Artificial Intelligence Review (2024) 57:315
https://doi.org/10.1007/s10462-024-10931-y

A review of graph neural network applications in mechanics-related domains

Yingxue Zhao[1] · Haoran Li[1] · Haosu Zhou[1] · Hamid Reza Attar[1] · Tobias Pfaff[2] · Nan Li[1]

Accepted: 28 August 2024 / Published online: 4 October 2024

Abstract
Mechanics-related tasks often present unique challenges in achieving accurate geometric and physical representations, particularly for non-uniform structures. Graph neural networks (GNNs) have emerged as a promising tool to tackle these challenges by adeptly learning from graph data with irregular underlying structures. Consequently, recent years have witnessed a surge in complex mechanics-related applications inspired by the advancements of GNNs. Despite this process, there is a notable absence of a systematic review addressing the recent advancement of GNNs in solving mechanics-related tasks. To bridge this gap, this review article aims to provide an in-depth overview of the GNN applications in mechanics-related domains while identifying key challenges and outlining potential future research directions. In this review article, we begin by introducing the fundamental algorithms of GNNs that are widely employed in mechanics-related applications. We provide a concise explanation of their underlying principles to establish a solid understanding that will serve as a basis for exploring the applications of GNNs in mechanics-related domains. The scope of this paper is intended to cover the categorisation of literature into solid mechanics, fluid mechanics, and interdisciplinary mechanics-related domains, providing a comprehensive summary of graph representation methodologies, GNN architectures, and further discussions in their respective subdomains. Additionally, open data and source codes relevant to these applications are summarised for the convenience of future researchers. This article promotes an interdisciplinary integration of GNNs and mechanics and provides a guide for researchers interested in applying GNNs to solve complex mechanics-related tasks.

Keywords Machine learning · Artificial intelligence · Graph neural networks · Mechanics-related applications · Graph representation methodologies · GNN architectures

✉ Nan Li
n.li09@imperial.ac.uk

[1] Dyson School of Design Engineering, Imperial College London, London, UK

[2] Google DeepMind, London, UK

1 Introduction

1.1 Preamble to machine learning approaches in mechanics-related domains

Mechanics, as a fundamental discipline in the physical sciences, plays a pivotal role in various applications, for example, solid mechanics, fluid mechanics, and interdisciplinary mechanics-related domains. Solid mechanics delves into the motion and deformation of solid materials under external forces (Wang and Qin 2020). Fluid mechanics investigates the behaviours of fluids in stationary or moving states. Interdisciplinary mechanics-related applications tackle complex or multi-disciplinary systems drawing upon mechanics principles alongside other domains such as earth science, biology, or thermodynamics.

The mechanics-related domains often involve complex mathematical descriptions due to nonlinear, multiscale and multiphysics coupling phenomena (Groen et al. 2012; Lebon and Ramière 2023). Consequently, deriving analytical solutions for these mathematical models that apply to real-world scenarios becomes challenging. Conventionally, experiments are often conducted to provide real-world observation of physical behaviours, offering insights into new phenomena, and devise practical solutions to complex problems. However, experimental approaches are often resource-expensive and time-consuming. To mitigate this, numerical simulations provide a cost-effective means to address complex challenges, enabling iterative assessments without the need for costly experimental prototyping trials and offering the potential for parallel execution (Behbahani et al. 2009; Pietrzyk 2000; Pinto et al. 2017; Sharma and Kalamkar 2016). While numerical simulations have long played a pivotal role in offering robust and validated methods to solve mechanics-related tasks, they come with their own set of challenges. As mechanics problems become increasingly complex and non-linear with the advancement of mechanics-related technologies and use scenarios, the computational efforts required can become prohibitively high. Therefore, integrating novel, cutting-edge machine learning (ML) techniques with state-of-the-art numerical simulations or experiments holds the potential to significantly enhance the efficiency of handling mechanics-related tasks.

Machine learning is a subset of artificial intelligence (AI) that involves the development of algorithms and statistical models enabling computers to identifying patterns and correlations between input and output features, typically in a data-driven way. In recent years, ML techniques, particularly the artificial neural network (ANN) models, have seen widespread adoption across various mechanics-related domains, including solid mechanics (Kumar and Kochmann 2022), fluid mechanics (Brunton et al. 2020), and interdisciplinary domains (Guo et al. 2022). These ML models provide valuable insights, aiding in navigating complex parameter spaces inherent in mechanics-related tasks. By leveraging data derived from established physical laws, machine learning models can implicitly learn the underlying physics, enabling rapid and accurate predictions.

