frontiers | Frontiers in Materials

TYPE Original Research
PUBLISHED 24 March 2023
DOI 10.3389/fmats.2023.1128954

Check for updates

OPEN ACCESS

EDITED BY
Tae Yeon Kim,
Khalifa University, United Arab Emirates

REVIEWED BY
Jesus Martinez-Frutos,
Polytechnic University of Cartagena, Spain
Xingfei Wei,
Johns Hopkins University, United States

*CORRESPONDENCE
Stéphane P.A. Bordas,
✉ stephane.bordas@alum.northwestern.edu

SPECIALTY SECTION
This article was submitted to Computational Materials Science, a section of the journal Frontiers in Materials

RECEIVED 21 December 2022
ACCEPTED 06 March 2023
PUBLISHED 24 March 2023

CITATION
Deshpande S, Sosa RI, Bordas SPA and Lengiewicz J (2023), Convolution, aggregation and attention based deep neural networks for accelerating simulations in mechanics.
Front. Mater. 10:1128954.
doi: 10.3389/fmats.2023.1128954

Convolution, aggregation and attention based deep neural networks for accelerating simulations in mechanics

Saurabh Deshpande[1], Raúl I. Sosa [2], Stéphane P. A. Bordas[1]* and Jakub Lengiewicz[1,3]

[1]Department of Engineering, Faculty of Science, Technology and Medicine, University of Luxembourg, Belval, Luxembourg, [2]Department of Physics and Materials Science, Faculty of Science, Technology and Medicine, University of Luxembourg, Belval, Luxembourg, [3]Institute of Fundamental Technological Research, Polish Academy of Sciences, Warsaw, Masovian, Poland

Deep learning surrogate models are being increasingly used in accelerating scientific simulations as a replacement for costly conventional numerical techniques. However, their use remains a significant challenge when dealing with real-world complex examples. In this work, we demonstrate three types of neural network architectures for efficient learning of highly non-linear deformations of solid bodies. The first two architectures are based on the recently proposed CNN U-NET and MAgNET (graph U-NET) frameworks which have shown promising performance for learning on mesh-based data. The third architecture is Perceiver IO, a very recent architecture that belongs to the family of attention-based neural networks–a class that has revolutionised diverse engineering fields and is still unexplored in computational mechanics. We study and compare the performance of all three networks on two benchmark examples, and show their capabilities to accurately predict the non-linear mechanical responses of soft bodies.

KEYWORDS

surrogate modeling, deep learning-artificial neural network, CNN U-NET, graph U-net, perceiver IO, finite element method

1 Introduction

The ability to make fast or real-time predictions of the response of physical systems is essential for a variety of engineering applications. Notable examples of this can be found in the field of robotics (Rus and Tolley, 2015; Choi et al., 2021) and medical simulations (Cotin et al., 1999; Courtecuisse et al., 2014; Bui et al., 2018; Mazier et al., 2021), which have the potential to advance personalized medicine and improve computer-assisted and robotic surgery (Chen et al., 2020; Dennler et al., 2021). In computational physics and chemistry, fast and accurate predictions are fundamental for studying complex systems, such as those arising in biology and materials science (Friesner, 2005), or in drug discovery (De Vivo et al., 2016). In many cases, the necessary accuracy of these predictions requires complex models that can be expressed through partial differential equations and solved numerically using methods such as the finite element method (FEM) at continuum scales or specialized *ab initio* approaches at the atomic or quantum scales. However, these high-fidelity computational models are often too slow for real-time or practical purposes, and therefore approximate or surrogate models must be developed to achieve the necessary speed-ups.

At the same time, the 21st century has seen an explosion of measurement data, much of which is available as public datasets in various scientific domains, including structural mechanics, material science, and meteorology (Zakutayev et al., 2018; Elouneg et al., 2022; Gholamalizadeh et al., 2022). The availability of this data, combined with the rapid growth in computational resources, has led to the increasing importance of machine learning (ML) techniques (Butler et al., 2018; Bock et al., 2019; Schleder et al., 2019) for solving forward and inverse engineering problems. This includes surrogate and data-driven approaches that aim to enable modeling (Barrios and Romero, 2019), accelerate computationally costly direct numerical simulations (Rupp et al., 2012; Wirtz et al., 2015; Capuano and Rimoli, 2019; Weerasuriya et al., 2021), and even discover new material laws (Liu et al., 2017; Flaschel et al., 2021). The increasing use of ML in engineering and other fields has also spurred the development of various methods and algorithms for improving the accuracy and efficiency of these techniques.

Within the class of machine learning methods for surrogate and data-driven modeling, deep learning (DL) approaches have seen great success due to their ability to efficiently extract complex relationships present in the underlying data. DL models have been successfully employed for a range of tasks in diverse fields such as computational physics and chemistry, material science, computational mechanics, computer vision, natural language processing, and many others (Oishi and Yagawa, 2017; Schütt et al., 2017; Jha et al., 2018; Voulodimos et al., 2018; Schmidt et al., 2019; Brown et al., 2020; Choudhary et al., 2022). In computational chemistry, machine learning force fields (MLFFs), see (Unke et al., 2021), have seen great success in recent years for accelerating costly *ab initio* simulations. For instance, Deep Tensor Neural Network (DTNN) (Schütt et al., 2017) and SchNet (Schütt et al., 2017) models have been shown to accurately predict forces in a variety of molecules and could be used in applications such as protein folding and material design. Similarly, computational mechanics has witnessed an increasing use of DL surrogate models as a replacement for costly direct numerical simulations (Abueidda et al., 2021; Mianroodi et al., 2021). What is common to all the above-mentioned cases is that deep learning techniques rely on deep artificial neural networks (deep ANNs, or DNNs), which must be trained on a sufficiently large amount of data. While this training process is computationally costly, once trained, the predictions of DL models are extremely efficient.

Obtaining necessary amount of training data is often difficult when it originates from physical experiments. This can be due to multiple factors, such as high costs, risks and difficulties associated with the experiments, or data privacy clauses. There are two possible approaches to deal with the scarcity of experimental data. The first approach relies on enhancing the DL model with the information on underlying physics—an approach popularly termed as Physics Informed Neural Networks (PINN) (McFall and Mahan, 2009; Mao et al., 2020; Samaniego et al., 2020; Odot et al., 2022). The second approach includes the underlying physics implicitly, through high-fidelity simulations done *in silico* to provide the necessary amount of synthetically generated data, which has shown to be useful in various applications (Le et al., 2017; Aydin et al., 2019; Pfeiffer et al., 2019; Vijayaraghavan et al., 2021; Kim et al., 2022). In this work, we will follow the latter approach and will focus on DL surrogate models that are trained on synthetically generated data from finite element simulations in non-linear elasticity.

One of the most important aspects that will be studied in this work is the architecture of deep neural networks. The majority of DL approaches that are present in the literature are based on fully connected networks, which can be inefficient and prone to overfitting when applied to high-dimensional inputs. If such large inputs are structured, they fall under the umbrella of geometric deep learning (GDL) (Bronstein et al., 2021), a concept that has gained increasing interest in recent years. In this work, we will compare three architectures that can efficiently handle high-dimensional structured inputs: convolutional neural networks (CNNs), graph neural networks (GNNs), and attention-based networks.

Convolutional neural networks (CNNs) are known to outperform traditional fully-connected ANNs, and this has been demonstrated in various domains, including physics-based simulations (Guo et al., 2016; Deshpande et al., 2022a; Krokos et al., 2022b; El Haber et al., 2022). CNNs work on the principle of parameter sharing and local convolution operations, which enables efficient training on large inputs. Their disadvantage is that the inputs/outputs of CNNs are restricted to grid inputs, such as images, videos, or structured FE meshes. However, CNNs have found their successors, the graph neural networks (GNNs), that can work with any structure of inputs/outputs.

Graph-based approaches leverage the topological information of the input to perform local operations in the respective neighborhood only, and can learn efficiently on generally structured data. Recently, GDL methods have shown promising performance for their applications as well in the field of mechanics, (Battaglia et al., 2018; Vlassis et al., 2020; Pfaff et al., 2021; Krokos et al., 2022a; Strönisch et al., 2022). More recently (Deshpande et al., 2022b), proposed MAgNET, a novel graph U-Net framework for efficiently learning on mesh-based data. In this work we utilise it to accurately predict non-linear deformations of solids.

Attention-based approaches, similar to human cognitive attention, work by allowing the DL model to focus on certain parts of the input data that are relevant to the task at hand. This is done through a fully trainable process that, without the need to introduce topological information or enforce structural restrictions, allows the neural network to extract dependencies from throughout the whole input domain. This type of approach has led to significant strides in a wide range of areas, starting from computer vision (Xu et al., 2015) to natural language processing (Devlin et al., 2018; Baevski et al., 2020), as well as becoming the basic building block of the Transformer architecture (Vaswani et al., 2017). Recently the Perceiver IO (Jaegle et al., 2022), a new type of architecture that builds upon Transformers, has been proposed as a general-purpose model that can handle data from arbitrary settings. Since Perceiver IO has been shown to achieve several state-of-the-art results without the need for problem-specific architecture engineering, we will compare its performance on non-linear deformation prediction of solids based on mesh data against the previously discussed models.

To summarise, in this work we will compare three DNN architectures: two architectures presented in our earlier works, i.e., CNN U-Net framework (Deshpande et al., 2022a), and MAgNET framework (Deshpande et al., 2022b), as well as the attention-based architecture, Perceiver IO (Jaegle et al., 2022), which has not been explored for its applications in mechanics yet. We show the capabilities of three frameworks by learning on non-linear FEM datasets and by cross-comparing their performance. In **Section 2**, we will introduce the three DNN architectures, in **Section 3**, we will study their performance, and in **Section 4** we will summarize the results and discuss future directions.

2 Materials and methods

As previously mentioned in the introduction, in this paper we propose three types of deep neural network (DNN) frameworks that can be used as surrogate models to replace computationally expensive non-linear FEM solvers. The proposed DNN frameworks are trained on force-displacement FEM datasets that are given in the mesh format. Once trained, these surrogate DNN models are able to quickly and accurately simulate the mechanical responses of bodies subjected to external forces. The outline of study pursued in this paper is shown in **Figure 1**.

Any input mesh can be categorised into either a structured mesh or an arbitrary unstructured mesh. In this work we introduce three types of DNN network architectures. The CNN U-Net network can only be (straightforwardly) used for structured meshes, while the MAgNET and Perceiver IO networks are more general and are capable of handling arbitrary mesh inputs. All these frameworks are discussed in detail in the following subsections.

TABLE 1 **Properties of deep neural network architectures studied in this work.**

Framework	Type	Supported mesh
CNN	U-Net (convolution operation)	structured
MAgNET	Graph U-NET (MAg operation)	arbitrary
Perceiver IO	Transformer (attention mechanism)	arbitrary

2.1 DNN frameworks for predicting mechanical deformations

Below we introduce three different types of neural network architectures which can efficiently predict non-linear deformations of bodies subjected to external traction and body forces. All the proposed DNN frameworks directly operate on the finite element mesh data thereby making it very convenient to be used as surrogate models in place of conventional FEM solver. The first two i.e., CNN U-Net and MAgNET belong to the family of U-Net architecture while Perceiver IO is based on Transformer-type attention, see **Table 1**.

2.1.1 CNN U-Net

CNNs were originally proposed for performing classification and regression tasks on image, video like data but lately are even being used for generic inputs such as mesh data which is crucial to many scientific applications. In particular, U-Net like architectures have shown great potential in learning on large-scale inputs and lately have been successfully used

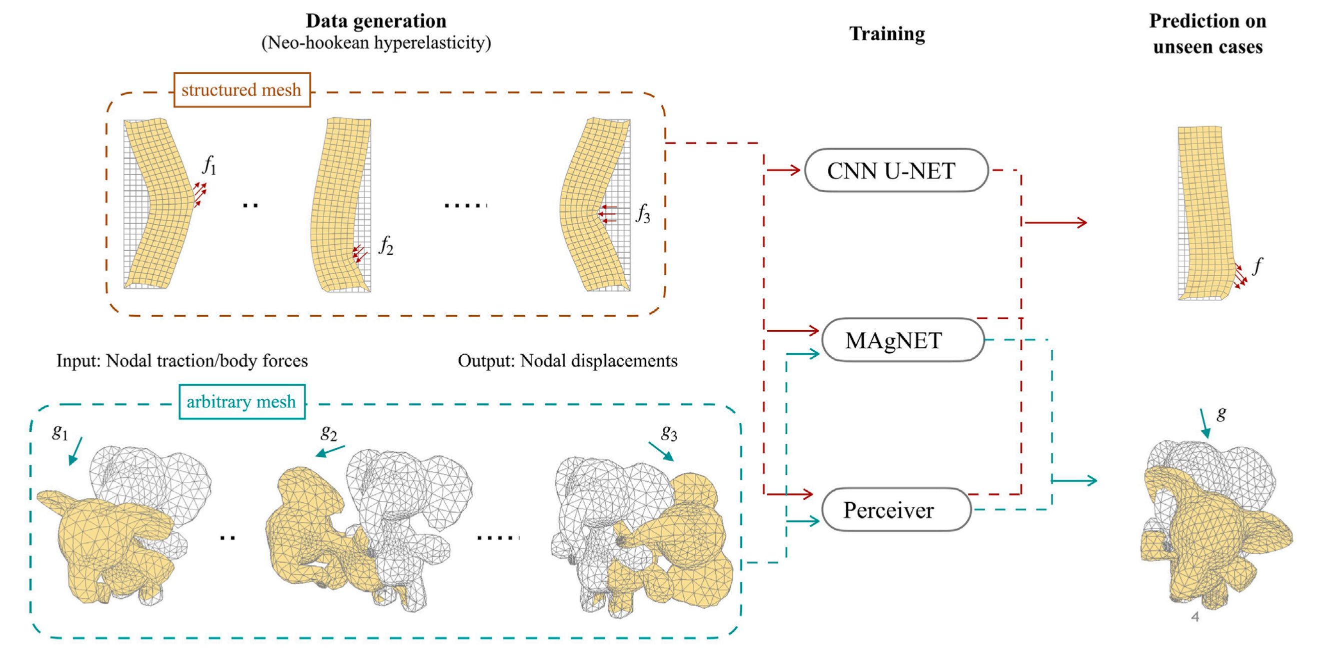

FIGURE 1
Outline of the neural network surrogate frameworks for predicting body deformations. (Left) Training datasets for structured and arbitrary mesh cases are generated by using a non-linear FEM solver. (Middle) Proposed neural network frameworks are trained on these datasets. For structured mesh case, all NN frameworks are used while for arbitrary unstructured meshes only MAgNET and Perceiver IO networks are used. (Right) Trained networks are then used as surrogate models to predict the deformation of bodies under unseen forces.

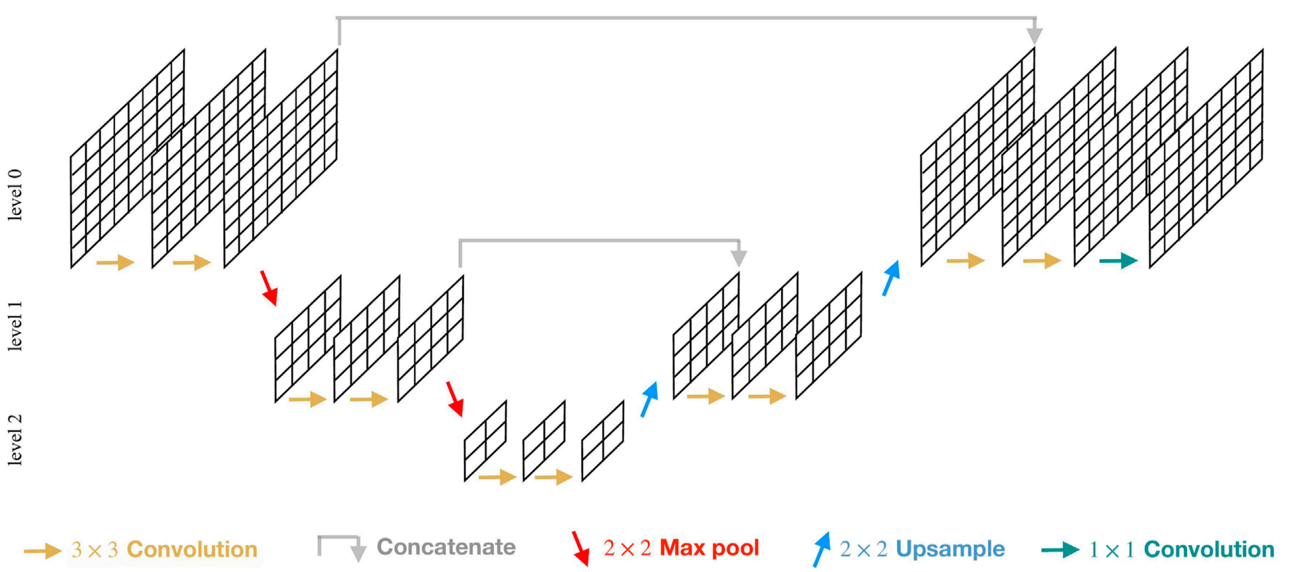

FIGURE 2
Schematic of CNN architecture used for generic structured 2D mesh inputs.

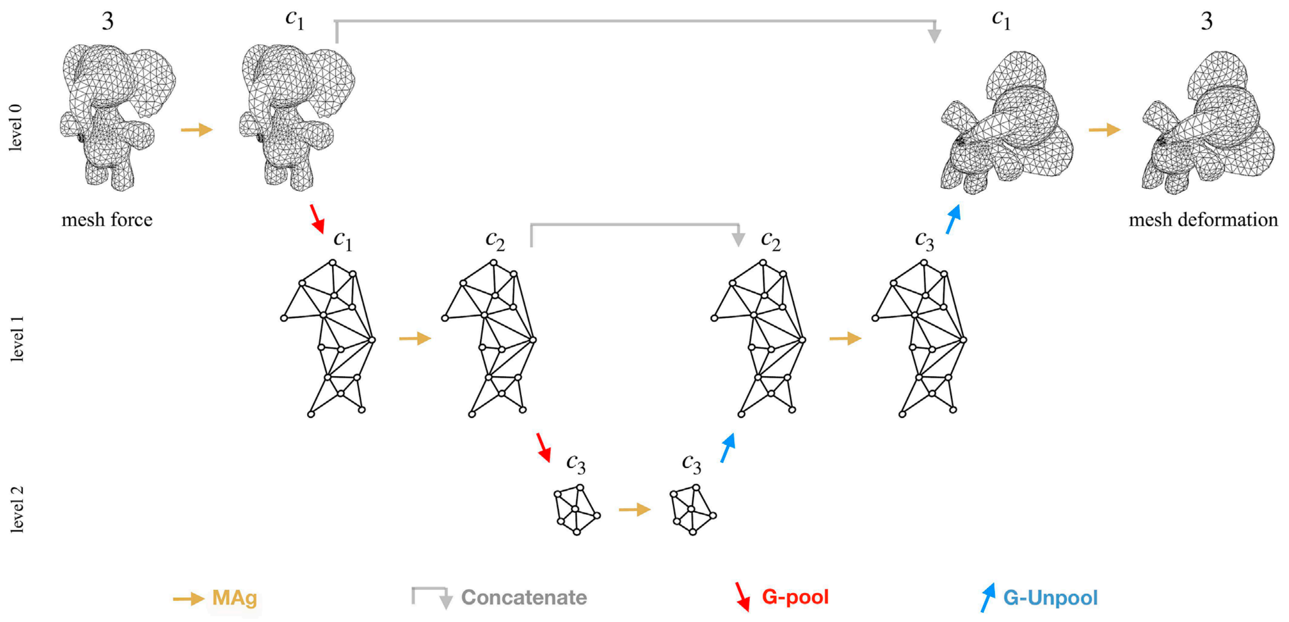

FIGURE 3
Schematic of the MAgNET architecture used in this work. It takes external forces on arbitrary mesh as an input to gives mesh displacements as output.

for simulating mechanical responses of materials as well (Mianroodi et al., 2021; Deshpande et al., 2022a). The name U-Net comes from the particular U-shaped architecture which involves a series of convolutional and pooling operations. Convolutional layers are responsible for non-linear transformations whereas pooling enables learning through low-fidelity representation thus making the network capable of learning on high-dimensional inputs. Experiments presented in this work are carried out by using the CNN U-Net framework proposed by (Deshpande et al., 2022a), see **Figure 2**.

One major limitation of the CNN is that it cannot straightforwardly accommodate unstructured mesh inputs. To overcome this issue, the simplest approach embeds a structured grid on unstructured meshes with a naive mapping between unstructured and grid node values (Mendizabal et al., 2019). While more sophisticated approaches are proposed to make unstructured

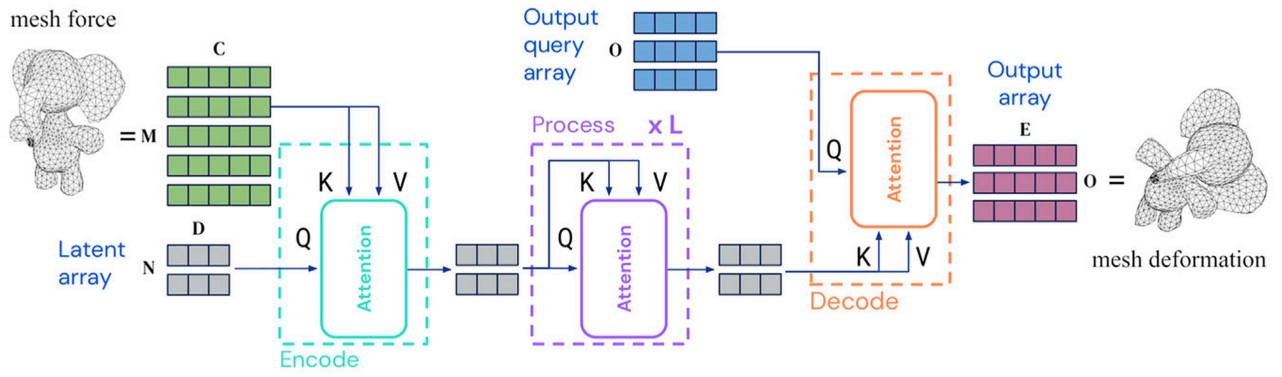

FIGURE 4
Schematic of the Perceiver IO architecture, (Jaegle et al., 2022), used for external forces on arbitrary mesh as inputs and mesh displacement as outputs.

meshes compatible to be used with CNN framework (Brunet et al., 2019). However, they perform poorly on complicated geometries and come with an associated preprocessing cost; they are not considered in the scope of this work.

2.1.2 MAgNET

In an attempt to generalise CNN to arbitrary unstructured meshes, very recently (Deshpande et al., 2022b) proposed the MAgNET framework, see **Figure 3**. MAgNET architecture belongs to the family of graph U-Net architectures and it is proposed for efficient learning on mesh structured data. MAgNET directly accepts arbitrary mesh inputs (such as forces/stresses/displacements of nodes in the mesh) values thus making it very convenient to be used with existing numerical solvers.

MAgNET relies on the so-called MAg layer (Multichannel Aggregation layer) which is capable of learning non-linear transformations between input and output data existing in the mesh format. MAg extends the concept of local operations in convolution layers to arbitrary mesh inputs by performing aggregation of nodal feature values in the respective neighborhood nodes only. It leverages the topology of inputs and performs learnable local aggregations with heterogeneous window sizes as opposed to the fixed-size window in the case of CNN. While its graph pooling/unpooling layers enable efficient learning on large-dimension inputs through reduced graph representation.

2.1.3 Perceiver IO

The Perceiver IO architecture (Jaegle et al., 2022), was developed with the goal of achieving a DL scheme that can easily integrate and transform arbitrary information for arbitrary tasks. This architecture employs an attention encoder that maps inputs from a wide range of modalities to a fixed-size latent space using cross-attention, this latent space is then further processed using self-attention as an usual Transformer and decoded into the output domain *via* cross-attention, see **Figure 4**. This process allows the network to scale to large and multi-modal data since it decouples the bulk of the network's processing from the size and modality-specific details of the input. In this work, we will leverage this property and use Perceiver IO to learn non-linear deformations on unstructured meshes without adding any information or restrictions about how to treat the underlying data structure. During training, Perceiver IO automatically learns the important dependencies that exist in the input domain, composed of arbitrarily unstructured mesh data, and transforms them into the corresponding output which consists of displacement data.

2.2 Input/output and training of DNN surrogate models

As motivated in **Section 2**, the proposed DNN frameworks are trained on the force-displacement datasets. Let us denote the neural network in consideration as h, it is parameterised by trainable parameters, $\boldsymbol{\theta}$. In all the cases, h accepts external forces, $\mathbf{f}$, on all degrees of freedom (dofs) of mesh as the input. And as an output it predicts displacement vector, $\mathbf{u}$, of all dofs (same size as the input), i.e., $h : \boldsymbol{f} \to \boldsymbol{u}$.

In the case of CNN and MagNET, forces associated with X, Y, Z degrees of freedom are kept in different channels. For instance, in the case of CNN U-Net, X, Y directional forces on a 2D quad mesh with $n_x \times n_y$ nodes are fed to the network as a $2 \times n_x \times n_y$ tensor. For MAgNET, they are provided as a single dimensional tensor of shape $2 \cdot n_x \cdot n_y$, with X and Y directional forces concatenated together (each representing a different channel). On the other hand, Perceiver IO inputs are first kept as a single array of size $2 \cdot n_x \cdot n_y$ to which we apply 1 × 1 convolution kernels to project it to a tensor of shape $2 \cdot n_x \cdot n_y \times 256$, adding 256 channels. Finally, to this channel dimension we concatenate trainable 1D positional embeddings, thus leaving us with a tensor of shape $2 \cdot n_x \cdot n_y \times 512$ that we will use as an input for the network.

Now, for a given training dataset $\{(\mathbf{f}_1, \mathbf{u}_1), \ldots, (\mathbf{f}_N, \mathbf{u}_N)\}$, h is trained by minimizing mean squared error between true and predicted values as to get optimised parameters $\boldsymbol{\theta}^*$ as:

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \ \frac{1}{N} \sum_{i=1}^{N} \| h(\mathbf{f}_i, \boldsymbol{\theta}) - \mathbf{u}_i \|_2^2 \qquad (1)$$

Performance of h (i.e., the neural network in consideration) over respective test dataset $\{(\mathbf{f}_1, \mathbf{u}_1), \ldots, (\mathbf{f}_M, \mathbf{u}_M)\}$ is measured in terms of mean absolute error which is computed for each example (e_m) and for the entire test set ($\bar{e}$) as follows:

$$
\begin{aligned}
e_m &= \frac{1}{\mathcal{F}} \sum_{i=1}^{\mathcal{F}} |h(\mathbf{f}_m)^i - \mathbf{u}_m^i|, \\
\bar{e} &= \frac{1}{M} \sum_{m=1}^{M} e_m, \\
\sigma(e) &= \sqrt{\frac{1}{M-1} \sum_{m=1}^{M} (e_m - \bar{e})^2},
\end{aligned}
\tag{2}
$$

where $\mathcal{F}$ stands for the number of dofs of the mesh. The maximum error over the entire test dataset is defined as

$$
e_{\max} = \max_{m,i} |h(\mathbf{f}_m)^i - \mathbf{u}_m^i|. \tag{3}
$$

3 Results

We validate proposed neural network frameworks on two examples, representing a 2D and a 3D problem respectively. For the 2D example, a structured mesh is considered so that all frameworks including CNN can be applied to it. Whereas for the 3D example an arbitrary unstructured mesh is considered.

3.1 Generation of hyperelastic FEM training data

As motivated in the methodology section, training datasets of non-linear displacement solutions are generated by applying random traction and body forces on the given discretisation. The number of cases generated randomly (dataset size) has been chosen large enough to generalise well to unseen arbitrary forces. The dataset is split into the training (95%) and testing (5%) part, and the pairs of input force and output displacement solutions from the training dataset are then fed to train different types of neural networks. The proposed neural network frameworks are validated on two examples, both following Neo-Hookean hyperelastic law. To avoid the divergence of the non-linear FEM solver, both traction and body forces are applied in incremental load steps. All the computations are performed using AceFEM framework (Korelc, 2002).

For the 2D case, a rectangular domain made of soft material and discretized by 8×32 mesh with 217 quad elements is considered. It is constrained at 4 corner nodes as shown in **Figure 5A**. It is subjected to traction forces of random magnitude, location, and direction in the region prescribed by the pink line. Body forces are ignored in this case. **Table 2** provides detailed information about datasets including the material properties and external force ranges used for the generation of 2D and 3D datasets. For the 2D case, a lower range of Y-direction force density is chosen since it does not contribute much to generating large deformation solutions.

In the 3D case, a continuum toy elephant model is discretised with 6,627 tetrahedron elements. It is subjected to fixed boundary conditions by constraining nodes on the bottom region of the legs. Body forces in random magnitude and direction are applied in the transverse directions (see **Figure 5B**) to generate datasets of force-displacement pairs. External tractions are ignored in this case.

3.2 Implementation details

As introduced in the methodology section, CNN and MAgNET frameworks belong to the family of U-Net architectures, while Perceiver IO leverages Transformer-style attention. It has to be noted that all the frameworks are robust, and do not need fine hyperparameter tuning.

2D case: For the CNN U-Net, a 4-level architecture with 3 max-pooling/upsampling operations is used. At each level, two convolution layers with 3×3 filters are applied with 64, 128, 256, and 256 channels at respective levels. In the case of MAgNET, a 5-level graph U-Net architecture with 4 graph pooling/unpooling operations is used. At each level 2 MAg layers (with A^2 adjacency, refer (Deshpande et al., 2022b)) are applied with 8, 16, 16, 32, and 32 channels at respective levels. For both 2D and 3D case MAgNET architectures, the seed of the first graph pooling operation is chosen by grid search, while seeds for other pooling layers are kept constant. This is done to have the maximum possible coarsened representation of the lowest-level graph. This ensures propagating boundary condition information with a minimum number of MAg operations at the lowest level. CNN U-Net is trained with a batch size of 16 for 32,000 epochs and MAgNET is trained with a batch size of 4 for 10,000 epochs.

In the case of Perceiver IO we defined 512 inputs (standing for dofs of the example) with a total embedding size of 512 following the procedure detailed in **Section 2.2**. We also selected a total of 128 latent arrays of dimension 210 for performing cross-attention in the encoder, self attention in latent space and inputs for the decoder. For the decoder's output query array we used an index dimension of 512, which defines the size of the outputs, and a channel dimension of 210 equal to the dimension size of the latents. We used a total of 3 blocks for the latent array processing, with 2 self-attention layers per block and 2 self-attention heads per layer. Both the encoder and decoder worked with 2 cross-attention heads each. The selection of these hyper-parameters was determined in a coarse exploratory fashion, with the goal of reducing the number of network parameters while maintaining its performance. Perceiver IO is trained with a batch size of 16 for 264,140 epochs.

3D case: For the MAgNET, 7-level graph U-Net architecture is used with 6 graph pooling/unpooling operations. Again at each level, 2 MAg layers (with A^2 adjacency) are applied with 6, 6, 6, 12, 12, 24, 24 channels at respective levels. The complex topology of this particular mesh demands more number graph pooling layers, this ensures propagating boundary condition information with a minimum number of MAg operations at the lowest level. On the other hand, the only change with respect to the 2D case for Perceiver IO is an increase of the input and output dimension from 512 to 5,835 in both the encoder and decoder. MAgNET architecture for this case is trained for 1200 epochs with a batch size of 4, whereas Perceiver IO is trained for 32,580 epochs with a batch size of 16.

CNN U-Net and MAgNET networks are trained using the Adam optimizer (Kingma and Ba, 2014), whereas Perceiver IO is trained using AdamW optimizer (Loshchilov and Hutter, 2017) as implemented in the original paper. CNN and MAgNET are implemented using TensorFlow (Abadi et al., 2015), while Perceiver IO is implemented using PyTorch (Paszke et al., 2019). All the implementations in this work are performed using HPC facilities of the University of Luxembourg (Varrette et al., 2014).

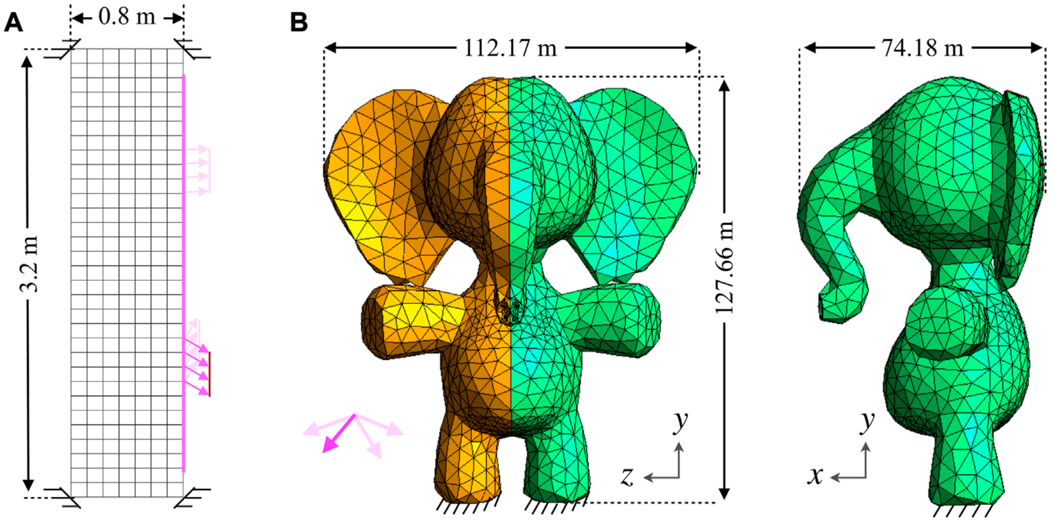

FIGURE 5
Schematics of dataset generation for **(A)** 2D example subjected to external traction forces. **(B)** 3D example subjected to external body forces. External tractions and body forces are indicated with pink arrows.

TABLE 2 Desciption of FEM datasets.

Case	Material properties (E [Pa], ν)	N. of FEM DOFs ($\mathcal{F}$)	External traction/body force density range	Dataset size (train + test)
2D domain	100, 0.3	512	$f_x = -24$ to 24 N/m, $f_y = -8$ to 8 N/m	7,124 + 372
3D elephant	3×10^6, 0.4	5,835	$b_x, b_z = -0.35$ to 0.35 N/kg, $b_y = 0$ N/kg	7,600 + 400

TABLE 3 Error metrics over the test set using the proposed NN frameworks. *M* stands for the number of test examples, and $\bar{e}$, $\sigma(e)$, e_{max} are error metrics defined in Section 2.2.

Example	Framework	M	$\bar{e}$ [m]	$\sigma(e)[m]$	e $(E)_{max}[m]$
2D	CNN	372	0.06 E-3	0.2–4	0.001
	MAgNET		0.17 E-3	0.7–4	0.021
	Perceiver IO		0.02 E-3	0.1–4	0.001
3D	MAgNET	400	8.92 E-3	1.9–3	0.307
	Perceiver IO		2.60 E-3	1.1–3	0.098

3.3 Performance on unseen examples

Proposed DNN frameworks are trained and tested on the datasets generated as illustrated in Section 3.1. The maximum nodal displacement for the 2D case is 0.35 m and for the 3D case it is 140.04 m, i.e., we compute displacements of all the nodes for every single example, and then choose particular examples for which the maximum nodal displacement is observed. Table 3 summarises the performance of neural networks on the two test datasets. It shows that all three networks are capable of predicting mechanical deformation responses with a very low error.

To compare, we observed that CNN U-NET and Perceiver IO gave lower error metrics than MAgNET for the 2D case, which has relatively low dimensional input. As the size and complexity of the mesh increased in the 3D case, both MAgNET and Perceiver IO performed well, with Perceiver IO giving slightly better error metrics.

3.4 Training and inference of DNN frameworks

First, we compare the training convergence of proposed DNN frameworks by comparing the mean square loss plots for both 2D and 3D cases. Figure 6A shows that for the relatively smaller dimension inputs as in the 2D case, both CNN and Perceiver IO observed to learn more efficiently when compared to MAgNET. Figure 6B shows that both MAgNET and perceiver are able to learn efficiently on the complex mesh data as observed in the 3D case. However, MAgNET could learn more quickly than Perceiver IO, also MAgNET can learn efficiently even with the increased mesh complexity and input size. In case of Perceiver IO, as the size and complexity of mesh further increases, it becomes less and less robust and fails to learn efficiently. We observed that Perceiver IO failed to learn on the data with input dimension higher than 10^4.

A possible interpretation of this behavior is that in the case of MAgNET topological information is externally provided through the adjacency matrix. Hence MAgNET can efficiently learn by leveraging inter-dependencies between different nodal feature values in the data. On the other hand Perceiver IO implicitly

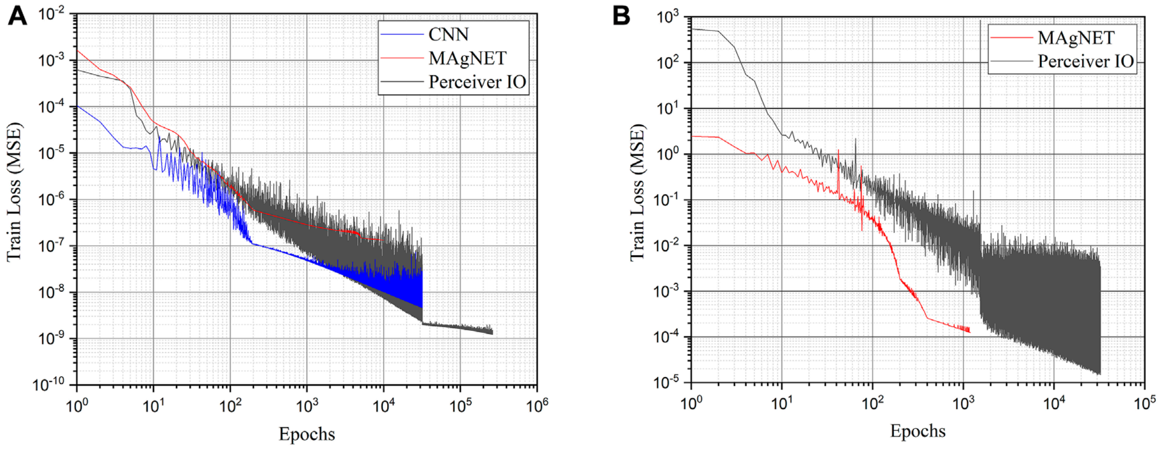

FIGURE 6
Training convergence for the proposed neural network frameworks for the **(A)** 2D case **(B)** 3D case.

TABLE 4 Comparison of training and inference times for all the three networks implemented in this work.

Example	Framework	N. Parameters (× E6)	Training time (hrs)	Inference time (s)	FEM solver time (s)
2D	CNN U-Net	4.8	18	0.021	0.6
	MAgNET	4.5	132	0.040	
	Perceiver IO	1.9	521	0.006	
3D	MAgNET	33.9	161	0.217	2.5
	Perceiver IO	4.4	312	0.006	

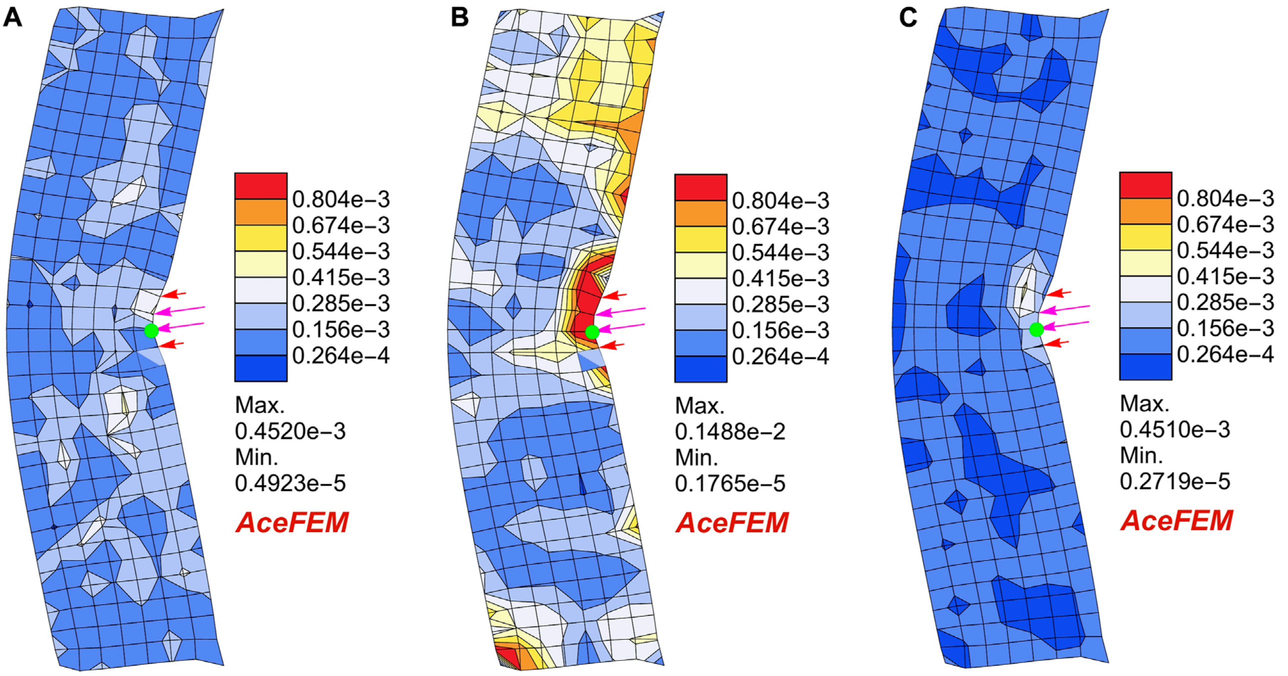

FIGURE 7
Prediction error for different neural network frameworks when compared to true FEM solution, plotted on the deformed mesh (obtained using the same framework). Force with line density of (−21.6645, −2.99384) N is applied as shown with pink arrows. True displacement of the green node is 0.35 m. Nodal error contours obtained **(A)** using CNN U-Net **(B)** using MAgNET **(C)** using Perceiver IO.

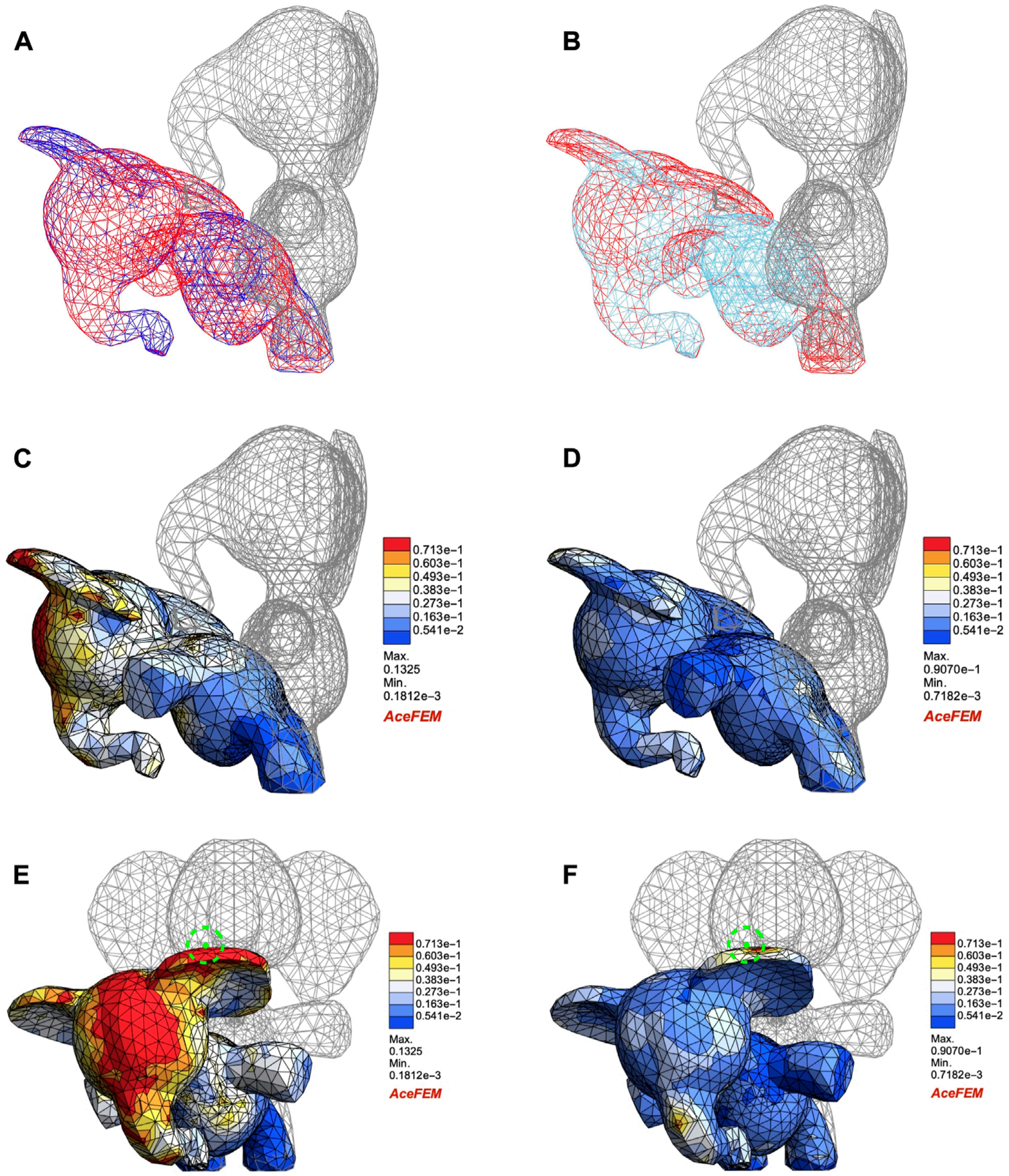

FIGURE 8
Deformation of elephant mesh subjected to external body force density (0.34, 0.0, 0.35) N/kg. First column represents MAgNET solutions while the second column represents Perceiver IO solutions. **(A,B)** Deformed meshes using MAgNET (dark blue) and Perceiver IO (sky blue) respectively, for comparison FEM mesh is presented in red. The rest position is indicated with gray mesh. **(C,D)** Side view of nodal error contours when compared to the FEM solution, plotted on the deformed meshes for MAgNET and Perceiver IO solution respectively. **(E,F)** Front view of nodal error contours for MAgNET and Perceiver IO respectively. The true displacement of the green node is 140.04 m.

learns the nodal data dependencies and as the size of the input data increases, the task to find these inter-dependencies gets more difficult. Evidence of this behavior can be seen in Figure 6, where it is clear that Perceiver IO is optimizing through a much more complex objective function with a higher density of local minimas.

Once trained, proposed DNN frameworks are fast at the inference stage while predicting unseen examples. Table 4 provides training and inference time (for a single test example) for all three frameworks, for comparison FEM solution time is also provided. In particular, Perceiver IO takes a much longer time during the training phase but is extremely fast at the inference stage. It could make predictions on both small scale (2D) and large scale (3D) inputs in almost similar time. It has to be noted that to ensure the convergence of the iterative solver, the non-linear FEM problem is solved with incremental load steps. Hence the solution time for FEM increases with the magnitude of external force. Whereas trained DNN frameworks take almost similar time at the inference stage irrespective of external force magnitudes.

3.5 Qualitative analysis of individual examples

In this section, we analyze deformations of individual examples by giving a qualitative comparison of predictions obtained using different networks. In particular, we analyze test examples with the maximum nodal displacement for 2D as well as the 3D case. In both cases, we plot nodal error contours standing for the absolute difference between the DNN prediction and the true FEM solution.

3.5.1 2D case

The analyzed example in the **Figure 7** stands for the maximum nodal displacement example in the 2D test dataset. The node indicated by the green dot has the maximum nodal displacement of 0.35 m. While the pink arrows represent corresponding nodal forces for the line density force applied on those four nodes.

Figure 7 shows absolute error counters of nodal displacements predicted using proposed DNN frameworks when compared with the FEM solution. All the proposed neural network frameworks can accurately predict the deformed mesh. Percentage prediction error (when compared to the true FEM solution) for the green node is 0.03%, 0.42%, 0.06% using CNN, MAgNET, and Perceiver IO network respectively. We observed that for the small-scale structured inputs, both Perceiver IO and CNN U-Net could make better predictions when compared to MAgNET. In case of Perceiver IO, advantage likely comes from the network leveraging its capability of learning long-range correlations more accurately. This stems from the fact that Perceiver's inputs are not constrained by any topological assumption.

3.5.2 3D case

The 3D case is aiming to demonstrate the performance for unstructured meshes, that are commonly used when dealing with real world geometries. Tetrahedron discretisation of the continuous 3D domain is irregular and the mesh topology is complex. The fixed nodes are topologically far from the tip of the elephant trunk, thus making it challenging to communicate the boundary condition information. We show that both MAgNET and Perceiver IO can learn on such complex real-world examples efficiently.

Again we consider the test example with maximum nodal displacement. **Figure 8** shows that both MAgNET (**Figures 8A, C, E**) and Perceiver IO (**Figures 8B, D, F**) solutions are able to predict non-linear mesh deformations accurately. We further analyse the absolute nodal error counters for both predictions by plotting them on the deformed meshes predicted using respective DNN frameworks. Both front (**Figures 8E, F**) and side views (**Figures 8C, D**) obtained using MAgNET and Perceiver IO solutions respectively indicate low prediction errors for both frameworks. The green node (at the tip of the ear) shown in (**Figures 8E, F**) has a maximum nodal displacement of 140.04 m for this example. The percentage prediction error when compared to the true FEM solution for this green node is 0.03%, and 0.02% using MAgNET and Perceiver IO networks respectively. Both MAgNET and Perceiver IO are observed to make efficient predictions, with Perceiver IO giving relatively low nodal errors for this demanding case. Also, owing to the lesser number of trainable parameters, Perceiver IO is much faster at the inference stage.

4 Conclusion

In this work, we demonstrated the capabilities of three promising deep neural network (DNN) frameworks for accurate and fast predictions of non-linear deformations of solid bodies. We compared their performance on two benchmark examples, in which data was generated by the finite element method. Although we only tested the frameworks for the Noe-Hoohean material model, they are compatible with more general hyperelastic models, such as Mooney–Rivlin or Ogden models. As such, they promise to be used as surrogate models for non-linear computational models in mechanics.

The comparison included two very recent DNN frameworks, MAgNET and Perceiver IO, that are naturally able to work with arbitrarily structured data at inputs/outputs, including complex finite element meshes that originate from real-world applications. The third compared framework, CNN U-Net, could only operate on grid inputs/outputs, and we suggested possible remedies to extend it to work with arbitrary unstructured meshes. When looking at prediction capabilities, especially interesting are the capabilities of the Perceiver IO network, which demonstrated to give better predictions with a lesser number of parameters, as compared to MAgNET and CNN U-Net. Additionally, the use of Perceiver IO creates a direct link to rapidly advancing research in ML and AI communities, which promises further advancements.

MAgNET and Perceiver IO are designed to be flexible in terms of the input and output structures, allowing them to potentially be applied to a wide range of problems. One possible application for these types of neural networks is in *ab initio* multi-scale modeling, which is also pursued in our team, see Hauseux et al. (2020). These methods could be used to accelerate computationally expensive accurate simulations of large atomic systems by helping to connect atomic-level simulations with the macroscopic continuum description of materials. As such, these neural networks could lead to significant strides in the field of materials science.

One of the first possible future extensions of the presented frameworks would be to incorporate the physics-informed neural network paradigm. This can be easily achieved by incorporating relevant physical laws in the optimization objective of the training procedure. Such extension can further increase the accuracy of predictions and accelerate the training procedure. Another possible extension is to consider a much wider class of phenomena and models, including buckling instabilities and more general history/time-dependent phenomena (visco-elasticity, dynamics, plasticity, etc.), which would allow tackling more challenging problems in solid mechanics, see e.g., (Vijayaraghavan et al., 2021). Going beyond mechanics, these approaches can be also adopted for a much wider range of engineering and scientific applications.

Data availability statement

The datasets presented in this study can be found in online repositories. The names of the repository/repositories and accession number(s) can be found below: https://doi.org/10.5281/zenodo.7585319, https://github.com/saurabhdeshpande93/convolution-aggregation-attention.

Author contributions

SD conceptualized and created the core structure of the manuscript with the help of JL. SD and RS carried out all the simulations presented in this work. SD wrote the main part and assembled all of the manuscript, RS contributed to multiple sections. JL and SB revised the text in detail. All authors read, discussed, and approved the final version.

Funding

This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie grant agreement No. 764644. Jakub Lengiewicz would like to acknowledge the support from EU Horizon 2020 Marie Sklodowska Curie Individual Fellowship *MOrPhEM* under Grant 800150. Stéphane Bordas, Jakub Lengiewicz and Raúl I. Sosa are grateful for the support of the Fonds National de la Recherche Luxembourg FNR grant QuaC C20/MS/14782078. Stéphane Bordas received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 811099 TWINNING Project DRIVEN for the University of Luxembourg.

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

Author disclaimer

This paper only contains the author's views and the Research Executive Agency and the Commission are not responsible for any use that may be made of the information it contains.

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., et al. (2015). *TensorFlow: Large-scale machine learning on heterogeneous systems Software available from tensorflow.org*.

Abueidda, D. W., Koric, S., Sobh, N. A., and Sehitoglu, H. (2021). Deep learning for plasticity and thermo-viscoplasticity. *Int. J. Plasticity* 136, 102852. doi:10.1016/j.ijplas.2020.102852

Aydin, R. C., Braeu, F. A., and Cyron, C. J. (2019). General multi-fidelity framework for training artificial neural networks with computational models. *Front. Mater.* 6, 61. doi:10.3389/fmats.2019.00061

Baevski, A., Zhou, Y., Mohamed, A., and Auli, M. (2020). wav2vec 2.0: A framework for self-supervised learning of speech representations. *Adv. Neural Inf. Process. Syst.* 33, 12449–12460.

Barrios, J. M., and Romero, P. E. (2019). Decision tree methods for predicting surface roughness in fused deposition modeling parts. *Materials* 12, 2574. doi:10.3390/ma12162574

Battaglia, P. W., Hamrick, J. B., Bapst, V., Sanchez-Gonzalez, A., Zambaldi, V., Malinowski, M., et al. (2018). *Relational inductive biases, deep learning, and graph networks*. arXiv preprint arXiv:1806.01261.

Bock, F. E., Aydin, R. C., Cyron, C. J., Huber, N., Kalidindi, S. R., and Klusemann, B. (2019). A review of the application of machine learning and data mining approaches in continuum materials mechanics. *Front. Mater.* 6, 110, doi:10.3389/fmats.2019.00110

Bronstein, M. M., Bruna, J., Cohen, T., and Veličković, P. (2021). *Geometric deep learning: Grids, groups, graphs, geodesics, and gauges*. arXiv preprint arXiv:2104.13478.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., et al. (2020). Language models are few-shot learners. *Adv. neural Inf. Process. Syst.* 33, 1877–1901.

Brunet, J.-N., Mendizabal, A., Petit, A., Golse, N., Vibert, E., and Cotin, S. (2019). "Physics-based deep neural network for augmented reality during liver surgery," in *Medical image computing and computer assisted intervention – miccai 2019*. Editors D. Shen, T. Liu, T. M. Peters, L. H. Staib, C. Essert, S. Zhou, et al. (Cham: Springer International Publishing), 137–145.

Bui, H. P., Tomar, S., Courtecuisse, H., Cotin, S., and Bordas, S. P. A. (2018). Real-time error control for surgical simulation. *IEEE Trans. Biomed. Eng.* 65, 596–607. doi:10.1109/TBME.2017.2695587

Butler, K. T., Davies, D. W., Cartwright, H., Isayev, O., and Walsh, A. (2018). Machine learning for molecular and materials science. *Nature* 559, 547–555. doi:10.1038/s41586-018-0337-2

Capuano, G., and Rimoli, J. J. (2019). Smart finite elements: A novel machine learning application. *Comput. Methods Appl. Mech. Eng.* 345, 363–381. doi:10.1016/j.cma.2018.10.046

Chen, A. I., Balter, M. L., Maguire, T. J., and Yarmush, M. L. (2020). Deep learning robotic guidance for autonomous vascular access. *Nat. Mach. Intell.* 2, 104–115. doi:10.1038/s42256-020-0148-7

Choi, H., Crump, C., Duriez, C., Elmquist, A., Hager, G., Han, D., et al. (2021). On the use of simulation in robotics: Opportunities, challenges, and suggestions for moving forward. *Proc. Natl. Acad. Sci.* 118, e1907856118. doi:10.1073/pnas.1907856118

Choudhary, K., DeCost, B., Chen, C., Jain, A., Tavazza, F., Cohn, R., et al. (2022). Recent advances and applications of deep learning methods in materials science. *npj Comput. Mater.* 8, 59–26. doi:10.1038/s41524-022-00734-6

Cotin, S., Delingette, H., and Ayache, N. (1999). Real-time elastic deformations of soft tissues for surgery simulation. *IEEE Trans. Vis. Comput. Graph.* 5, 62–73. doi:10.1109/2945.764872

Courtecuisse, H., Allard, J., Kerfriden, P., Bordas, S. P., Cotin, S., and Duriez, C. (2014). Real-time simulation of contact and cutting of heterogeneous soft-tissues. *Med. image Anal.* 18, 394–410. doi:10.1016/j.media.2013.11.001

De Vivo, M., Masetti, M., Bottegoni, G., and Cavalli, A. (2016). Role of molecular dynamics and related methods in drug discovery. *J. Med. Chem.* 59, 4035–4061. doi:10.1021/acs.jmedchem.5b01684

Dennler, C., Bauer, D. E., Scheibler, A.-G., Spirig, J., Götschi, T., Fürnstahl, P., et al. (2021). Augmented reality in the operating room: A clinical feasibility study. *BMC Musculoskelet. Disord.* 22, 451. doi:10.1186/s12891-021-04339-w

Deshpande, S., Bordas, S. P. A., and Lengiewicz, J. (2022b). MAgNET: A graph U-net architecture for mesh-based simulations. *arXiv*. doi:10.48550/ARXIV.2211.00713

Deshpande, S., Lengiewicz, J., and Bordas, S. P. (2022a). Probabilistic deep learning for real-time large deformation simulations. *Comput. Methods Appl. Mech. Eng.* 398, 115307. doi:10.1016/j.cma.2022.115307

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). *Bert: Pre-training of deep bidirectional transformers for language understanding*. arXiv preprint arXiv:1810.04805.

El Haber, G., Viquerat, J., Larcher, A., Ryckelynck, D., Alves, J., Patil, A., et al. (2022). Deep learning model to assist multiphysics conjugate problems. *Phys. Fluids* 34, 015131. doi:10.1063/5.0077723

Elouneg, A., Bertin, A., Lucot, Q., Tissot, V., Jacquet, E., Chambert, J., et al. (2022). *In vivo* skin anisotropy dataset from annular suction test. *Data Brief* 40, 107835. doi:10.1016/j.dib.2022.107835

Flaschel, M., Kumar, S., and De Lorenzis, L. (2021). Unsupervised discovery of interpretable hyperelastic constitutive laws. *Comput. Methods Appl. Mech. Eng.* 381, 113852. doi:10.1016/j.cma.2021.113852

Friesner, R. A. (2005). *Ab initio* quantum chemistry: Methodology and applications. *Proc. Natl. Acad. Sci.* 102, 6648–6653. doi:10.1073/pnas.0408036102

Gholamalizadeh, T., Moshfeghifar, F., Ferguson, Z., Schneider, T., Panozzo, D., Darkner, S., et al. (2022). Open-full-jaw: An open-access dataset and pipeline for finite element models of human jaw. *Comput. Methods Programs Biomed.* 224, 107009. doi:10.1016/j.cmpb.2022.107009

Guo, X., Li, W., and Iorio, F. (2016). “Convolutional neural networks for steady flow approximation,” in *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining* (New York, NY, USA: Association for Computing Machinery), 481–490. doi:10.1145/2939672.2939738

Hauseux, P., Nguyen, T.-T., Ambrosetti, A., Ruiz, K. S., Bordas, S. P. A., and Tkatchenko, A. (2020). From quantum to continuum mechanics in the delamination of atomically-thin layers from substrates. *Nat. Commun.* 11, 1651. doi:10.1038/s41467-020-15480-w

Jaegle, A., Borgeaud, S., Alayrac, J.-B., Doersch, C., Ionescu, C., Ding, D., et al. (2022). “Perceiver IO: A general architecture for structured inputs and outputs,” in *International conference on learning representations*.

Jha, D., Ward, L., Paul, A., Liao, W.-k., Choudhary, A., Wolverton, C., et al. (2018). Elemnet: Deep learning the chemistry of materials from only elemental composition. *Sci. Rep.* 8, 1–13. doi:10.1038/s41598-018-35934-y

Kim, Y., Mishra, S., Jin, S., Panda, R., Kuehne, H., Karlinsky, L., et al. (2022). “How transferable are video representations based on synthetic data?,” in *Thirty-sixth conference on neural information processing systems datasets and benchmarks track*.

Kingma, D. P., and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv*. doi:10.48550/ARXIV.1412.6980

Korelc, J. (2002). Multi-language and multi-environment generation of nonlinear finite element codes. *Eng. Comput.* 18, 312–327. doi:10.1007/s003660200028

Krokos, V., Bordas, S. P. A., and Kerfriden, P. (2022a). *A graph-based probabilistic geometric deep learning framework with online physics-based corrections to predict the criticality of defects in porous materials*. doi:10.48550/ARXIV.2205.06562

Krokos, V., Bui Xuan, V., Bordas, S. P. A., Young, P., and Kerfriden, P. (2022b). A bayesian multiscale cnn framework to predict local stress fields in structures with microscale features. *Comput. Mech.* 69, 733–766. doi:10.1007/s00466-021-02112-3

Le, T. A., Baydin, A. G., Zinkov, R., and Wood, F. (2017). Using synthetic data to train neural networks is model-based reasoning. In *2017 international joint conference on neural networks (IJCNN)* (IEEE), 3514–3521.

Liu, Y., Zhao, T., Ju, W., and Shi, S. (2017). Materials discovery and design using machine learning. *J. Materiomics* 3, 159–177. doi:10.1016/j.jmat.2017.08.002

Loshchilov, I., and Hutter, F. (2017). Decoupled weight decay regularization. *arXiv*. doi:10.48550/ARXIV.1711.05101

Mao, Z., Jagtap, A. D., and Karniadakis, G. E. (2020). Physics-informed neural networks for high-speed flows. *Comput. Methods Appl. Mech. Eng.* 360, 112789. doi:10.1016/j.cma.2019.112789

Mazier, A., Ribes, S., Gilles, B., and Bordas, S. P. (2021). A rigged model of the breast for preoperative surgical planning. *J. Biomechanics* 128, 110645. doi:10.1016/j.jbiomech.2021.110645

McFall, K., and Mahan, J. (2009). Artificial neural network method for solution of boundary value problems with exact satisfaction of arbitrary boundary conditions. *IEEE Trans. neural Netw.* 20, 1221. 1233. doi:10.1109/tnn.2009.2020735

Mendizabal, A., Márquez-Neila, P., and Cotin, S. (2019). Simulation of hyperelastic materials in real-time using deep learning. *Med. Image Anal.* 59, 101569. doi:10.1016/j.media.2019.101569

Mianroodi, J. R., H Siboni, N., and Raabe, D. (2021). Teaching solid mechanics to artificial intelligence—A fast solver for heterogeneous materials. *Npj Comput. Mater.* 7, 99–10. doi:10.1038/s41524-021-00571-z

Odot, A., Haferssas, R., and Cotin, S. (2022). Deepphysics: A physics aware deep learning framework for real-time simulation. *Int. J. Numer. Methods Eng.* 123, 2381–2398. doi:10.1002/nme.6943

Oishi, A., and Yagawa, G. (2017). Computational mechanics enhanced by deep learning. *Comput. Methods Appl. Mech. Eng.* 327, 327–351. doi:10.1016/j.cma.2017.08.040

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., et al. (2019). “Pytorch: An imperative style, high-performance deep learning library,” in *Advances in neural information processing systems* (Curran Associates, Inc.), 32, 8024–8035.

Pfaff, T., Fortunato, M., Gonzalez, A., and Battaglia, P. (2021). “Learning mesh-based simulation with graph networks,” in *International conference on learning representations*.

Pfeiffer, M., Riediger, C., Weitz, J., and Speidel, S. (2019). Learning soft tissue behavior of organs for surgical navigation with convolutional neural networks. *Int. J. Comput. Assisted Radiology Surg.* 14, 1147–1155. doi:10.1007/s11548-019-01965-7

Rupp, M., Tkatchenko, A., Müller, K.-R., and von Lilienfeld, O. A. (2012). Fast and accurate modeling of molecular atomization energies with machine learning. *Phys. Rev. Lett.* 108, 058301. doi:10.1103/PhysRevLett.108.058301

Rus, D., and Tolley, M. T. (2015). Design, fabrication and control of soft robots. *Nature* 521, 467–475. doi:10.1038/nature14543

Samaniego, E., Anitescu, C., Goswami, S., Nguyen-Thanh, V., Guo, H., Hamdia, K., et al. (2020). An energy approach to the solution of partial differential equations in computational mechanics via machine learning: Concepts, implementation and applications. *Comput. Methods Appl. Mech. Eng.* 362, 112790. doi:10.1016/j.cma.2019.112790

Schleder, G. R., Padilha, A. C., Acosta, C. M., Costa, M., and Fazzio, A. (2019). From DFT to machine learning: Recent approaches to materials science–a review. *J. Phys. Mater.* 2, 032001. doi:10.1088/2515-7639/ab084b

Schmidt, J., Marques, M. R., Botti, S., and Marques, M. A. (2019). Recent advances and applications of machine learning in solid-state materials science. *npj Comput. Mater.* 5, 83–36. doi:10.1038/s41524-019-0221-0

Schütt, K., Kindermans, P.-J., Sauceda Felix, H. E., Chmiela, S., Tkatchenko, A., and Müller, K.-R. (2017). “Schnet: A continuous-filter convolutional neural network for modeling quantum interactions,” in *Advances in neural information processing systems*. Editors I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, et al. (Curran Associates, Inc.), 30.

Schütt, K. T., Arbabzadah, F., Chmiela, S., Müller, K. R., and Tkatchenko, A. (2017). Quantum-chemical insights from deep tensor neural networks. *Nat. Commun.* 8, 13890–13898. doi:10.1038/ncomms13890

Strönisch, S., Meyer, M., and Lehmann, C. (2022). “Flow field prediction on large variable sized 2d point clouds with graph convolution,” in *Proceedings of the platform for advanced scientific computing conference* (New York, NY, USA: Association for Computing Machinery). PASC ’22. doi:10.1145/3539781.3539789

Unke, O. T., Chmiela, S., Sauceda, H. E., Gastegger, M., Poltavsky, I., Schütt, K. T., et al. (2021). Machine learning force fields. *Chem. Rev.* 121, 10142–10186. doi:10.1021/acs.chemrev.0c01111

Varrette, S., Bouvry, P., Cartiaux, H., and Georgatos, F. (2014). Management of an academic hpc cluster: The ul experience. doi:10.1109/HPCSim.2014.6903792

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., et al. (2017). Attention is all you need. *Adv. neural Inf. Process. Syst.* 30.

Vijayaraghavan, S., Wu, L., Noels, L., Bordas, S. P. A., Natarajan, S., and Beex, L. A. A. (2021). Neural-network acceleration of projection-based model-order-reduction for finite plasticity: Application to RVEs. *arXiv*. doi:10.48550/ARXIV.2109.07747

Vlassis, N. N., Ma, R., and Sun, W. (2020). Geometric deep learning for computational mechanics part i: Anisotropic hyperelasticity. *Comput. Methods Appl. Mech. Eng.* 371, 113299. doi:10.1016/j.cma.2020.113299

Voulodimos, A., Doulamis, N., Doulamis, A., and Protopapadakis, E. (2018). *Deep learning for computer vision: A brief review*. Computational intelligence and neuroscience, 2018.

Weerasuriya, A. U., Zhang, X., Lu, B., Tse, K. T., and Liu, C. (2021). A Gaussian process-based emulator for modeling pedestrian-level wind field. *Build. Environ.* 188, 107500. doi:10.1016/j.buildenv.2020.107500

Wirtz, D., Karajan, N., and Haasdonk, B. (2015). Surrogate modeling of multiscale models using kernel methods. *Int. J. Numer. Methods Eng.* 101, 1–28. doi:10.1002/nme.4767

Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhudinov, R., et al. (2015). “Show, attend and tell: Neural image caption generation with visual attention,” in *International conference on machine learning* (PMLR), 2048–2057.

Zakutayev, A., Wunder, N., Schwarting, M., Perkins, J. D., White, R., Munch, K., et al. (2018). An open experimental database for exploring inorganic materials. *Sci. data* 5, 1–12. doi:10.1038/sdata.2018.53