeNN contains multiple loss function terms as shown in the figure. Each loss function can either come from the hierarchical DNNs or the operational layers. These functions can be opti-mized simultaneously or separately depending on the problem. This unique feature of the HiDeNN provides the flexibility to solve problems with scarce and abundant data by combining it with physics.

3. Application of HiDeNN Framework

In this section, three examples of HiDeNN are discussed in detail to demonstrate the framework's capability.

3.1. HiDeNN for learning the discretization

In this example, we will use the HiDeNN to solve a solid mechanics problem and capture stress concentration by training the position of the nodes used for the discretization to minimize the potential energy of the system. In HiDeNN, interpolation function for approximating the solution is obtained by constructing neural network and training the weights and biases simultaneously [50]. Figure 3 presents a 2D bi-linear HiDeNN element, $N(x, y; \boldsymbol{w}, \boldsymbol{b}, \boldsymbol{A})$ at node (x_I, y_J), constructed by using the well-defined building blocks proposed in [50]. The arguments $\boldsymbol{w}$, $\boldsymbol{b}$, and $\boldsymbol{A}$ are the weights, biases, and activation function of the neural networks. Here, both $\boldsymbol{w}$ and $\boldsymbol{b}$ are functions of nodal positions. Therefore, updating the weights and biases during training implies updated nodal coordinates. The inter- polation function at (x_I, y_J) can be expressed as $N(x, y; x^*, y^*, \boldsymbol{A})$ where x^*, y^* are the updated nodal positions. As illustrated in the Figure 3, the inputs of the HiDeNN element are the nodal coordinates, (x, y), while the output are the nodal displacements, $\boldsymbol{u}_x$ and $\boldsymbol{u}_y$. When the nodal positions are fixed, HiDeNN is equivalent to standard FEM, 120 while when the nodal coordinates,x^*, y^*, in the weights and biases are updated during training, HiDeNN is able to I J accomplish better results like the r-adaptivity in FEM. However, differentiating from stress-based error estimator in r-adaptivity, the proposed HiDeNN method updates the nodal position by learning the performance through structured deep neural networks. The updated nodal coordinates will replace the original nodal coordinates after the training process until the optimal solution accuracy is achieved.

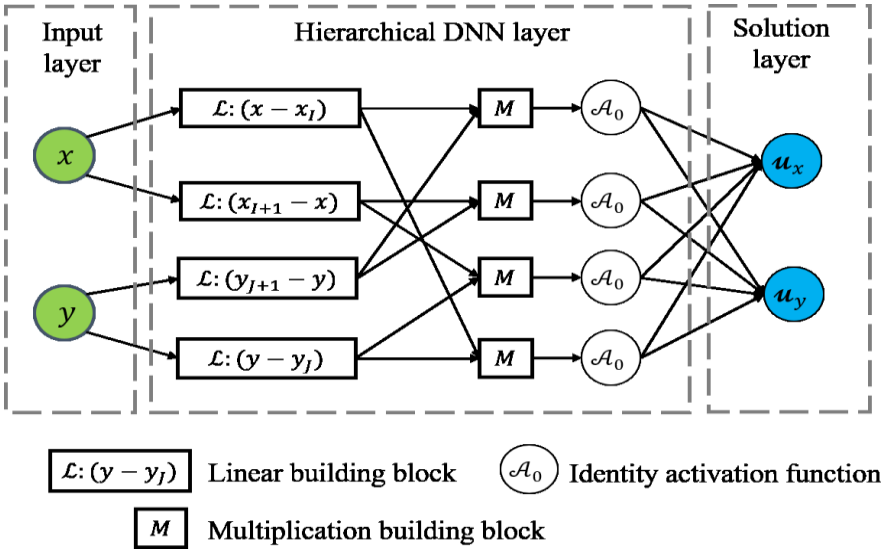


Figure 3: Construction of the bi-linear HiDeNN element at nodal (x_I, y_J) using building blocks proposed in [50]. The input of the unit neural network is nodal coordinates (x, y), while the output is the nodal displacements (u_x, u_y). Note that the weights and biases in the above neural network are functions of nodal positions (x^*, y^*). The training of the neural network is equivalent to find the optimally nodal positions to achieve I J optimal performance for the loss function in Eq.1a.

By assembling the HiDeNN elements, a unified neural network is formed to solve any problem of interests. Figure 4 shows the assembled neural network for a 2D problem. In the operations layer, the neuron $f_1(\cdot)$ is used to formulate the Neumann boundary conditions while the Dirichlet boundary condition is automatically satisfied through the optimization of the loss function. The weights of green arrows in the Figure 4 represent the constitutive model, in this case, the stiffness matrix for an elastic problem. The neural network in Figure 4 is a variation of the HiDeNN framework in Figure 2 for solving the problems with known governing equations. The

input is the spatial coordinates, the PHY-NN is the construction of the interpolation function and the solution is the displacements. The operation layer is used to define the loss function (total potential energy) given in Eq. 1a. For more detail please refer to [50]. Here the HiDeNN method is implemented in PyTorch v1.6.0, and the training

$$L(\boldsymbol{u}^h; \boldsymbol{f}, \boldsymbol{t}) = \frac{1}{2}\int_\Omega \sigma(uh) : \varepsilon(\boldsymbol{u}^h)d\Omega - \int_\Omega \boldsymbol{u}^h \cdot \boldsymbol{f}d\Omega + \int_{\Gamma_t} \boldsymbol{u}^h \cdot \bar{\boldsymbol{t}}d\Gamma \quad (1a)$$

$$\boldsymbol{u}^h(x, y; \boldsymbol{x}^*, \boldsymbol{y}^*, \boldsymbol{A}) = \sum_{n=N} N_n(x, y; x^*, y^*, \boldsymbol{A})u_n \quad (1b)$$

where $\boldsymbol{u}^h$ is the displacement field, $\boldsymbol{x}^*$ and $\boldsymbol{y}^*$ are the vector used to save the nodal positions, N is the total number of nodes, u_n and $N_n(x, y; x^*, y^*, \boldsymbol{A})$ denote the nodal displacement and interpolation function at node n. σ and ε are th n n stress and strain tensors, respectively, $\boldsymbol{f}$ is the body force, and $\boldsymbol{t}$ is the external traction applied on boundary Γ_t. For linear elastic problem, we have $\varepsilon = \frac{1}{2}(\nabla \boldsymbol{u} + (\nabla \boldsymbol{u})^T)$. Note that to avoid the inverted elements when the nodes are moved during training, a stop criterion for detecting a jump of the loss function is added. Inversion of an element will cause the loss function will increase suddenly, at which point the training will be stopped and the previous iteration will be taken as the final result.

$$\frac{L_{n+1} - L_n}{L_n}$$

where e_L denotes the change of the loss function between the neighbor iterations. When $e_L > 0.2$, the training is 1stopped and the previous iteration is taken as the results. To assess the method, we compare the computational cost of HiDeNN with the standard FEM. To do this, we fix the nodal positions during optimization similar to standard FEM. Under such conditions, HiDeNN solves a problem by minimizing a loss function, which is the potential energy of the structure for a mechanics problem, using a state-of- the-art optimizer (i.e. the Adam method [52]) available in most deep learning software packages. Figure 5 illustrates the problem used for the study. The test problem is an elastic material under simple tensile loading with four initial elliptical voids, solved under the plane stress condition. The domain of the test problem is a squa for the variables, such as nodal displacements and nodal positions, is performed by using the autograd package Paszke et al. [51] in PyTorch on a laptop with an Intel(R) Core(TM) i7-9750H @2.60 GHz and an NVIDIA GeForce RTX 2060 Graphics Processing Unit (GPU).

with dimensions 2 by 2 The displacement of left side of the domain is fixed while a uniform loading of F =20 is applied to the right side along the +x-direction. Young's modulus, E of the elastic material is 10^5, and the Poisson's ratio, u, is 0 3. The domain is discretized by conformal mesh with differing numbers of quadrilateral elements using Abaqus [53].

We consider several conformal meshes with an increasing number of degrees of freedom: 1154 , 2022 , 4646 , 8650 ,16 612 , 33 340 , 65 430 , 130 300 , 259 430 , 1 236 948 and 2 334 596 First, we solve the problem using Abaqus and the displacements at each node are later used as the reference for estimating the HiDeNN solution. Here, the $||e||_L1$error of the displacement defined in Eq. 3 is used for estimation. If $||e||_L1 < 10^{-6}$, the HiDeNN computations are

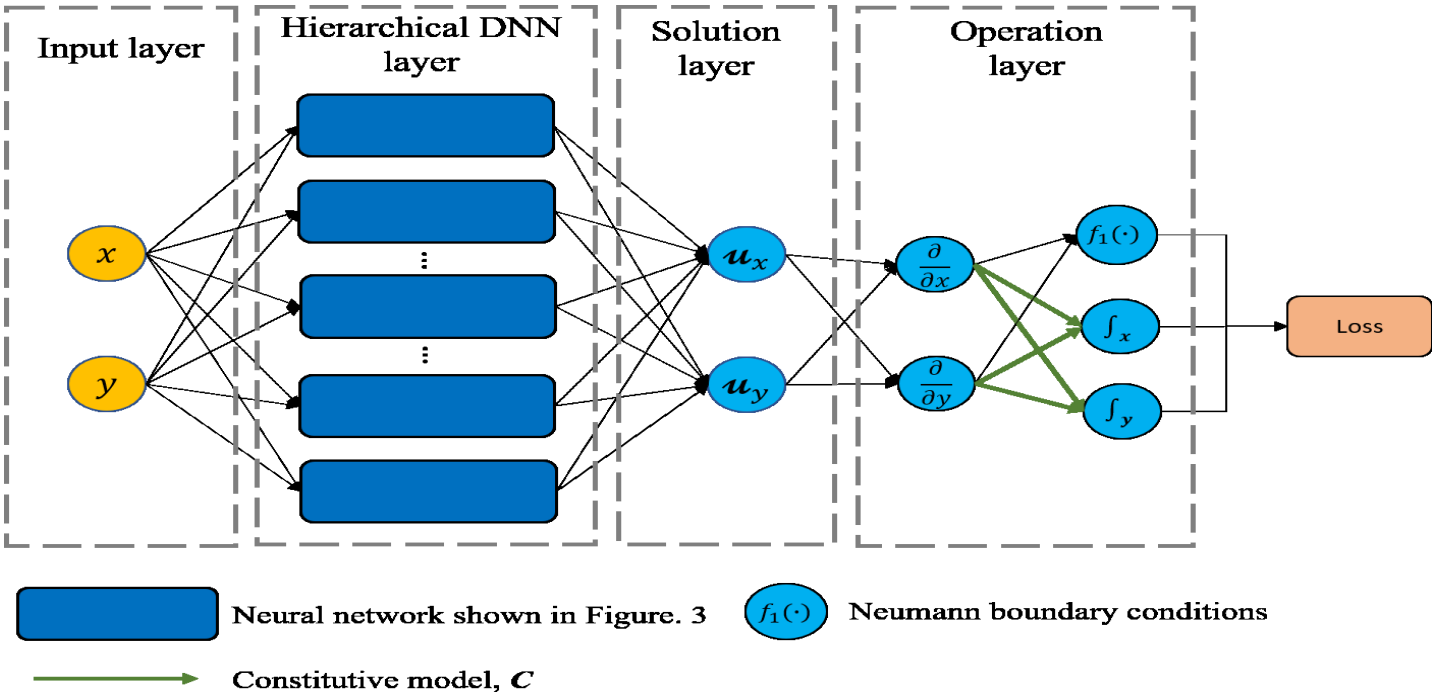


Figure 4: The assembled neural network by the unit neural network in Figure 3 for solving the 2D elastic problem. The input of the neural network is the nodal coordinates I. The output is the solution of the nodal displacements. The operation layer is used to formulate the governing equations. The loss function is defined in Eq.1a.

$$|u^{Abaqus}_I|$$

u_I is the displacement at node I obtained by Abaqus, while u_I is the corresponding value obtained by HiDeNN method. I is the index for the nodes in the domain, and n denotes the total number of nodes within the domain.

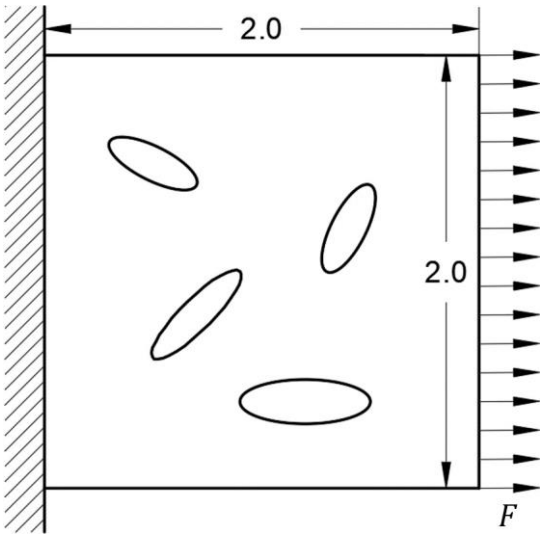


Figure 5: Schematic diagram of the test problem, a square domain with four initial elliptical voids. The dimensions of the square domain is 2 × 2. Young's modulus of the material is 10^5 and Poisson's ratio is 0.3. The left side of the domain is fixed while a uniform load of F = 20 is applied to the right-side of the domain.

The computational time of HiDeNN with respect to the degrees of freedom (DOFs) is plotted on logarithmic axes in Fig. 6. As can be seen, the computational cost of HiDeNN increases with the degrees of freedom (DOFs). It has an approximately linear relationship with the DOFs on the log-log plot with slope slightly larger than 1. This implies that computational cost increases more quickly than the number of degrees of freedom.

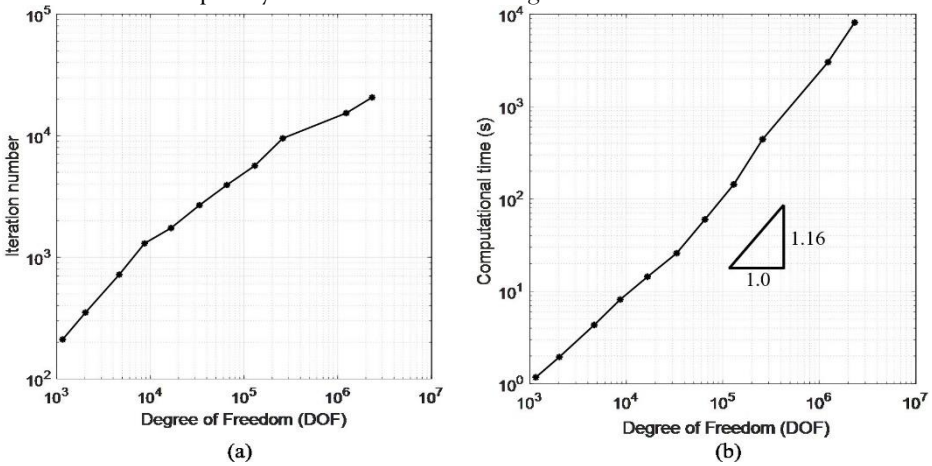


Figure 6: Computational iterations and time of HiDeNN with respect to the number of degrees of freedom. Computational time increases slightly faster than the degrees of freedom. (a) Iteration number versus degrees freedom, (b) Computational time versus degrees of freedom.

In order to show how the HiDeNN can "intelligently" capture the stress concentrations, we relax the nodal position constraints in the neural network and train the nodal positions and nodal displacements simultaneously. For comparison, a convergence study for the maximum local stress is conducted in Abaqus [53] with a convergence criterion of less than 1% change between subsequently more refined meshes. The converged mesh is taken as the reference solu-tion to examine the performance of the HiDeNN. The converged mesh and the stress distributions are given in Figs. 7(a), (b), and (c), respectively. The maximum local stress within the test geometry converges to 77.7 for a mesh with 3 867 168 quadrilateral elements and 7 748 156 DOFs. As illustrated in the figure, the maximum local stress occurs near the top corner of the bottom left ellipse. In order to capture the stress peak, an extremely fine mesh is required at this region when using standard FEM (refer to Figure 7(b).

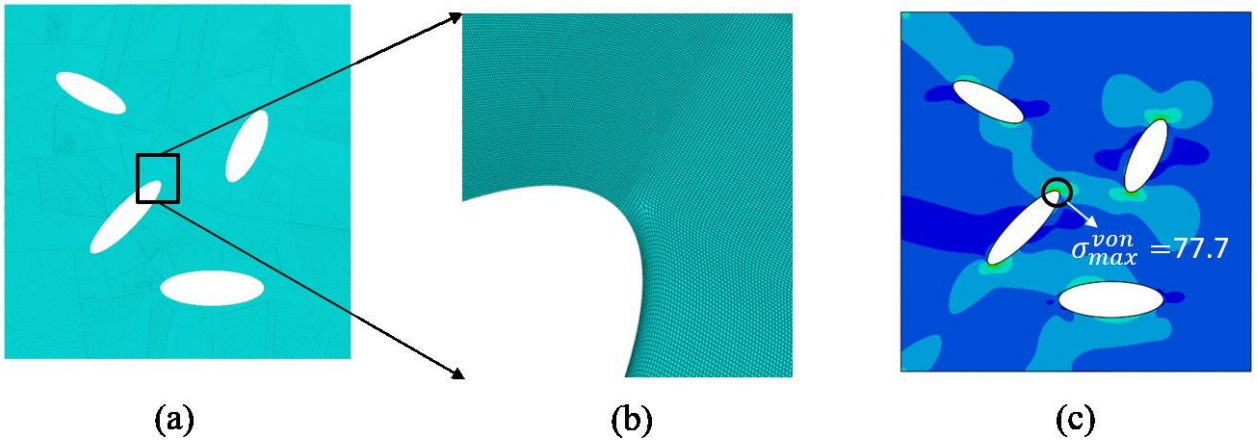


Figure 7: Converged conformal mesh and the corresponding FEM results (von Mises stress) for the test problem with four elliptical holes. (a) Full domain with four elliptical holes, (b) detail of the converged mesh near the stress concentration, (c) stress distribution inside the full domain for the converged solution in Abaqus.

For a one-to-one comparison between the FEM and HiDeNN solutions, the test problem is discretized with same conformal meshes as Abaqus. Four meshes are used, with 524 quadrilateral elements with 1154 DOFs, 938 quadrilateral elements with 2022 DOFs, 2194 quadrilateral elements with 4646 DOFs, and 4143 quadrilaterial elements with 8650 DOFs, as shown in Figure 8(e)-(h). Both HiDeNN and FEM are applied.

In HiDeNN, any parameter is treated as an input dimension in the parameter space of D, (refer to Figure 2). For functionally distributed fiber-reinforced composite analysis, the primary parameter is fiber volume fraction.

Using SCA, RVEs with different volume fractions are analyzed to form a training data set. Afterwards, a feed forward neural network (FFNN) with the parameter of fiber volume fraction as the input is trained to rapidly compute the microscale response. Figure 10 presents the FFNN for the microscale analysis. The inputs of the FFNN include macroscale strains and the fiber volume fraction and the outputs are the homogenized stress of the RVE model at macroscale integration points.

As a demonstration, 2D tensile bar under plane stress consisting of graded microstructure is modeled using the structured HiDeNN for two-scale analysis. Figure 9 shows the 2D model, the functional distribution of fiber-reinforced microstructures in a bar, as well as the microstructure of the RVE for different volume fractions. The dimension of the bar is 2.0 by 1.0 Fixed boundary conditions are applied at the left side of the bar and a uniform tension F of 10^5 is applied in the +x direction at the right side of the bar. The volume fraction of the fiber is distributed following the of $\rho(x, y) = (y - 0.5)\sin(8x)$; high volume fraction and low volume fraction RVEs are periodically distributed along the x direction. The aim is to induce multiple stress concentrations and investigate the performance the proposed methodology in detecting the stress concentration by learning the problem with HiDeNN. To examine the performance, the problem is solved with two different discretization strategies: one with 320 × 160 quadrilateral elements and another with 80 × 40 quadrilateral elements using the finite element method, with the material response with mesh of 80×40 four-nodes elements by training both the nodal displacements and nodal positions simultaneously.

To solve this problem, 30 different RVEs with volume fraction ranging from 0.19 to 0.59 are solved using SCA. It takes 2513 s to solve for each RVE and around 5 s to train the neural network. MATLAB is used for training.

The results of the HiDeNN model for the functional composite are given in Figure 11 including both the nodes and von Mises stress, σ^{von}, distributions. Due to the microstructure variation there are five stress concentrations at macroscale. Even with a coarse mesh, the HiDeNN network can obtain a result within 1% of the fine mesh solution obtained with FEM by moving nodes to stress concentration regions (0.480 for HiDeNN versus 0.481 for the fine mesh FEM solution).

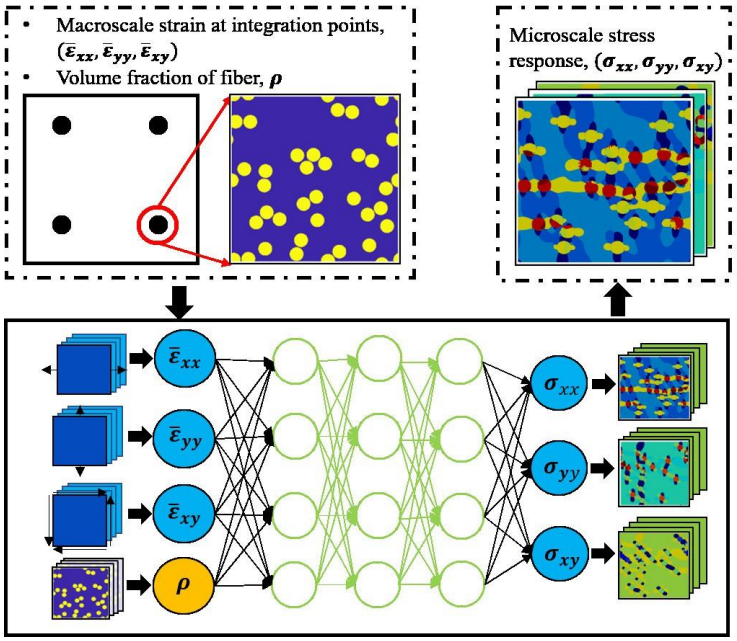
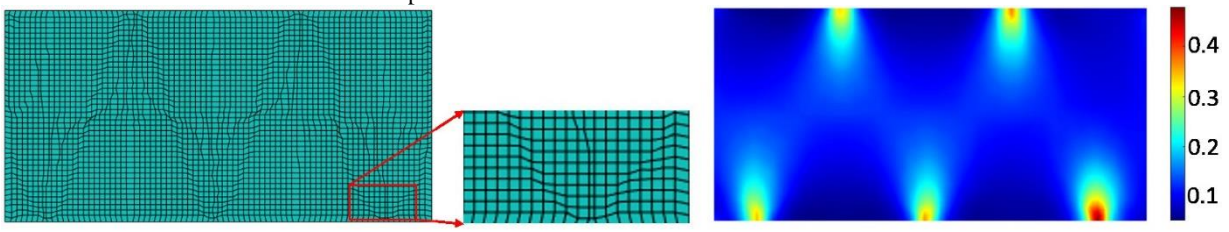


Figure 10: The sub-neural networks of performing the analysis for RVE model attributed to the interpolation points at microscale. The input of the sub-neural networks includes the macroscopic strain at integration points and the volume fraction of the fiber. The output of the neural networks are the microscale stresses of the RVE model.



(a) (b)

Figure 11: Discretization and stress distribution obtained by multiscale HiDeNN framework. (a) Final state of the discretization after learning the optimal nodal position by HiDeNN; (b) Stress distribution by using the HiDeNN to solve the functionally distributed fiber-enforced composite. The maximum stress is 0.480 by using HiDeNN analysis. The difference is 0.3% comparing with the converged value obtained by four times fine mesh while the difference from same uniform mesh is 9.71%.

3.3. HiDeNN for multivariate system: discovery of governing dimensionless numbers from data

The HiDeNN can handle data in a high-dimensional parametric space, $p_1 \sim p_n$ as shown in Figure 2. However, a large number of input parameters often causes two severe problems: first, the number of data required for training the network exponentially increases with the dimensionality of the inputs, i.e., the curse of dimensionality [58]; second, a large number of parametric inputs with complex dependencies and interactions could significantly degrade the capability of extrapolation and prediction of the network [59]. In order to reduce the dimensionality of the input parameters such that HiDeNN can be applied to a wide range of science and engineering problems, we proposea dimensionality of the original input parameters by automatically discovering a smaller set of governing dimensionless numbers and transforming the high-dimensional inputs to the dimensionless set. The DimensionNet can identify appropriate pre-processing functions and parameter layers for HiDeNN.

To illustrate the performance and features of the proposed DimensionNet we use it to "rediscover" well-known dimensionless numbers, e.g., Reynolds number (Re) and relative roughness (Ra*), in a classical fluid mechanics problem: laminar to turbulent transition in rough pipes [60, 61, 62, 63]. We use the experimental data collected by Nikuradse [61] to demonstrate that the proposed DimensionNet can recreate the classical governing dimensionless numbers and scaling law.

A schematic of turbulent pipe flow is presented in Figure 12. The dependent parameter of interest is the dimensionless resistance factor h that can be expressed as [61]

$$\frac{p_1 - p_2}{l}\frac{2d}{\rho U^2}, \tag{8}$$

where $p_1 - p_2$ represents the pressure drop from inlet to outlet, l is the length of the pipe, d is the diameter of the circular pipe, ρ is the density of fluid and U measures the average velocity over a steady-state, i.e., fully-developed, section of the pipe.

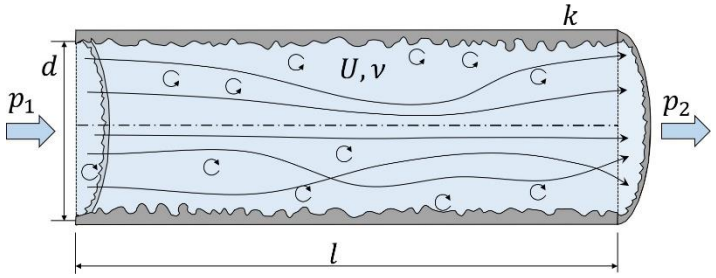


Figure 12: Schematic of the fluid flow in a rough pipe with dimensional quantities, including inlet pressure p_1, outlet

pressure p_2, pipe diameter d, pipe length l, average steady-state velocity U, kinematic viscosity u, and surface roughness Ra. We postulate that the resistance factor h depends on four parameters: the steady-state velocity of fluid U, kinematic viscosity u, pipe diameter d, and surface roughness of the pipe Ra: h = f(U, u, d, Ra).

We assume that there are only two governing dimensionless parameters in this system (the maximum number of the governing dimensionless parameters can be determined by dimensional analysis). To discover these two dimensionless combinations from the dataset, we take the experimental data [61] with various U, u, d, Ra as the four inputs of the DimensionNet, and log(100h) as the output to be consistent with the original results.,

Figure 13: Schematic of the dimensionally invariant deep network (DimensionNet). The four inputs are p_1 = U, p_2 = u, p_3 = d, and p_4 = Ra. The output is u = log(100h). The number of neurons at each layers depends on the applied problem. The network structure presented in this figure is just for illustration. The weights of the first layer $\mathbf{w}^{(1\,j)}$ can be predetermined from the dimensional matrix B, in which the rows are the dimensions and the columns are the input parameters. For example, the dimensional matrix B for the pipe flow problem is expressed as

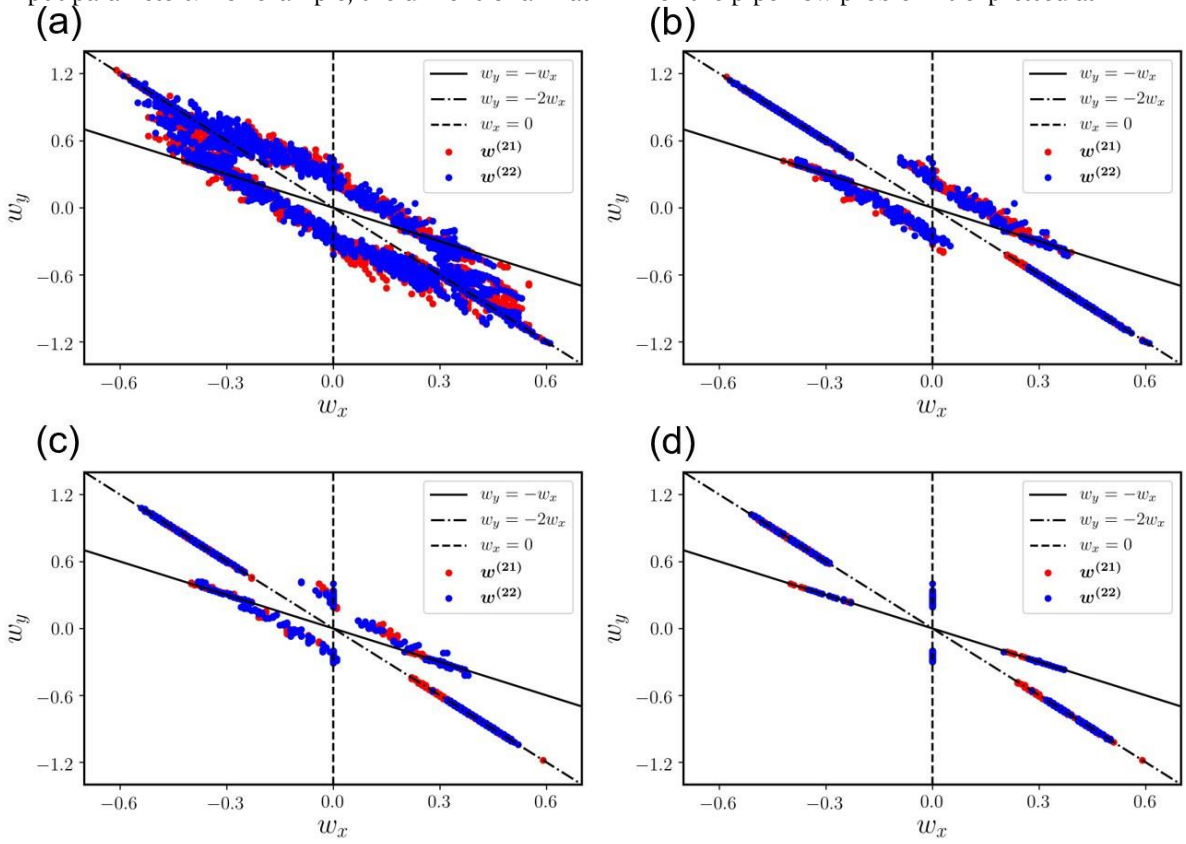


Figure 14: Distribution of the identified weights of the basis, i.e., $\mathbf{w}^{(21)}$ and $\mathbf{w}^{(22)}$ from snapshot results with high R^2 (greater than or equal to 0.98) and different training BIC thresholds: a) no BIC threshold; b) results with BIC≤ 0; c) results with BIC≤ −750; and d) results with BIC≤ −1500.

activation functions for the neurons at the second layer of the scaling network). They are the well-known Reynolds number and relative surface roughness [61]

$$\mathrm{Re} = \frac{\mathrm{Ud}}{\mathrm{u}} \tag{24}$$

$$\mathrm{Ra}^* = \frac{\mathrm{Ra}}{\mathrm{d}} \tag{25}$$

The scaling law or similarity function captured by the DimensionNet can be expressed as

$$\log(100h) = f\left(\log\left(\frac{Ud}{\nu}\right), \log\left(\frac{Ra}{d}\right)\right) = f(\log(Re), \log(Ra^*)) \quad (26)$$

The coefficients of determination R^2 are shown in Figure 15(a) for the training set and 15(b) for test set. The R^2 values are nearly 1, indicating the good predictive capability of the DimensionNet. Figure 15(c) shows the two-dimensional pattern hidden in the original four-dimensional parametric space, and this low-dimensional pattern is governed by two identified dimensionless numbers, i.e., Reynolds number Re and relative roughness Ra^*. In this study, we assume the number of governing dimensionless parameters is known, but it does not have to be known for a general problem. If we do not know the number of dimensionless parameters, we would start at one and train the DimensionNet and see if the network can converge to a highly accurate result. If so, we conclude that there is only one governing dimensionless number in the problem. If not, we will set up one more dimensionless parameter and re-train the DimensionNet. We will repeat this procedure until we find a converged result. In this way, we can identify the number of governing dimensionless number for a problem or a system without governing equations.

Traditionally, the dimensionless numbers are identified by dimensional analysis [66] or from normalized governing equations [67]. However, for many complex systems the optimal dimensionless numbers cannot be determined by using dimensional analysis alone [68], and for many applications we do not have well-tested governing equations of the problems or only know part of them. For those problems, we can alternatively use the proposed DimensionNet to discover the governing dimensionless numbers purely from data. The identified smaller set of dimensionless numbers informs HiDeNN such that it can predict more complex behaviors of the problems in a more accurate and efficient manner. The DimensionNet involves the principle of the similitude and dimensional invariance [67]. It can eliminate the physically inherent dependency between the dimensional input parameters without any loss of accuracy, and thus has better extrapolation capability than traditional dimensionality reduction method such as principal component analysis (PCA).

The proposed DimensionNet is a very general tool and thus can be applied to many other physical, chemical and biological problems where abundant data are available but complete governing laws and equations are vague [69]. The identified reduced parameter list can be used as the input to the HiDeNN. It can significantly improve the efficiency and interpretability of the network and avoid overfitting by reducing the input space and dependency.

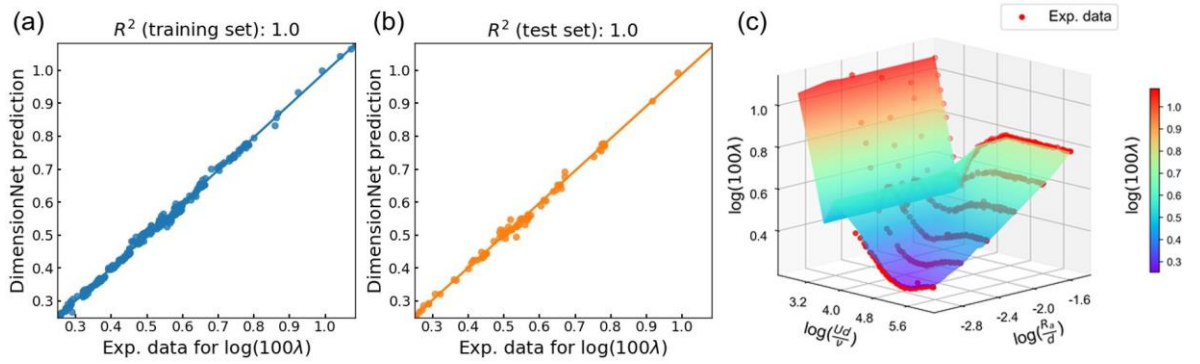


Figure 15: Comparison between experimental data and DimensionNet prediction: a) R^2 for the training dataset; b) R^2 for the test dataset; and c) captured relationship between identified dimensionless numbers and resistance factor. The points represent experimental data [61] and the surface represents the DimensionNet result.

4. Extension of HiDeNN to Solve Challenging Problems

This section demonstrates a typical AI solution method for one example of each type of the challenging problems introduced in section 1, and make note of challenges with these existing methods that might be mitigated by using HiDeNN.

4.1. Type 1: Purely data-driven problems

The case study involves finding the salient relationship between the local thermal history and ultimate tensile strength

in a thin wall built by direct energy deposition with Inconel alloy 718. In this case, we assume there is no known physical law connecting these two factors; thus, an AI/ML method is used to infer the relationship.

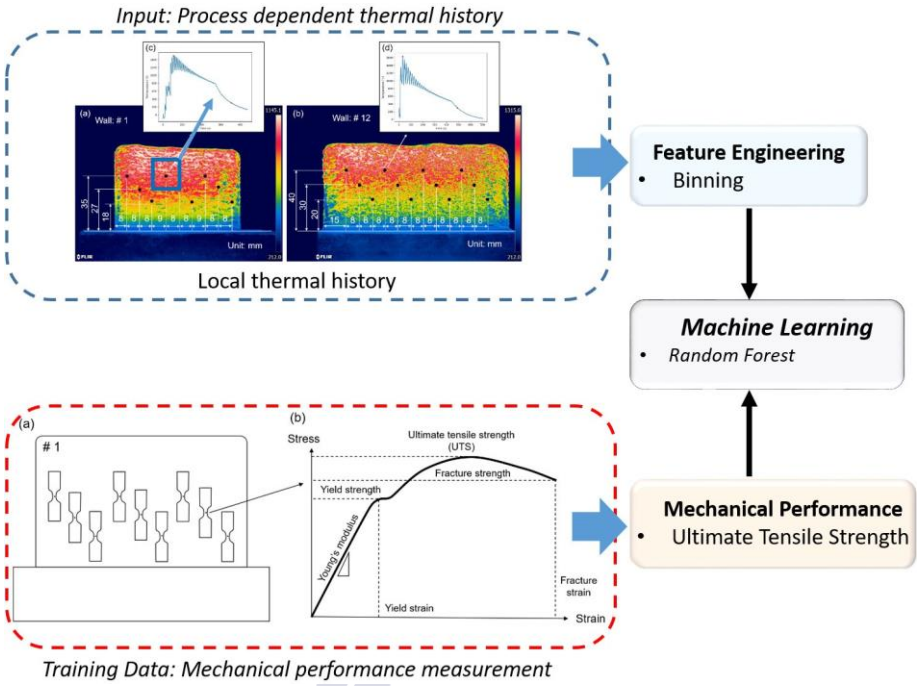


Figure 16: Schematic diagram outlining the AI framework to link thermal history with part performance. Extracted local thermal history at designated points are reduced by binning. The target database is built using ultimate tensile strength obtained by experiments. Machine learning training uses Random Forest algorithm.

Figure 16 shows the framework used in this example. Infrared (IR) imaging records the thermal history for each point in an AM built thin wall. A total of 12 such walls are considered for the study each having 120 layers with 0.5 mm layer height. Details of the experiments are reported in [70]. A total of 135 temperature-time histories and corresponding ultimate tensile strength are accumulated as samples. Because of the high dimensional nature of collected thermal histories, a binning technique for dimension reduction is applied as shown in Figure 17(a). The total temperature range from the first peak to the end from all the collected samples are divided into bins of 50 °C and time spent (in seconds) in all those bins are considered as features. In this way, the continuous thermal histories are converted into an N × M matrix where N is the number of samples and M is the number of total bins (see Figure 17(b)). The corresponding ultimate tensile strength of the AM parts are collated into a N × 1 vector. Using the binned data, a Random Forest (RF) regression [71] supervised machine learning is used to link the reduced thermal history with mechanical performance. The coefficient of determination R^2 with 95% confidence interval for both training and testing data (split as 80% training-20% testing) are shown in Figure 18. Increasing the number of considered features (number of bins in the input matrix), tends to increase the R^2 both in training and testing. Thirty five features completely describe all the thermal histories in the sample. The training time for the random forest algorithm is 0.18 s on a 2.3 GHz Intel Core i5 processor. Although this AI approach can capture very complex relationships between temperature history and ultimate tensile strength in AM, our model overfits the data as indicated by the difference of R^2 between training and test datasets as shown in Figure 18. Hence, an alternative dimension reduction or regression method is required. With no prior knowledge it is hard to decide which AI tool or technique should be chosen.

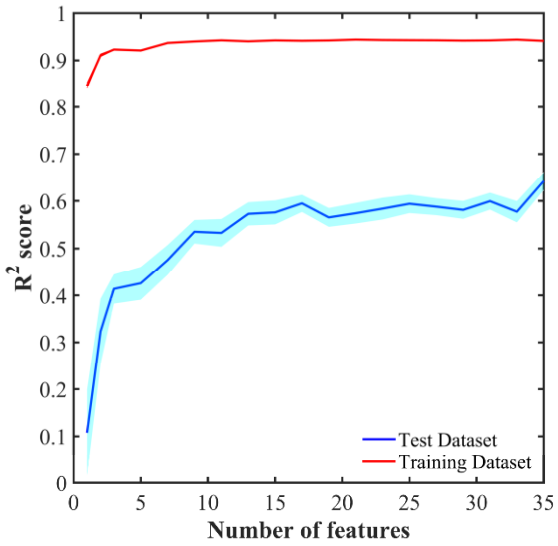


Figure 18: Variation in the R^2-score with considered number of features in Random Forest algorithm. Shaded regions represent 95% confidence interval.

We can use the HiDeNN framework to solve this problem and obtain insight on the governing physics as shown in Figure 19. To solve this example, the HiDeNN will consist of input layer (location, time, temperature, and manufacturing process parameter (such as scan speed) as inputs), pre-processing functions, EXP-NN layer, solution layer, and operation layers. If we look back to Figure 2, spatial location is Ω, processing history is t, and manufacturing conditions are p. The pre-processing function can be designed so that high-dimensional thermal history data at different locations on the wall becomes tractable for learning algorithms. A candidate function for this operation can be continuous wavelet transformation function. The combination of solution and operation layers will discover unknown physics relating thermal history and mechanical property in AM built wall. The loss function can be defined as,

$$L = \frac{1}{S^{NT}} \sum_{i=1}^{} \left(UT^{exp} - UTS^{exp} \right)^2 + h\| \mathbf{P}(\mathbf{x}, t, T, UTS^{exp}) - \Theta(\mathbf{x}, t, T, UTS^{exp}) \|_2 \quad (27)$$

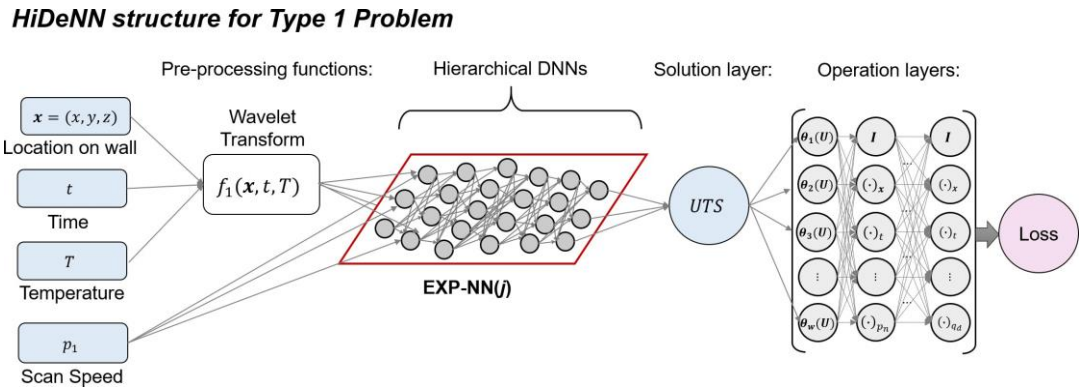


Figure 19: Proposed HiDeNN framework to solve Type 1 problem. The input spatiotemporal variables are first pre-processed by wavelet transform before entering DNN. The DNN uses those processed inputs and parameters to predict the UTS, and the UTS will enter the loss function for training variables in DNN to improve prediction.

where, L is the loss function, N_T is the number training samples, UTS_{exp} is the ultimate tensile strength from experimental observations, UTS_i is the predicted ultimate tensile strength from the HiDeNN, h is the Lagrange multiplier, $\mathbf{P}(\mathbf{x}, t, T, UTS_{exp})$ is a function of operators and expressions such as addition, multiplication, differentiation, or integration, $\boldsymbol{\Theta}(\mathbf{x}, t, T, UTS_{exp})$ is a function of position (location on the wall), time, and temperature, and $\|.\|_2$ is the L_2 norm. The first term of Eq. 27 comes from the hierarchical DNN layer while the second term comes from the operations layer. Combined minimization of these two terms with the Lagrange multiplier for the latter one will give us a mathematical expression for the relationship between spatiotemporal co-ordinates, temperature, and ultimate tensile strength, revealing unknown physics. One concern is that the experimental data contain noise and uncertainty. In order to tackle this problem the hierarchical DNN layer can be a Bayesian neural network resulting in probabilistic terms in the mathematical expression. This will be a part of our future research on HiDeNN.

4.2. Type 2: Mechanistically insuflcient problems with limited data

Type 2 problems are problems for which the available physical information is incomplete. For example, the governing equations may be known, but all the parameters in the governing equations are not explicitly identified. To illustrate, we present here how fatigue life of an AM part can be predicted from statistical information about microstructures with porosity. In this case, we know the governing physics of the problem on the continuum scale but there is limited data relating microstructural porosity and process parameters, and the spread in fatigue life is quite large making empirical fatigue predictions inaccurate. By incorporating experimental images directly higher simulation fidelity is achieved, with the trade-off of higher computational expense.

To predict fatigue response a computational crystal plasticity material law is used [72, 73], which predicts the local cyclic change in plastic shear strain (denoted $\Delta\gamma^p$). This cyclic change saturates relatively quickly (up to 10 cycles may be needed, but in this case after about 3 or 4 cycles), and the saturated value is used as input to a microstructurally fatigue life using, e.g., reference experimental data for the material of interest.

The crystal plasticity and FIP methods have been implemented in already explained Self-consistent Clustering Analysis (SCA) with crystal plasticity material law (termed CPSCA, as described in previous works [73, 75, 76, 48]). Example images, a schematic of the solution method, and the resulting prediction of number of incubation cycles for an example microstructure from various possible images is shown in Figure 20. For this model there are 16 clusters in the matrix phase and 4 in the void phase, selected to balance accuracy and computation cost based on prior experience with similar systems [48]. Constructing the "offline" data for each image in the SCA database cost about 200 s but need only be run once to provides a complete training set for all possible boundary conditions for that image using an implementation of the FFT-based elastic analysis in Fortran. The "online" part of SCA took about 15 or 20 seconds per loading condition per microstructure image to compute fatigue crack incubation life, N_{inc}, using crystal plasticity. While a comparison to an DNS solution with crystal plasticity has not been conducted for this case (see [48] for more thorough analysis), this represents about a factor of about two speed up even when comparing an elastic analysis with DNS versus a full crystal plasticity analysis with the online SCA method for one loading condition. The more loading conditions required, the more favorable this comparison becomes for SCA as no re-training is required after the initial "offline" data is generated. The loading conditions shown are approximately uniaxial tension/compression in the vertical axis (extracted from a multiscale simulation, so uniaxiality is not fully guaranteed), specified via applied deformation gradient in each voxel. The resulting information can be used as training data for the HiDeNN, as shown mathematically in the loss function given in Eq. 28.

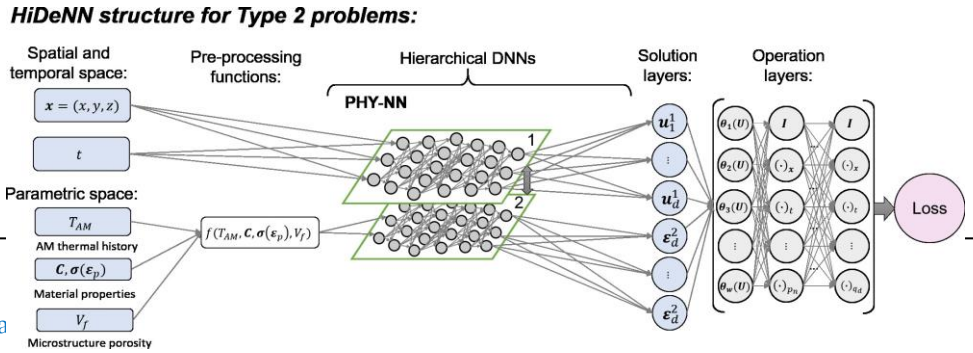


Figure 21: Proposed HiDeNN framework for type 2 problem. The inputs will be different AM processing parameters and associated material and microstructure parameters. The HiDeNN structure could solve the macroscale problem, a AM process simulation, to obtain thermal history. Microscale could be solved for microstructure fatigue life based on the thermal history and material parameters.

For this example, HiDeNN could be applied to construct a relationship between the process, experimental microstructural images, and material performance. The relationship can be regarded as a new material performance prediction formulation, where microstructural features can be directly considered by using a deep convolutional neural network (CNN) as the NN within HiDeNN for image feature identification. A proposed framework for solving this problem is shown in Figure 21. For an AM build, the x and t can be location and process history. In the parametric input, we can consider the process parameters, basic material properties, and images of porosity (and potentially other microstructural features). The pre-processing functions are employed to prepare the 3D images for the CNN. The solution layer contains the fatigue life and the mechanical response of the RVE. The operation layer is designed.

5. Future Outlooks of HiDeNN

The article so far discusses the construction and application of HiDeNN framework and how we can apply the framework to three challenging problems (see Section 4) in computational science and engineering. This section discusses on necessary future extensions of HiDeNN.

To solve type 2 or mechanistically insuflcient problems with limited data, we might need to leverage the available experimental data from literature. However, the data coming from multiple sources are bound to suffer from lack of similarity in experimental conditions and thus cannot be applied directly as input. For such physical problems, we know some information about the system and data can come from known physics. In order to develop the DNN layers of HiDeNN (see Figure 2), we can use transfer learning technique to merge experimental observations from different sources and existing data on the system. We have demonstrated a representative example on how transfer learning can be incorporated in HiDeNN. However, more research is needed to develop a method that can smoothly combine the EXP-NN and PHY-NN through transfer learning. Combining HiDeNN with symbolic regression methods, such as genetic programming [69], might provide explicit mathematical expressions, which sheds the important insights on the problems of interest. In order to solve multi-scale problems, the HiDeNN can be explore further so that it can directly take experimental results (such as micrographs) into the framework as input. Another possible application of HiDeNN would be in computational bio-mechanics where the governing physical equations of many problems (like bone growth) are not known. These problems fall under type 1 or purely data-driven problems. By implementing HiDeNN, leveraging diagnostic results and/or images, the governing equations of bio- mechanics problems can be extracted as a function of the complex inputs such as age, sex, genetic information, or bone mineral density.

6. Summary

We present a novel framework, HiDeNN, as a narrow AI methodology to solve a variety of computational science and engineering problems. HiDeNN can assimilate many data-driven tools in an appropriate way, which provides a general approach to solve challenging computational problems from different fields. A detailed discussion on the construction of HiDeNN highlights the flexibility and generality of this framework. We illustrate an application of HiDeNN to perform multiscale analysis of composite materials with heterogeneous microstructure. Unique features of HiDeNN can offer automatic enrichment at the locations of strain concentration thus capturing effect of variable microstructure at part-scale. The results imply HiDeNN's ability to be applied to a class of computational mechan ics problem where each material point at macroscale corresponds to non-uniform structure at microscale such as functional gradient alloy materials. We need further research to make HiDeNN automatic for arbitrary 3D problems. Furthermore, we apply HiDeNN to discover governing dimensionless parameters from experimental mechanistic data. The successful application of HiDeNN to such problems implies that similar framework can be applied to the field where the explicit physics is scarce, such as additive manufacturing. Finally, we propose future outlooks for solving three challenging problems using the same proposed AI framework. We demonstrate that HiDeNN has extra-ordinary features and can be a general solution method that takes advantage of ever increasing data from different experiments and theoretical model for fast prediction. A word of caution is that HiDeNN is still a proposed framework and further extensions and validations are needed before it can become a generally applicable AI

framework to solve problems in diverse fields from mechanical engineering to biological science in the near future.

References

[1] M.-W. Staff, Merriam-Webster's collegiate dictionary, volume 2, Merriam-Webster, 2004.
[2] E. Rich, K. Knight, Artificial intelligence, Second Edition, Mc Graw-Hill, 1990.
[3] A. M. Turing, Computing machinery and intelligence, in: Parsing the turing test, Springer, 2009, pp. 23–65.
[4] A. Kaplan, M. Haenlein, Siri, siri, in my hand: Who's the fairest in the land? on the interpretations, illustrations, and implications of artificial intelligence, Business Horizons 62 (2019) 15–25.
[5] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, et al., Mastering the game of go without human knowledge, nature 550 (2017) 354–359.
[6] B. Goertzel, P. Wang, A foundational architecture for artificial general intelligence, Advances in artificial general intelligence: Concepts, architectures and algorithms 6 (2007) 36.
[7] N. Peek, C. Combi, R. Marin, R. Bellazzi, Thirty years of artificial intelligence in medicine (aime) conferences: A review of research themes, Artificial intelligence in medicine 65 (2015) 61–73.
[8] L. Deng, D. Yu, et al., Deep learning: methods and applications, Foundations and Trends® in Signal Processing 7 (2014) 197–387.
[9] I. Goodfellow, Y. Bengio, A. Courville, Deep learning, MIT press, 2016.
[10] T. Kirchdoerfer, M. Ortiz, Data-driven computational mechanics, Computer Methods in Applied Mechanics and Engineering 304 (2016) 569 81–101.
[11] R. Iban˜ez, D. Borzacchiello, J. V. Aguado, E. Abisset-Chavanne, E. Cueto, P. Ladeve`ze, F. Chinesta, Data-driven non-linear elasticity: constitutive manifold construction and problem discretization, Computational Mechanics 60 (2017) 813–826.
[12] W. K. Liu, G. Karniakis, S. Tang, J. Yvonnet, A computational mechanics special issue on: data-driven modeling and simulation—theory, methods, and applications, 2019.
[13] A. L. Tarca, V. J. Carey, X.-w. Chen, R. Romero, S. Dra˘ghici, Machine learning and its applications to biology, PLoS computational biology 3 (2007).
[14] E. Garc´ıa-Cano, F. A. Cos´ıo, L. Duong, C. Bellefleur, M. Roy-Beaudry, J. Joncas, S. Parent, H. Labelle, Prediction of spinal curve progression in adolescent idiopathic scoliosis using random forest regression, Computers in biology and medicine 103 (2018) 34–43.
[15] S. S. Halabi, L. M. Prevedello, J. Kalpathy-Cramer, A. B. Mamonov, A. Bilbily, M. Cicero, I. Pan, L. A. Pereira, R. T. Sousa, N. Abdala, et al., The rsna pediatric bone age machine learning challenge, Radiology 290 (2019) 498–503.
[16] N. Wagner, J. M. Rondinelli, Theory-guided machine learning in materials science, Frontiers in Materials 3 (2016) 28.
[17] J. Schmidt, M. R. Marques, S. Botti, M. A. Marques, Recent advances and applications of machine learning in solid-state materials science, npj Computational Materials 5 (2019) 1–36.
[18] D. Ushizima, M. Noack, A. Hexemer, Data science and machine learning for polymer films and beyond, Bulletin of the American Physical
[19] C. Drouillet, J. Karandikar, C. Nath, A.-C. Journeaux, M. El Mansori, T. Kurfess, Tool life predictions in milling using spindle power with the neural network technique, Journal of Manufacturing Processes 22 (2016) 161–168.
[20] B. Zhang, S. Liu, Y. C. Shin, In-process monitoring of porosity during laser additive manufacturing process, Additive Manufacturing 28
[21] Z. Gan, H. Li, S. J. Wolff, J. L. Bennett, G. Hyatt, G. J. Wagner, J. Cao, W. K. Liu, Data-driven microstructure and microhardness design in additive manufacturing using a self-organizing map, Engineering 5 (2019) 730–735.
[22] Y. Yu, T. Hur, J. Jung, I. G. Jang, Deep learning for determining a near-optimal topological design without any iteration, Structural .
[23] X. Lei, C. Liu, Z. Du, W. Zhang, X. Guo, Machine learning-driven real-time topology optimization under moving morphable component- based framework, Journal of Applied Mechanics 86 (2019).
[24] I. Sosnovik, I. Oseledets, Neural networks for topology optimization, Russian Journal of Numerical Analysis

and Mathematical Modelling 34 (2019) 215–223.
[25] M. Bessa, R. Bostanabad, Z. Liu, A. Hu, D. W. Apley, C. Brinson, W. Chen, W. K. Liu, A framework for data-driven analysis of materials under uncertainty: Countering the curse of dimensionality, Computer Methods in Applied Mechanics and Engineering 320 (2017) 633–667.
[26] Y. R. Tabar, U. Halici, A novel deep learning approach for classification of eeg motor imagery signals, Journal of neural engineering 14.
[27] C. Fan, Y. Sun, Y. Zhao, M. Song, J. Wang, Deep learning-based feature engineering methods for improved building energy prediction, Applied energy 240 (2019) 35–45.
[28] E. A. del Rio-Chanona, J. L. Wagner, H. Ali, F. Fiorelli, D. Zhang, K. Hellgardt, Deep learning-based surrogate modeling and optimization for microalgal biofuel production and photobioreactor design, AIChE Journal 65 (2019) 915–923.
[29] S. H. Rudy, S. L. Brunton, J. L. Proctor, J. N. Kutz, Data-driven discovery of partial differential equations, Science Advances 3 (2017).
[30] S. L. Brunton, J. L. Proctor, J. N. Kutz, Discovering governing equations from data by sparse identification of nonlinear dynamical systems, Proceedings of the national academy of sciences 113 (2016) 3932–3937.
[31] K. Champion, B. Lusch, J. N. Kutz, S. L. Brunton, Data-driven discovery of coordinates and governing equations, Proceedings of the National Academy of Sciences 116 (2019) 22445–22451.
[32] H. Li, O. L. Kafka, J. Gao, C. Yu, Y. Nie, L. Zhang, M. Tajdari, S. Tang, X. Guo, G. Li, S. Tang, G. Cheng, W. K. Liu, Clustering discretization methods for generation of material performance databases in machine learning and design optimization, Computational Mechanics 64 (2019)
[33] F. Fritzen, M. Ferna´ndez, F. Larsson, On-the-fly adaptivity for nonlinear twoscale simulations using artificial neural networks and reduced order modeling, Frontiers in Materials 6 (2019) 75.
[34] S. Xiao, R. Hu, Z. Li, S. Attarian, K.-M. Bjo¨rk, A. Lendasse, A machine-learning-enhanced hierarchical multiscale method for bridging from molecular dynamics to continua, Neural Computing and Applications (2019) 1–
[35] H. Chan, B. Narayanan, M. J. Cherukara, F. G. Sen, K. Sasikumar, S. K. Gray, M. K. Chan, S. K. Sankaranarayanan, Machine learning classical interatomic potentials for molecular dynamics from first-principles training data, The Journal of Physical Chemistry C 123 (2019)
[36] W. Yan, S. Lin, O. L. Kafka, Y. Lian, C. Yu, Z. Liu, J. Yan, S. Wolff, H. Wu, E. Ndip-Agbor, et al., Data-driven multi-scale multi-physics models to derive process–structure–property relationships for additive manufacturing, Computational Mechanics 61 (2018) 521–541.
[37] X. Li, Y. Zhang, H. Zhao, C. Burkhart, L. C. Brinson, W. Chen, A transfer learning approach for microstructure property predictions, Scientific reports 8 (2018) 1–13.
[38] J. Gao, M. Shakoor, H. Jinnai, H. Kadowaki, E. Seta, W. K. Liu, An inverse modeling approach for predicting filled rubber performance,Computer Methods in Applied Mechanics and Engineering 357 (2019) 112567.
[39] M. A. Bessa, P. Glowacki, M. Houlder, Bayesian machine learning in metamaterial design: Fragile becomes supercompressible, Advanced Materials 31 (2019) 1904845.
[40] Y. Liu, T. Zhao, W. Ju, S. Shi, Materials discovery and design using machine learning, Journal of Materiomics 3 (2017) 159–177.
[41] C. P. Gomes, B. Selman, J. M. Gregoire, Artificial intelligence for materials discovery, MRS Bulletin 44 (2019) 538–544.
[42] M. Mozaffar, R. Bostanabad, W. Chen, K. Ehmann, J. Cao, M. Bessa, Deep learning predicts path-dependent plasticity, Proceedings of the National Academy of Sciences 116 (2019) 26414–26420.
[43] H. Yang, X. Guo, S. Tang, W. K. Liu, Derivation of heterogeneous material laws via data-driven principal component expansions, Computational Mechanics 64 (2019) 365–379.
[44] S. Tang, G. Zhang, H. Yang, Y. Li, W. K. Liu, X. Guo, Map123: A data-driven approach to use 1d data for 3d nonlinear elastic materials modeling, Computer Methods in Applied Mechanics and Engineering 357 (2019) 112587.
[45] S. Tang, Y. Li, H. Qiu, H. Yang, S. Saha, S. Mojumder, W. K. Liu, X. Guo, Map123-ep: A mechanistic-based data-driven approach for numerical elastoplastic analysis, Computer Methods in Applied Mechanics and Engineering 364 (2020) 112955.
[46] S. L. Brunton, J. N. Kutz, Data-driven science and engineering: Machine learning, dynamical systems, and

control, Cambridge University Press, 2019.
[47] Z. Liu, M. Bessa, W. K. Liu, Self-consistent clustering analysis: an efficient multi-scale scheme for inelastic heterogeneous materials, Computer Methods in Applied Mechanics and Engineering 306 (2016) 319–341.
[48] C. Yu, O. L. Kafka, W. K. Liu, Self-consistent clustering analysis for multiscale modeling at finite strains, Computer Methods in Applied Mechanics and Engineering 349 (2019) 339–359.
[49] M. Tajdari, A. Pawar, H. Li, F. Tajdari, A. Maqsood, E. Cleary, S. Saha, Y. J. Zhang, J. F. Sarwark, W. K. Liu, Image-based modeling for adolescent idiopathic scoliosis:mechanistic machine learning analysis and prediction, Under review at Computer Methods in Applied Mechanics and Engineering in the same special issue (2020).
[50] L. Zhang, L. Cheng, H. Li, J. Gao, C. Yu, R. Domel, Y. Yang, S. Tang, W. K. Liu, Hierarchical deep learning neural networks: Finite elements and beyond, Acceptance after minor revision in Computational Mechanics (submitted August 24, 2020).
[51] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, A. Lerer, Automatic differentiation in pytorch (2017).
[52] D. P. Kingma, J. Ba, Adam: A method for stochastic optimization, arXiv preprint arXiv:1412.6980 (2014).
[53] ABAQUS/Standard User's Manual, Dassault Syste`mes Simulia Corp, United States, 2016.
[54] T. Hughes, J. Cottrell, Y. Bazilevs, Isogeometric analysis: Cad, finite elements, nurbs, exact geometry and mesh refinement, Computer Methods in Applied Mechanics and Engineering 194 (2005) 4135 – 4195.