1.2 Typical workflow for applying machine learning to mechanics-related tasks

A typical schematic for learning mechanics-related tasks by integrating ML techniques is illustrated in Fig. 1. The typical ML process for mechanics-related tasks starts with a clear definition of the mechanics-related problem, this can be categorised into prediction, classification, and inverse design problems, such as predicting the stress field of a cantilever beam under uniformly distributed loading (Fu et al. 2023), classifying the

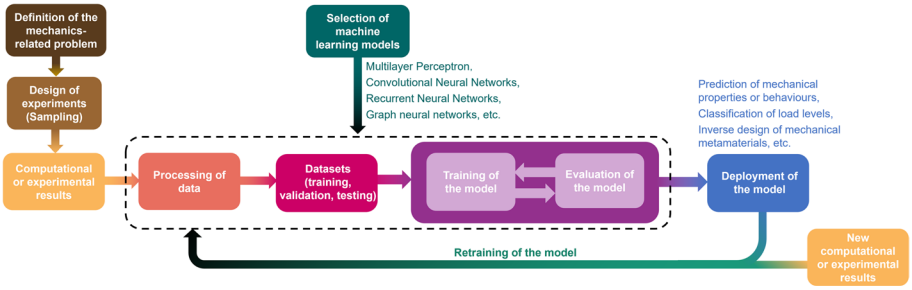

Fig. 1 Typical schematic for learning mechanics-related tasks

load levels of architected metamaterials (Guo and Buehler 2020), and navigating the design space exploration of metamaterials (Zheng et al. 2023). The selection of suitable ML models is guided by the nature of the mechanics-related problem, with a detailed discussion of their advantages and limitations provided in Sect. 1.3. For instance, convolutional neural networks (CNNs) are commonly used for image-based problems due to their excellent capability in learning and predicting from image data (Afshar et al. 2019). For mechanical systems where the focus is on the evolution of system behaviours over time, recurrent neural networks (RNNs) are adopted for their ability to learn from long-range temporal data (Zhou et al. 2019). Additionally, graph neural networks (GNNs) are utilised for problems involving data that can be represented as graphs, such as the connectivity and interactions in a finite element (FE) mesh (Pfaff et al. 2020).

In the meantime, the design of experiments is conducted to develop training, validation, and test datasets. Sampling techniques such as Latin hypercube sampling (Iman et al. 1980) can be adopted to effectively explore the input parameter space, ensuring that the sampling points are well distributed across the entire design space of input variables. Based on the sampled variable values, numerical simulations or experiments are carried out to generate raw data. These raw data then undergo data processing step to meet the input and output data shape requirements of ML models. During this phase, feature engineering may be applied, where relevant indicators for the mechanical system are selected based on domain knowledge. For example, in a typical fluid mechanics problem involving the prediction of aerofoil flow field, the inputs of the ML model commonly include the geometric features of the aerofoil as well as flow conditions such as the angle of attack and freestream velocity. The output indicators are commonly velocity or pressure fields. Additionally, the data processing step may involve normalisation and the application of padding to format the data optimally for efficient model training.

Once the ML model is built using open-source ML platforms, the ML model is then trained and evaluated. There are two main types of training approaches namely data-driven, and physics-informed training. In the data-driven training approaches, the training dataset is used to fit the model, allowing it to learn the relationships within the data. Whereas the physics-informed training approaches integrate physical laws directly into the training process, potentially enhancing the ability of ML to learn physical principles without relying heavily on large training datasets (Nguyen-Thanh et al. 2020). In the model evaluation phase, the validation dataset is employed to tune hyperparameters and make decisions regarding model architecture, helping to prevent overfitting by providing an independent check of performance during training. Cross-validation methods

such as k-fold validation (James et al. 2021), could be employed to rigorously assess the generalisability of the ML model to unseen data. The test data that are completely separate from both the training and validation datasets are used to evaluate the final ML model performance, providing an unbiased assessment of its generalisability to new, unseen data.

The model deployment phase involves implementing the final tested ML model into practical applications, enabling it to provide solutions to the previously defined mechanics-related problem. It is important to note that after model deployment, the model may require retraining or fine-tuning as new simulation or experimental data becomes available. This ensures that the model remains accurate and effective, adapting to new conditions or leveraging transfer learning for related tasks. Incorporating a retraining process is often crucial for maintaining the model's performance in real-world applications.

1.3 Evolution of popular machine learning approaches in mechanics-related tasks

As mentioned in the previous section, in recent decades, ML techniques, particularly the ANN models, have seen widespread adoption across various mechanics-related domains. According to Janiesch et al. (2021) and Goodfellow et al. (2016), an ANN is a computational model that mimics the functionalities of neurons of the human brain that can automatically learn and extract the features detection in every hidden layer. The features extracted from each layer are usually in a hierarchical way as the number of layers gets deeper. Based on the number of layers in the ANN model, the ANN algorithms could be hierarchically summarised into shallow neural networks and deep neural networks (DNNs), as illustrated in Fig. 2.

Shallow ANNs are typically referred to as ANNs with no more than two hidden layers. Shallow multilayer perceptrons (MLPs) are commonly employed in various mechanics applications due to their simplicity and interpretability. The MLP is one of the most fundamental neural network architectures, where every neuron in a layer is connected to every neuron in the preceding and succeeding layers, creating a densely interconnected matrix of weights between the layers. For instance, Narayanasamy and Padmanabhan (2012) used an

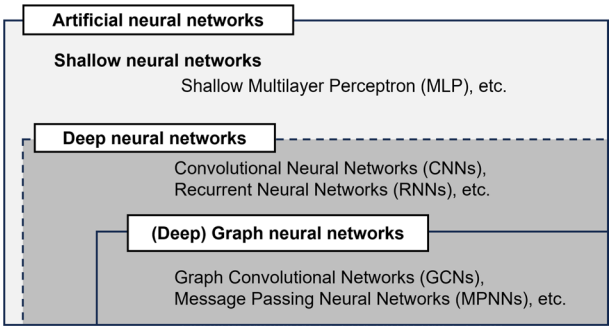

Fig. 2 Venn diagram of popular ANN algorithms that are commonly adopted in mechanics-related tasks. Typical examples for each ANN class are provided. DNNs are more expressive and can model complex systems at the cost of interpretability. Therefore, the gradient of background shading (ranging from light to dark grey) symbolises this increase in modelling power and decrease in interpretability across the ANN classes. Since there has not yet been a versatile demarcation of DNNs and shallow neural networks in the literature (Janiesch et al. 2021; Schmidhuber 2015), a dashed line is adopted to identify the boundary between the two classes (Adapted from Janiesch et al. (2021))

MLP model with two hidden layers to predict the springback angle in an air-bending process. Al-Jarrah and AL-Oqla (2022) used a shallow MLP to predict the mechanical properties of green fibres. Nakamura et al. (2022) compared the abilities of linear stochastic estimation and a shallow MLP with a single hidden layer by considering two canonical fluid flow regression problems. They discovered that the shallow MLP was more resistant to noisy perturbations and uncertainties in data.

However, when dealing with complex and non-linear mechanics-related tasks, MLPs also face significant limitations. Firstly, MLPs are characterised by their fully connected architecture, which implies that they lack the inherent ability to capture spatial and temporal structures in the data. In mechanics-related tasks, the prediction indicators often involve physical fields such as stress, strain, or velocity, which exhibit strong spatial correlations. When using MLPs to process such data, especially for complex structures with intricate geometries, they cannot effectively distinguish and utilise the local structural information. Moreover, due to their small number of hidden layers, shallow MLPs are usually less capable of recognising or modelling complex hierarchical patterns in mechanical systems.

In recent years, DNNs have become widely used in mechanics-related tasks. DNNs have deeper hidden layers that enable more detailed feature learning, making them highly advantageous in handling large-scale and noisy datasets, which are characteristic of many mechanics-related tasks. Common DNNs applied in mechanics-related tasks include CNNs and RNNs. RNNs are specifically designed for processing sequence data such as signal processing. Unlike traditional feedforward neural networks, RNNs have connections that loop back on themselves, allowing them to maintain a memory of previous inputs by passing information from one timestep to the next (Lipton et al. 2015). In mechanics-related tasks, RNNs are particularly valuable for modelling and predicting the behaviour of mechanical systems that evolve over time. For example, Zhou et al. (2019) used a deep RNN model incorporating two long short-term memory (LSTM) layers and one bidirectional LSTM layer to identify the impact load of nonlinear structures. However, while RNNs excel at capturing temporal dependencies, their inputs are often well encapsulated to a sequence of scalar values. Typical RNNs do not include layers specifically designed to handle spatial information and therefore may fall short in addressing spatial relationships, which are frequently encountered in mechanics-related tasks. For example, spatial relationships could be the displacement propagation through a structure when a force is applied at a specific point (Black and Najafi 2022), or the gradient of temperature across a material during heat transfer (Mozaffar et al. 2021).

CNNs are typically deep learning models that consist of multiple layers of interconnected neurons or grid pixels. The convolutional layer is the fundamental element of CNNs, where a series of kernels is applied to the input image, allowing the network to extract features like the geometric shapes of the mechanical system. These extracted feature maps are then passed on to other layers, such as pooling layers or additional convolutional layers that facilitate the network in learning intricate relationships between inputs and outputs. Therefore, CNNs are well designed for detecting spatial patterns of regular grid structures. In the solid mechanics domain, CNNs have been successfully adopted to predict various physical fields, including heat transfer (Tamaddon-Jahromi et al. 2020), stress distributions (Bolandi et al. 2022b, 2022a; Kantzos et al. 2019; Khadilkar et al. 2019; Liang et al. 2018; Nie et al. 2019; Spruegel et al. 2017), material deformations (Attar et al. 2022, 2021a, 2023, 2021b; Xu et al. 2021; Zhou et al. 2021; Zhu and Li 2023), and topology optimisations (Attar et al. 2022, 2023). Meanwhile, in the fluid mechanics domain, CNNs have found vast applications in turbulence modelling (Kutz 2017), flow visualisation and analysis (Morimoto et al. 2021; Rabault et al. 2017), aerodynamics performance prediction (Afshar et al. 2019;

Viquerat and Hachem 2020; Zhang et al. 2017), flow disturbance detection and sensor optimisation (Hou et al. 2019; Provost et al. 2020).

However, CNNs also face limitations when addressing certain mechanics-related tasks. This is because CNNs are inherently tied to Eulerian representations, while many mechanics-related tasks benefit from Lagrangian representations. For instance, when modelling the deformation of a cantilever beam structure under prescribed boundary conditions (Black and Najafi 2022), the use of CNNs can be less effective due to their inability to naturally track the movement and deformation of material points within the structure. Moreover, CNNs are often inefficient when dealing with irregular grids, which is essential for most mechanics-related tasks. For instance, higher spatial resolution is often allocated to areas where increased precision is necessary, such as around the tip of a crack or the wing of an aerofoil. However, when using CNNs with regular grids to process such inputs, the fine resolution required in these critical areas may be lost. As a result, the detailed features in these regions can be inadequately represented, leading to less faithful representation of the mechanical system. Furthermore, CNNs suffer from low scalability issues. CNN models usually require fixed-sized input images because they carry out convolutional operations with fixed parameters, such as filter size and stride. When the input images in a training dataset are of different sizes, additional image pre-processing methods are required, which can be highly time-consuming and inefficient. Additionally, CNNs are not inherently permutation invariant. This means that, without extensive data augmentation techniques such as rotation, translation, and flipping, CNNs may struggle to recognise features that appear in different orientations or positions within the input grid. This lack of permutation invariance can further limit the generalisability of model, particularly when the mechanics-related tasks involve structures or phenomena that can appear in varied spatial configurations.

To address these abovementioned limitations of CNNs, GNNs have been introduced as a specialised type of neural network designed for processing graph data that are commonly associated with mechanics-related tasks. In this context, a "graph" refers to a structure composed of nodes (vertexes) and their interconnections (edges) (Zhou et al. 2020). To illustrate, graphs in mechanics-related tasks can be locally connected through FE meshes, where the mesh nodes represent graph nodes, and the mesh edges represent graph edges. GNNs operate by aggregating and updating node embeddings through multiple layers, with each node gathering information from its neighbours through the edges (Veličković 2023; Zhou et al. 2020). This makes GNNs highly adaptable for learning from irregular data, akin to that found in mechanical domains.

GNNs can directly handle irregular grid information without converting it into a regular grid, as required by CNN architectures. This flexibility avoids the loss of information due to data conversion and allows GNNs to more accurately capture complex geometries in mechanical systems. Moreover, GNNs offer flexibility in graph representations, allowing researchers to freely define the form of nodes and edges based on specific mechanics tasks. For example, in a beam problem, nodes in the graph can represent element integration points if the focus is on stress and strain, or mesh nodes if the interest lies in nodal displacement or velocity. This flexibility allows GNN models to effectively capture the complex relationships between these graph nodes in a manner highly relevant to the mechanics-related task at hand. Furthermore, while the convolution operation of CNNs is usually limited to a local neighbourhood, GNNs can span across more distant nodes, allowing information aggregation as long as there are edges between two nodes. Furthermore, in GNNs, graphs for mechanics-related tasks are locally connected through FE meshes and can be processed in a permutation-invariant manner (Black and Najafi 2022; Pfaff et al.

2020; Sanchez-Gonzalez et al. 2020). This setup facilitates local computations, allowing the models to learn local rules that are inherently more robust to geometries outside the training regime. Several GNN models have demonstrated improved generalisability to data that, while still adhering to the same underlying physical laws, may appear out-of-distribution to human observers (Black and Najafi 2022; Meyer et al. 2022; Pfaff et al. 2020; Sanchez-Gonzalez et al. 2020; Shao et al. 2023).

1.4 General introduction to GNNs and their applications

1.4.1 General introduction to GNNs

Sperduti and Starita (1997) pioneered the application of neural networks to graph data with an innovative approach centred on directed acyclic graphs (DAGs). Based on this motivation, Scarselli et al. (2009) proposed the first GNN model, which is capable of directly processing graphs. This allows handling a broader range of data types, such as cyclic graphs, thus representing a significant advancement over previous approaches restricted to DAGs.

Inspired by the success of CNNs, researchers have developed techniques to adapt the principle of convolution operation to graph data. Monti et al. (2017) proposed a generalised CNN architecture for non-Euclidean data, which performs convolution operations on graphs and manifolds.

The convolution operation on graph data can be performed using either spectral or spatial approaches. The spectral approaches process graph signals in the Fourier domain and performs graph convolution, whereas the spatial approaches convolve directly in the graph domain using information from the neighbourhood. The continual refinement of these methodologies has led to state-of-the-art GNNs. This paper will focus on the GNN frameworks that utilise graph convolution as a propagation module because they are the most widely used frameworks in mechanics-related domains. More details on the GNN convolution operations will be introduced in Sect. 2.

1.4.2 Overview of GNN applications in mechanics and various domains

GNNs have been used to tackle a wide range of machine-learning tasks on graph-structured data due to their unique ability to capture the intricate dependencies between nodes and edges in graphs. Typical graph learning tasks can be classified into node-level, edge-level, and graph-level tasks. Node-level tasks include node classification, node regression, and node clustering. Node regression tasks predict the node features for unlabelled nodes; node classification involves assigning labels to individual nodes based on their attributes and the structure of their neighbourhood; node clustering aims to group nodes with analogous features or connectivity patterns together. Edge-level tasks include edge classification, determining the types of interaction between nodes, and link prediction, which predicts the likelihood of edge presenting between nodes. Graph-level tasks involve the prediction of the entire graph, including graph classification which classifies the graphs, and graph regression which assigns a numerical value to a graph. Node features predictions are the most commonly applied tasks for mechanics-related domains, whereas the other types of tasks are more frequently utilised in various other fields.

Figure 3 illustrates a typical implementation of GNNs in solid mechanics using a cantilever beam as an example. The workflow begins with data collection, either based on experiments or computational simulations such as FE analyses. The extracted features are

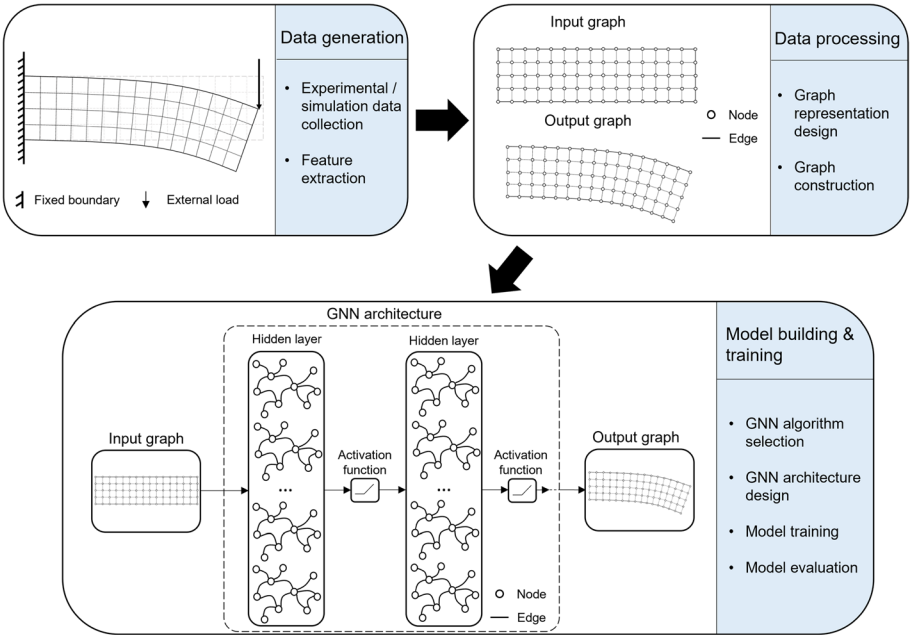

Fig. 3 Illustration of a general workflow for implementing GNNs in the solid mechanics domain—a cantilever beam example

then used to construct graph data. In the example, graphs are used to represent FE meshes where graph nodes and edges represent mesh nodes and the connectivity between nodes, respectively. The node features, or optionally edge features, need to be carefully defined to sufficiently describe the system. After preparing the graph data, the appropriate GNN algorithms are selected based on the characteristics of the specific problems. As a node regression task, the GNN architecture takes in the input graph and returns an updated graph with new node features. Details on different GNN algorithms are explained in Sect. 2. After constructing the GNN architecture, a suitable training strategy needs to be designed, including defining an appropriate loss function, and tuning hyper-parameters like batch size or learning rate. A well-trained model will then be capable of predicting the deformation of the cantilever beam given the input conditions. There are many other types of tasks that GNN has been used to solve, which will be covered in detail in Sect. 3.

The realm of GNNs has been extensively covered in existing general review papers, offering comprehensive insights into the algorithms and architectures (Abadal et al. 2022; Bacciu et al. 2020; Georgousis et al. 2021; Veličković 2023; Zhou et al. 2022). These papers not only provided the state-of-the-art taxonomies on GNN algorithms but have also shed light on the potential future directions for GNN development. Furthermore, several in-depth review papers have delved into specific applications, showcasing representative use cases and bridging the gap between theoretical foundations and practical implementations of GNN architectures (Asif et al. 2021; Gupta et al. 2021; Liang et al. 2022; Wu et al. 2021; Zheng et al. 2022; Zhou et al. 2020). Initially, GNNs gained promise in diverse domains such as image processing, natural language processing, and speech recognition

(Adamczyk 2022; Kipf and Welling 2016; Luo and Mesgarani 2019; Malekzadeh et al. 2021; Mandelli and Berretti 2022; Warey and Chakravarty 2022; Zijian Wang 2022). Then, their potential has progressively extended to engineering practices, for instance, spatial–temporal GNNs have been extensively employed in traffic forecasting applications such as traffic flow predictions (Bui et al. 2022; Jiang and Luo 2022; Ye et al. 2022). In the field of bioinformatics, the applications of GNNs in drug design and drug-target interactions have been summarised in dedicated review papers (Li et al. 2023b; Xiong et al. 2021; Yi et al. 2022; Zhang et al. 2021, 2022). Moreover, GNNs have been successfully utilised in predicting the molecular properties of materials (Reiser et al. 2022; Wieder et al. 2020), leveraging the ability of graph data to represent molecules and chemical bonds as nodes and edges. The state-of-the-art approaches of GNN have also found practical implementations in internet recommender systems, with several review papers highlighting their applications and future directions in this domain (Deng 2022; Gao et al. 2023; Wang et al. 2020; Wu et al. 2023). Furthermore, review papers covering other fields of applications, such as text classification, electronic design and power systems, have also contributed to our understanding of GNNs (Liao et al. 2022; Lopera et al. 2021; Malekzadeh et al. 2021).

1.5 Summary and main contributions of this paper

The main objective of this review article is to provide a comprehensive survey of GNN applications in mechanics-related domains, along with identifying key challenges and potential future directions in the corresponding domains. We include a summary of recent GNN advancements in solid mechanics, fluid mechanics and interdisciplinary mechanics-related domains for readers to gain comprehensive understanding of the application of GNNs across these domains. Our main contributions are:

- Summarised and illustrated the fundamental GNN algorithms that are commonly applied in mechanics-related domains.
- Provided a review of GNN applications in different mechanics-related domains, including various graph representation methods, novel GNN models, and further discussions.
- Summarised open data and source code of GNN models from the literature, serving as a repository for researchers interested in applying GNNs in the corresponding application subdomains.

This review article is organised as follows: Sect. 2 reviews the preliminaries and notations of the graph theory and the fundamental algorithms of GNNs. Section 3 summarises the GNNs applications in mechanics, classified into solid mechanics, fluid mechanics and interdisciplinary mechanics-related domains. The graph representation methods, GNN models, further discussions regarding challenges and future works are discussed for each mechanics subdomain. To facilitate future research, the open data and source codes provided by the literature in Sect. 3 are summarised in Sect. 3. Finally, Sect. 5 represents the conclusions and overall future directions for GNN applications in mechanics-related domains.

2 Fundamentals of graph neural networks

This section provides an overview of the fundamental concepts related to GNNs, including conventional notations used to describe graph data and neural network building blocks. Some relevant architectures of GNN building blocks are then discussed and categorised.

2.1 Conventional notations for GNNs

In general, a graph G can be represented by $G = (\mathbf{u}, V, E)$, where $\mathbf{u}$ is a global graph level feature, $V = \{v_1, v_2, \ldots, v_N\}$ and $E = \{e_1, e_2, \ldots, e_M\}$ denote sets of N nodes and M edges respectively. The adjacency matrix $\mathbf{A}_G$, which is an $n \times n$ matrix, describes the connection status of G. The component $\mathrm{A}_{ij} = 1$ if nodes v_i and v_j are connected by an edge, otherwise $\mathrm{A}_{ij} = 0$. For a special case when self-loops are present, i.e., an edge connecting a node to itself, the corresponding diagonal component of the adjacency matrix will be 1. In an undirected graph, $(v_i, v_j) = (v_j, v_i)$ for any edges and the adjacency matrix $\mathbf{A}_G$ is symmetric. Whereas in a directed graph, edges are orientated from one node to another and are irreversible. This adjacency matrix that is important for feature propagation in graph convolution operations which will be detailly explained in the following section. A directed edge $e_k = (v_{s_k}, v_{r_k}) \in E$ defines the edge connecting two nodes, specifically the sender node s_k and the receiver node r_k. The set of neighbouring nodes of a node v_i can be denoted by $N_G(v_i)$, and the degree of the node $d_G(v_i) = |N_G(v_i)|$ is the number of neighbouring nodes. The degree matrix $\mathbf{D}_G$ is a diagonal matrix with $\mathrm{D}_{ii} = d_G(v_i)$, it is essential for normalisation of the feature propagation process which will be discussed in the Sect. 2.3.3. The Laplacian matrix, which is a useful measure in the spectral graph theory, can be defined as $\mathbf{L}_G = \mathbf{D}_G - \mathbf{A}_G$. This provides insightful information of the structure of a graph, for instance, the smallest eigenvalue of the Laplacian indicates how well a graph is connected. Graph data also contains node and edge features $\mathbf{x}_v \in \mathbf{R}^{p_v}$ and $\mathbf{x}_e \in \mathbf{R}^{p_e}$, where p_v and p_e are the dimensions of node and edge feature vectors respectively. Matrices $\mathbf{X}_V \in \mathbf{R}^{N \times p_v}$ and $\mathbf{X}_E \in \mathbf{R}^{M \times p_e}$ respectively denote the node and edge feature matrices. Within a GNN, $\mathbf{H}^{(l)}$ and $\mathbf{h}^{(l)}$ denote the hidden feature matrix and vector on the l th layer of the neural network, where $\mathbf{H}^{(0)} = \mathbf{X}$ is the input feature matrix.

2.2 Convolution operation on graph data

Although different convolution operators aggregate and propagate information in different manners, the principles of graph convolution operations are similar to conventional CNNs. An image can be seen as a specially constrained case of graph data following Euclidean structure. Each pixel in an image can be considered a node containing node features like greyscale values or colours, and the connections between the neighbouring pixels represent the spatial relationships. During convolution operation, the features on neighbouring pixels are aggregated to the centre pixel and propagate to the next layer. Following similar principle, graph convolution operation is a more general procedure that can be applied to non-Euclidean graph data, allowing for more general applications beyond traditional image data. Figure 4 illustrates the difference between 2D image convolutional operation and graph convolutional operation.

The convolution operation on graph data can be divided into two categories: spectral approaches and spatial approaches. The spectral approaches involve convolving graph signals

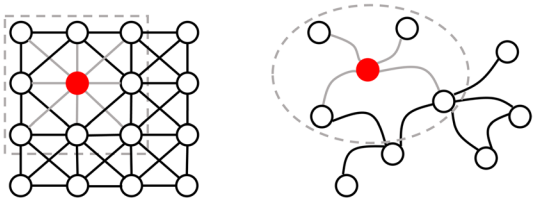

Fig. 4 Comparison between image convolution and graph convolution. **a** 2D convolution operation on an image. **b** Graph convolution operation, for the pink node, the features of the neighbouring nodes are aggregated through the grey edges to produce the updated representation in the next layer

in the spectral domain based on the foundation of graph signal processing, utilising the graph Fourier transform (Shuman et al. 2013). Whereas the spatial approaches perform convolution on neighbouring nodes in the graph domain. It should be noted that the differences between the aforementioned approaches are becoming indistinct as the GNN architecture evolves. Under highly specific conditions, certain spectral filter approximations are nearly equivalent to spatial methodologies in their application and outcomes (Georgousis et al. 2021).

2.3 GNN building blocks

Many popular GNN frameworks adopt mechanisms similar to that of CNNs that work on images, whereby aggregation of nodal information happens in the spatial domain. Similar to CNNs, a comprehensive GNN architecture are usually composed out of multiple layers each consisting of a building block with unique weights, the following subsections describe different variations of convolution operations within a single building block.

2.3.1 Message passing neural network (MPNN)

One of the most widely used frameworks is the message passing neural network (MPNN) proposed by Gilmer et al. (2017). This framework contains two forward-passing phases: a message passing phase and a readout phase. The message passing phase aggregates node features along edges with a message function $M_l(\cdot)$ and the update function $U_l(\cdot)$ both with learnable parameters typically based on MLP neural networks for each edge for message passing. This spatial graph convolution operation can be denoted as:

$$\mathbf{h}_v^{(l+1)} = U_l\left(\mathbf{h}_v^{(l)}, \sum_{u \in N(v)} M_l\left(\mathbf{h}_v^{(l)}, \mathbf{h}_u^{(l)}, \mathbf{x}_{vu}^e\right)\right) \tag{1}$$

where $\mathbf{x}_{vu}^e$ is the feature vector for edge between the two nodes v and u. The updated node hidden feature vector can either be directly passed to an output layer to perform node-level prediction or to a readout function $R(\cdot)$ based on MLPs to perform graph-level prediction:

$$\mathbf{x}_G = R\left(\mathbf{h}_v^{(L)} | v \in G\right) \tag{2}$$

where L denotes the number of total layers. The message passing process is a general form of many existing graph convolution operations. The MPNN serves as a comprehensive

framework incorporating a variety of GNN building blocks, with differently defined message functions, update functions and readout functions. These frameworks have been used in various applications in the field of engineering (Gao et al. 2024; Mozaffar et al. 2021; Pfaff et al. 2020; Prachaseree and Lejeune 2022; Sanchez-Gonzalez et al. 2020).

2.3.2 Graph network (GN)

Battaglia et al. (2018) proposed the graph network (GN), one of the most general frameworks that unifies a number of existing convolution approaches. The core component of a GN is the GN block which aggregates and updates information at the node, edge and graph level. A GN block defines three update functions $\phi(\cdot)$, and three aggregation functions $\rho(\cdot)$ on the node, edge and graph levels:

$$\begin{aligned}
\mathbf{h}_{e_k}^{(l+1)} &= \phi^e\left(\mathbf{h}_{e_k}^{(l)}, \mathbf{h}_{v_{r_k}}^{(l)}, \mathbf{h}_{v_{s_k}}^{(l)}, \mathbf{u}\right), \bar{\mathbf{h}}_{e_i}^{(l+1)} = \rho^{e\to v}\left(\mathbf{H}_{E_i}^{(l+1)}\right), \\
\mathbf{h}_{v_i}^{(l+1)} &= \phi^v\left(\bar{\mathbf{h}}_{e_i}^{(l+1)}, \mathbf{h}_{v_i}^{(l)}, \mathbf{u}\right), \quad \bar{\mathbf{h}}_{e}^{(l+1)} = \rho^{e\to u}\left(\mathbf{H}_{E}^{(l+1)}\right), \\
\mathbf{u}^{(l+1)} &= \phi^u\left(\bar{\mathbf{h}}_{e}^{(l+1)}, \bar{\mathbf{h}}_{v}^{(l+1)}, \mathbf{u}\right), \bar{\mathbf{h}}_{v}^{(l+1)} = \rho^{v\to u}\left(\mathbf{H}_{V}^{(l+1)}\right).
\end{aligned} \tag{3}$$

The update functions ϕ^e, ϕ^v and ϕ^u update the node, edge and graph features from one layer (l) to the next ($l+1$); a local aggregation function $\rho^{e\to v}$ aggregates the updated edge features to each node; two global aggregation functions $\rho^{e\to u}$ and $\rho^{v\to u}$ aggregate the updated edge and node features at the graph level. In general, the aggregation functions should be invariant to permutations of nodes and edges.

Diverse combinations of the activation and aggregation functions define many variations of the GNN building blocks. The MPNN can be seen as a variant of the GN without global processing functions.

In practice, most applications of GN are in the form of encoder–processer–decoder architecture consisting of multiple GN blocks. The encoder part transforms the features into a latent space, which are then subject to a sequence of processor GN blocks. The decoder GN blocks are similar to the encoder part but operate inversely to convert the latent vectors back into the ultimate form of output data.

2.3.3 Graph convolutional network (GCN)

As mentioned before, graph convolution can be operated in both the spatial and spectral domains. The spectral approaches process graph signals in the spectral domain by making use of the graph Fourier transform, specifically, by computing the eigen-decomposition of the graph Laplacian. A typical example of the spectral approaches is the Chebyshev spectral CNN (ChebNet) (Defferrard et al. 2016). ChebNet approximated the spectral graph kernel with the K-localised Chebyshev polynomial of the eigenvalue matrix of the Laplacian matrix. Spectral graph convolution computation can be expensive due to the eigen-decomposition of the Laplacian matrix, especially for large graphs.

Motivated by the ChebNet, Kipf and Welling (2016) proposed the graph convolutional network (GCN) by employing the first-order approximation of the Chebyshev polynomial. The propagation module can be described in matrix form as:

$$\mathbf{H}_V^{(l+1)} = f\left(\tilde{\mathbf{D}}_G^{-\left(\frac{1}{2}\right)}\tilde{\mathbf{A}}_G\tilde{\mathbf{D}}_G^{-\left(\frac{1}{2}\right)}\mathbf{H}_V^{(l)}\boldsymbol{\Theta}\right) \tag{4}$$

where l is the layer number, $f(\cdot)$ is the activation function, $\tilde{\mathbf{A}}_G = \mathbf{A}_G + \mathbf{I}_N$, and $\tilde{\mathbf{D}}_G = \text{diag}\left(\sum_{k=1}^{N}\tilde{\mathbf{A}}[1,k],\ldots,\tilde{\mathbf{A}}[N,k]\right)$, $\boldsymbol{\Theta}$ is a matrix filled with learnable parameters. Although this localised feature aggregation approach is derived from the graph Laplacian's spectral decomposition, the simplification avoids the need for full eigen-decomposition. The GCN framework applies convolutions by aggregating features from the immediate neighbours of each node. This aggregation is a localised operation, which is a characteristic of spatial methods. Therefore, in practice, GCNs can be seen as a hybrid leveraging the theoretical foundations of spectral approaches while implementing convolution in a spatial-like approach. Due to this characteristic, GCNs can be seen as specific cases of the MPNN and GN which operate without considering the edge features.

2.3.4 Graph attention network (GAT)

While the aforementioned frameworks assume identical contributions of neighbouring nodes, several approaches have been proposed to perform spatial graph convolution based on different weighting strategies. For example, the graph attention network (GAT) (Veličković et al. 2017) adopts an attention mechanism. The GAT utilises a self-attention mechanism which computes the weights of each node spatially beside the target node, and the attentional convolution operation can be denoted as:

$$\mathbf{h}_v^{(l+1)} = \sigma\left(\sum_{u\in N(v)} a_{vu}\mathbf{W}^{(l+1)}\mathbf{h}_u^{(l)}\right) \tag{5}$$

where σ is the sigmoid activation function, the attention weight a_{vu} is implicitly computed by neural networks to determine the importance of each neighbour to the node, $\mathbf{W}^{(l+1)}$ is the weight matrix of the $l+1$ layer.

2.3.5 GraphSAGE

A number of GNN frameworks encounter a challenge in the form of heterogeneous degrees of node connectivity, leading to a non-uniform distribution of neighbourhood sizes. This gives rise to computational inefficiencies when attempting to incorporate the entire node's neighbourhood during graph convolution operations on large graphs. This is because storing and processing the connectivity information of nodes with a high degree of connections requires substantial memory resources. In addition, the complexity of aggregating features from differently sized neighbourhoods slows down processing and influences computational efficiency, especially when normalising contributions from neighbours. Specially tailored aggregation functions need to be defined for different types of graph data in terms of graph sizes and connection types.

Hamilton et al. (2017) introduced an inductive representation learning technique on large graphs, namely the GraphSAGE (Sample and Aggregate). GraphSAGE implements a

sampling method to obtain a uniform number of neighbouring nodes for every node in the graph. The graph convolution operation can be defined by:

$$\mathbf{h}_v^{(l+1)} = \sigma\left(\mathbf{W}^{(l+1)} g_{l+1}\left(\mathbf{h}_v^{(l)}, \left\{\mathbf{h}_u^{(l)} \forall u \in S_{\mathcal{N}(v)}\right\}\right)\right) \tag{6}$$

where $S_{\mathcal{N}(v)}$ is a fixed sized sample of the neighbourhood of node v. The aggregation function g can be either mean, sum or max aggregator.

All the aforementioned GNN frameworks have been wide adopted in various mechanics-related applications. Based on the characteristics of each scenario, appropriate frameworks are selected to optimally address the specific challenges.

3 GNNs applied to various mechanics-related domains

This section summarises the GNN applications in mechanics-related domains based on existing papers from peer-reviewed journals, conferences, and preprints. According to Sect. 1.1, the mechanics-related applications of interest in this review paper are categorised into solid mechanics, fluid mechanics, and interdisciplinary mechanics-related domains. Such a categorisation scheme is based on unique challenges and commonalities of each domain, necessitating unique GNN models to address these challenges, thereby ensuring clarity in analysis and ease of reference for domain-focused readers. We particularly emphasise the utilisation of GNNs in solid mechanics applications, as most of their applications are reliant on Lagrangian representations which GNNs excel in.

3.1 Application of GNNs in the solid mechanics domain

This section presents a comprehensive review of GNN applications in solid mechanics domain with a detailed summary illustrated in Table 1. These applications were summarised into three main subdomains, including continuum and frame-based structures, mechanical metamaterials, fracture mechanics and tool wear predictions. For each subdomain, we reviewed the graph representation methods, GNN models and discuss the challenges and future works of the GNN applications.

3.1.1 Continuum and frame-based structures

The continuum and frame-based structures subdomain emphasises the study of macroscopic structures such as beams, columns, plates, and frames, and how these structures respond to external forces and moments. Typical GNN applications in this subdomain include predicting the deformation fields (Black and Najafi 2022; Deshpande et al. 2024; