sciendo

B&H Electrical Engineering, Volume 16, Special Issue, 2022:1-6
ISSN:2566-3143, eISSN:2566-3151, DOI: 10.2478/bhee-2022-0006

Original scientific paper

Submitted: September 6, 2022

Accepted: September 26, 2022

ABOUT APPLICATIONS OF DEEP LEARNING OPERATOR NETWORKS FOR DESIGN AND OPTIMIZATION OF ADVANCED MATERIALS AND PROCESSES

Seid Korić[1,2], Diab W. Abueidda[1]

Abstract: The paper explores the possibility of using the novel Deep Operator Networks (DeepONet) for forward analysis of numerically intensive and challenging multiphysics designs and optimizations of advanced materials and processes. As an important step towards that goal, DeepONet networks were devised and trained on GPUs to solve the Poisson equation (heat-conduction equation) with the spatially variable heat source and highly nonlinear stress distributions under plastic deformation with variable loads and material properties. Since DeepONet can learn the parametric solution of various phenomena and processes in science and engineering, it was found that a properly trained DeepONet can instantly and accurately inference thermal and mechanical solutions for new parametric inputs without re-training and transfer learning and several orders of magnitude faster than classical numerical methods.

Keywords: Machine Learning, Deep Operator Networks, Design, Optimization, Advanced Materials

INTRODUCTION

Many computationally processes used in design and optimization of advanced materials and processes, such as topological optimization, parametric and sensitivity studies and uncertainty quantification, require a vast number of forward functional evaluations in order to obtain converging statistic. These evaluations are traditionally performed by classical numerical methods such as finite elements, given in Figure 1. For high fidelity multiphysics models with complex nonlinear material behaviour, these simulations are often prohibitively computationally expensive in the design and optimization loops, particularly with traditional sampling methods such as Monte Carlo or Latin Hypercube. Instead of numerical forward evolutions, researchers have lately trained surrogate data-driven or physics-informed deep neural networks. Once properly trained on accelerated computing systems, such as graphical processing units (GPU), they can almost instantly inference accurate results for analysis, interactive design, and optimization. Fatehi et al. [1] used a hybrid machine learning and finite element approach for accelerated design of novel architectured ceramics materials. Similarly, Kollmann et al. [2] used convolutional neural networks in topological optimization and design of 2D metamaterials. Artificial neural networks were also successfully used in modelling and parametric studies of processing parameters in additive manufacturing processes [3], bioinspired hierarchical composite design and additive manufacturing [4], or in sensitivity and uncertainty quantification analysis in optical assembly processes [5], among many others.

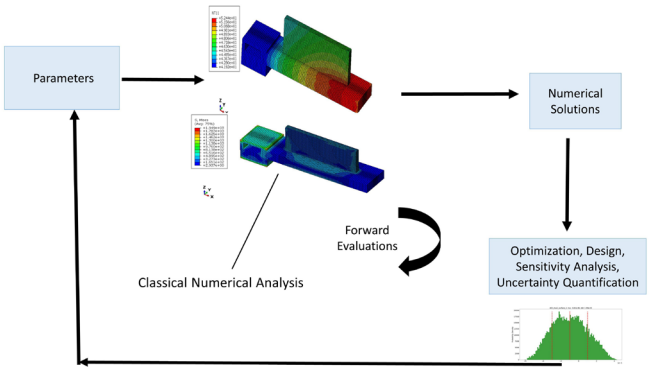

Figure 1: Classical Optimization Design Loop

However, predicting large number of spatial solution variables with parameters often requires large data-driven neural networks, which are difficult to train even on the latest high-performance computing (HPC) systems. In addition, physics-informed neural networks require re-training or transfer learning for any change in input parameters, such as loads, boundary conditions, material properties, or to advance solution between nonlinear solution increments [6]. To address this issue, Lu et al. [7] developed deep learning operator network DeepONet, a novel operator learning architecture, which can learn solution of parametric governing equations. The original

[1] University of Illinois, Illinois, USA
[2] Correspondence email: koric@illinois.edu
© 2022 Author(s). This is an open access article licensed under the Creative Commons Attribution License 4.0. (http://creativecommons.org/licenses/by/4.0/).

data-driven DeepONet formulation has been recently extended in [8] by the information from governing equations in so-called physics-informed DeepONet. When either data-driven or physics-informed DeepONet is adequately trained, it can almost instantly inference results when a new set of parametric values is presented, and without any re-training or transfer learning. This makes DeepONet an excellent candidate for forward functional evaluations in the design and optimization outer loops.

1. FORMULATIONS

1.1. Deep Operator Network (DeepONet)

A general DeepONet training framework is shown in Figure 2. Instead of just considering a collection of points on a domain $\bar{y}$, DeepONet also takes into account input functions u with m parameters. The solution operator $\hat{O}$ predicts a solution by combining the two inputs in a dot product given in (1).

$$\hat{O}(u)(\bar{y}) = \sum_{i=1}^{q} b_i t_i \tag{1}$$

The branch network gets u at each of its m points and predicts q intermediate values b_i, which are intermediate, also known as latent, outputs. Similarly, the trunk network receives y as an input and predicts q intermediate outputs in t_i. The hyperbolic tangent activation function is utilized by both the branch and trunk networks in order to simulate the complicated nonlinear functional relationships that exist between inputs and outputs. The loss function in (2) is based upon mean squared error:

$$Loss = \frac{1}{N * P} \sum_{i=1}^{N} \sum_{j=1}^{P} \left(\hat{O}(u_i)(\bar{y}_{u_i}^{j}) - s(u_i)(\bar{y}_{u_i}^{j}) \right)^2 \tag{2}$$

N is the number of u input functions (each with m parameters). In each training step, for each sample function u_i, the network predicts $\hat{O}(u_i)(\bar{y})$, which is evaluated at P output sensor locations and compared to the corresponding target solutions $s(u)(\bar{y})$ Targets (labels) are calculated by classical numerical methods, such as Finite Difference (FDM) or Finite Elements (FEM) methods, in an offline (data-generation) stage.

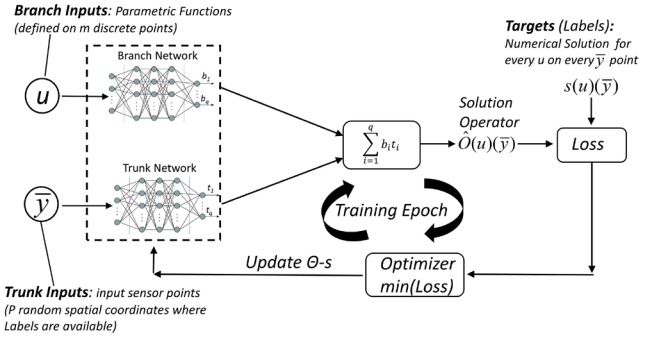

Figure 2: DeepONet Training Framework

Backpropagation is then used to determine the gradient of the loss function with respect to the weights in both networks as part of the learning process, and the Adam optimizer then minimizes the loss value by adjusting the weights. After a significant number of iterations using feedforward and backpropagation (also known jointly as epochs), DeepONet is able to learn the solution operator. With this knowledge, it is then able to infer a discrete point solution for an unknown input function with parametric values almost immediately. Interested readers can find a comprehensive explanation and overview of artificial neural networks in the work of Pattanayak et al. [9].

1.2. Conceptual Optimization Design Framework

A conceptual computational framework using DeepONet instead of numerical analysis for forward evolutions is shown on Figure 3. A previously trained and validated DeepONet is just performing inferencing (feedforward passes), which are mostly consisting of multiplications of dense matrices of trained parameters with vectors. These computational kernels are nowadays exceedingly well optimized, and therefore can provide solution predictions in a fraction of a second.

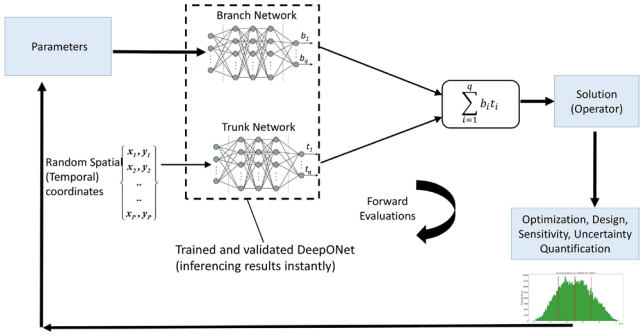

Figure 3: Conceptual design framework with trained DeepONet

In the rest of this exploratory work, we will investigate how DeepONet can learn the solution of a parametric heat conduction and nonlinear stress analysis - two important components of thermal, mechanical, or combined thermo-mechanical analysis. We also validate the DeepONet predictions against numerical analysis and compare the computational performance.

2. NUMERICAL EXAMPLES

2.1. Poisson equation with spatially variable source

We first use DeepONet to learn the solution of the Poisson equation, often known as heat-conduction equation. The equation is solved on a domain that is one unit square and has zero Dirichlet boundary conditions, as given in (3).

$$k_x \frac{\partial^2 T}{\partial x^2} + k_y \frac{\partial^2 T}{\partial y^2} + u(x,y) = 0 \qquad (x,y) \in [0,1]\mathrm{x}[0,1]$$
$$BC: \quad T(x,0) = T(x,1) = T(0,y) = T(1,y) = 0 \tag{3}$$

B&H Electrical Engineering, Volume 16, Special Issue, 2022:1-6
ISSN:2566-3143, eISSN:2566-3151, DOI: 10.2478/bhee-2022-0006

Where $T(x,y)$ is an unknown temperature function of two independent var iables, x and y; $u(x,y)$ is a source term function; and the diffusion coefficient is represented by $k_x = k_y = 0.01$. Physical processes such as heat conduction with moving heat source (laser head) in additive manufacturing, potential flow and pressure solvers in Computational Fluid Dynamics (CFD), electrostatics, gravity in astronomy, and molecular dynamics are all governed by the Poisson equation.

The data-driven DeepONet network, each branch with 5 hidden layers with 50 neurons, was trained during 80,000 epochs with 5000 (N=5000) training u samples, each with random heat source values on 144 discrete points (m=144). For each u data sample, target solutions were supplied for 400 output sensor points (P=400), whose coordinates were randomly picked in the 2D domain. During the offline data-generation stage, the second-order finite-difference method FDM solution on the 144x144 grid is used to generate the target solutions on those points. Figures 4 and 5 show a couple of testing samples, never seen by the DeepONet during its training, with random source distributions and their reference (ground truth) from FDM and predicted from DeepONet temperature solutions.

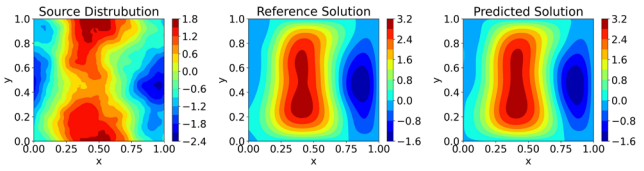

Figure 4: Temperature target solution versus prediction for first random testing sample

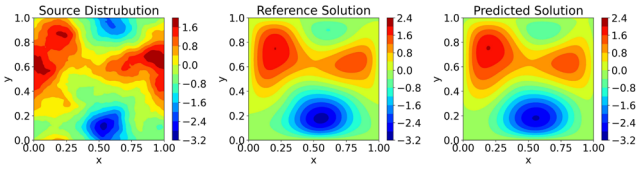

Figure 5: Temperature target solution versus prediction for second random testing sample

In spite of the fact that the peaks and valleys in the parametric source distributions u differ significantly between the test samples, the data-driven DeepONet correctly predicted their 2D diffusive temperature solution, both in the interior, which is governed by the Poisson equation, as well as on the zero-valued Dirichlet boundaries. The data-driven predictions made by DeepONet were validated by visual inspections of many more test samples, which confirmed their accuracy quantitatively. The mean or relative prediction error for 200 testing samples was less than 3%. Several hyper-parameters impact the testing error, such as training size, number of discrete points in a training sample, number of output sensors, network depth and width, and a study of their influence is beyond the scope of this work.

2.2. Nonlinear mechanical deformation with variable loads and material properties

The second case we use to train and validate DeepONet is a materially nonlinear (plastic) deformation. When a stress exceeds a certain critical value, known as the yield stress, the majority of real materials will experience some kind of permanent (plastic) deformation. This is particularly common for many advanced processes and materials, such as in energy absorption devices, additive manufacturing, materials processing, and forming, all experiencing a high level of permanent deformation and damage. Plastic deformation also influences electrical conductivity properties in many materials. In addition, the material frequently displays hardening, which is a further plastic deformation at elevated yield stress levels. Besides being permanent, the plastic deformation is also path-dependents, i.e., it depends on history of path of deformation, which makes it even more challenging to correctly predict stress solution with neural networks.

The total strain in elastic-plastic model is split into two components, written in rate form as:

$$\dot{\boldsymbol{\varepsilon}} = \dot{\boldsymbol{\varepsilon}}_{\mathbf{el}} + \dot{\boldsymbol{\varepsilon}}_{\mathbf{pl}} \tag{4}$$

Strain tensors $\dot{\boldsymbol{\varepsilon}}_{\mathbf{el}}\dot{\boldsymbol{\varepsilon}}_{\mathbf{pl}}$contain the components of the elastic, and plastic strain rates. The stress rate $\dot{\boldsymbol{\sigma}}$, and $\dot{\boldsymbol{\varepsilon}}, \dot{\boldsymbol{\varepsilon}}_{\mathbf{pl}}$, total, and plastic strain rates, are then related by the constitutive equations in (5):

$$\dot{\boldsymbol{\sigma}} = \mathbf{D} : (\dot{\boldsymbol{\varepsilon}} - \dot{\boldsymbol{\varepsilon}}_{\mathbf{pl}}) \tag{5}$$

$\mathbf{D}$ contains the isotropic elasticity material constants based on the elastic modulus and the Poisson ratio. The Huber-von Mises yield surface area, which provides the boundary of elastic stress space limit, is given by (6):

$$g\left(\boldsymbol{\sigma}, \bar{\varepsilon}_p\right) = \bar{\sigma} - \sigma_y(\bar{\varepsilon}_p) = 0 \tag{6}$$

Yield stress σ_y is a linear function of equivalent plastic strain $\bar{\varepsilon}_p$ to account for isotropic hardening, and $\bar{\sigma}$ is von Mises equivalent stress defined for 2D plane stress loading from its components given in (7):

$$\bar{\sigma} = \sqrt{\sigma_{11}^2 + \sigma_{22}^2 + \sigma_{11}\sigma_{22} + 3\sigma_{12}^2} \tag{7}$$

In a typical nonlinear finite element solution procedure, the mechanical equilibrium in (8) is solved incrementally on a global level, and the strain components are sent to each integration point in the domain, where plastic constitutive model integration takes place and updates plastic strains and stresses which are sent back to the global level.

$$\nabla \cdot \boldsymbol{\sigma}(\mathbf{x}) = 0 \tag{8}$$

To generate von-Mises stress solution for training and testing data samples, we used the python open source finite element code GRoT [10] with the explicit Runge-Kutta method for local integration. This integration method, according to Zienkiewicz and Taylor [11], has a second-order accuracy. The interested reader is directed to other references [11], [12] for a more in-depth explanation of finite element solution procedures for plastic and visco-plastic materials. We have again put more emphasis on the approach than the geometry and have employed a straightforward 2D domain, a square of 64 × 64 mm with a 1 mm thickness, and this time using real (steel like) material properties. The right edge nodes of the domain are under the variable tensile/compressive and shearing loads, whereas the left side is fully constrained. The vertical and horizontal loads, including their magnitudes and directions, as well as elastic modulus and yield stress, are the six parameters in the input function u. Linear quad elements with a 1 mm dimension make up the FE mesh. We used the same neural network from the Poisson equation, but with fewer, 2000, training samples and more, 2048, output sensor points, trained with 80,000 epochs. It is important to underline that the Huber-von Mises plasticity, 2D plane stress conditions, and rectangular geometries are not the limitations of the DeepONet approach for plastic deformation explored in this study.

Similarly to the Poisson's results, Figures 6 and 7 show a couple of testing samples with their reference (ground truth) from FEM and predicted from DeepONet von Mises stress solution. The testing samples have random loads, their signs, elastic modulus and yield stress, and the DeepONet never saw their parametric inputs during its training.

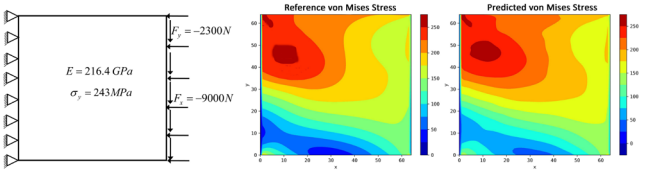

Figure 6: Von Mises stress target solution (middle) versus prediction (right) for first random testing sample

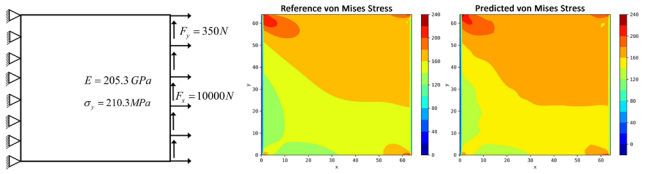

Figure 7: Von Mises stress target solution (middle) versus prediction (right) for second random testing sample

In these two samples with opposing tensile and compressive loads and stress concentrations in the corners, the DeepONet correctly predicted the stress peaks owing to yielding and plastic hardening. The quantitative correctness of the DeepONet von Mises stress predictions was validated by visual examination of the data from several different testing samples. The mean of the relative prediction error of the Six-parameter DeepONet model over 200 testing samples for these hyper-parameters was 2.9%.

3. COMPUTATIONAL PERFORMANCE

The code used in this research is written in JAX [13], a recently developed Python-based toolkit developed by Google for high-performance machine learning research. Advanced automated differentiation (grad), just-in-time compilation (JIT), and cross-device compute replication (pmap) are just a few of the sophisticated features of JAX.

Some of the computationally demanding kernels were offloaded to the V100 GPU via JIT, and pmap proved particularly useful for accelerating data generation in the application to solve the Posson's equation. Automatic differentiation was used by the Adam optimizer, which was also accessible in JAX as a high-level function. A node of the industrial HPC cluster iForge, which was hosted at the National Center for Supercomputing Applications (NCSA) and had four V100 Nvidia GPU cards, was used for computing in this work.

Table I shows the computational cost to make predictions from novel testing input data (inference) using trained DeepONet models, as well as the solution times of the numerical codes used for data generation. The solution times of the highly optimized numerical solvers, different than the codes used for data generation, an implicit finite-difference code with optimized Python libraries for Poisson's equation and Abaqus commercial finite element code [14] for plastic deformation, solving these problems separately on the same grid on iForge's CPUs, are also added.

Table I: Solution times, training and highly-optimized numerical solvers and DeepONet inference

	Num. Sol (Training)	Num. Sol (Optimized)	DeepONet Inference	Testing Error %
Poisson	1.9 sec.	0.05 sec	7 x 10^{-4} sec.	2.8
Plastic	25 sec	5 sec	1 x 10^{-3} sec.	2.9

Generation of training data is usually performed once and it takes 20 minutes for Poisson and a few hours for Plastic DeepONets, while their training takes 5-10 minutes on NVidia's V100 GPUs. Once the DeepONets are adequately trained, they can predict spatial solutions on the entire domains for new parametric values in a fraction of a second and two to four orders of magnitude faster than traditional (even highly optimized) numerical solvers. If we had used the standard fully connected feedforward neural networks instead of the DeepONet with dual networks with 50 outputs each, it would have required 4,096 outputs to generate predictions on the 64 x 64 grid. This would need significantly larger neural networks, and it is doubtful that even the newest and biggest high-performance computers could properly train such a network.

B&H Electrical Engineering, Volume 16, Special Issue, 2022:1-6
ISSN:2566-3143, eISSN:2566-3151, DOI: 10.2478/bhee-2022-0006

4. CONCLUSIONS

In this study, we looked at the potential applications of the novel Deep Operator Networks (DeepONet) for forward thermal and nonlinear mechanical analysis of computationally demanding designs and optimizations of advanced materials and processes. After introducing the dual branch-trunk neural network-based data-driven DeepONet architecture, we concentrated on learning temperature distribution with spatially variable heat source and plastic material behaviour with isotropic von Mises hardening with variable loads and material attributes as parametric inputs in this study. On contemporary GPUs, we found that DeepONets could learn temperature and von Misess stress solutions in a few minutes. Once the networks were adequately trained, we were able to inference accurate solutions almost instantly and without any re-training or transfer learning when new source distribution or loads and material property inputs are given from the testing dataset that the networks have never encountered before. Furthermore, a trained DeepONet model's weights and biases may be transferred to laptops and other low-end computing devices and used for inference without needing HPC, GPUs, or modeling tools. The authors believe that DeepONet and other advanced surrogate deep learning models will assist, or even replace, existing numerical solvers in the future whenever a vast number of prohibitively computationally expensive forward evaluations are required. The concept of our DeepONet-based computational framework outlined in our work is not limited to the design and optimization of advanced materials and processes, and it can be applied in many industrial sectors or other challenging numerical analysis or online control processes that require an excessive number of fast forward evaluations in science and engineering.

ACKNOWLEDGMENT

The authors would like to thank the National Center for Supercomputing Applications (NCSA) at the University of Illinois, and particularly its Industry Program Office (IPO) and the Center for Artificial Intelligence Innovation (CAII) for their support and hardware resources. We also want to thank Robert Konieczny, the developer of the GRoT python plasticity code used for data generation, for his support.

REFERENCES

[1] E. Fatehi, H. Yazdani Sarvestani, B. Ashrafi, A. H. Akbarzadeh: Accelerated design of architectured ceramics with tunable thermal resistance via a hybrid machine learning and finite element approach, Materials & Design, vol. 210, 110056, 2021

[2] H. T. Kollmann, D. W. Abueidda, S. Koric, E. Guleryuz, N. A. Sobh: Deep learning for topology optimization of 2D metamaterials, Materials & Design, vol. 196, 110056, 2021

[3] Q. Zhu, Z. Liu, J. Yan: Machine learning for metal additive manufacturing: predicting temperature and melt pool fluid dynamics using physics-informed neural networks, Computational Mechanics, vol. 67, pp. 619-635, 2021

[4] G. X. Gu, C. Chen, D. J. Richmond, M. J. Buehler: Bioinspired hierarchical composite design using machine learning: simulation, additive manufacturing, and experiment, Materials Horizons vol. 5.5, pp. 939-945, 2018

[5] S. Shahane, E. Guleryuz, D. W. Abueidda, A. Lee, J. Liu, X. Yu, R. Chiu, S. Koric, N. R. Aluru, P. M. Ferreira: Surrogate neural network model for sensitivity analysis and uncertainty quantification of the mechanical behaviour in the optical lens-barrel assembly, Computers & Structures, vol. 270, 106843, 2022

[6] D. W. Abueidda, Q. Lu, S. Koric: Meshless physics-informed deep learning method for three-dimensional solid mechanics, International Journal for Numerical Methods in Engineering, vol. 122(23), pp. 7182-7201, 2021

[7] L. Lu., P. Jin., G. Pang., Z. Zhang, G. E. Karniadakis, Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators, Nat. Mach. Intell., vol. 3, pp. 218–229, 2021

[8] S. Wang, H. Wang, P. Perdikaris, Learning the solution operator of parametric partial differential equations with physics-informed DeepONets, Science Advances, vol. 7(40), pp. 1-9, 2021

[9] S. Pattanayak: Pro Deep Learning with Tensorflow: a Mathematical Approach to Advanced Artificial Intelligence in python, Apress, Springer New York, 2017

[10] GRoT Linear and Nonlinear Finite Element Method solver, https://github.com/tutajrobert/grot

[11] O. C. Zienkiewicz and R. L. Taylor., 2000. The Finite Element Method 5th ed., Oxford Butterworth-Heinemann, Portsmouth, NH, 2000

[12] S. Koric, B. G Thomas: Efficient thermo-mechanical model for solidification processes, International Journal for Numerical Methods in Engineering, vol. 66, pp. 1955–1989, 2006

[13] J. Bradbury, R. Frostig, P. Hawkins, M. J. Johnson, C. Leary, D. Maclaurin, G. Necula, A. Paszke, J. VanderPlas, S. Wanderman-Milne, Q. Zhang: JAX: Composable transformations of Python+NumPy programs (2018).

[14] ABAQUS/Standard User's Manual, Version 2019. Providence, RI: Dassault Systèmes Simulia Corp, 2019

BIOGRAPHY

Seid Korić received his B.S. Degree in Mechanical Engineering from the University of Sarajevo in 1993. He earned M.S. in aerospace engineering (1999) and Ph.D. in mechanical engineering (2006) from the University of Illinois at Urbana-Champaign. Dr. Korić is the technical associate director at the National Center for Supercomputing Applications (NCSA) where he provides technical and strategic leadership for the NCSA's

Research Consulting directorate, whose staff work on solutions for grand challenges in industry and academia. He is also a research professor in the Mechanical Science and Engineering Department at the University of Illinois. His research interests spread wide, covering multiphysics, advanced manufacturing processes, sparse linear solvers, biomechanics, finite element methods, GPUs, artificial intelligence (AI), and industrial HPC computing.

Diab Abueidda earned his B.S. in Mechanical Engineering from the American University of Sharjah in 2013, M.Sc. in Mechanical Engineering from the Masdar Institute of Science and Technology in 2015, and Ph.D. in Mechanical Engineering from the University of Illinois at Urbana-Champaign in 2019. Dr. Abueidda is a research scientist at the National Center for Supercomputing Applications (NCSA) and a faculty affiliate at the Center for Artificial Intelligence Innovation (CAII). His research interests are in computational mechanics, architected materials, the confluence of artificial intelligence (AI) and numerical modelling, additive manufacturing, and topology optimization. Dr. Abueidda has published or contributed to over 30 highly influential research papers and articles on a diverse range of topics.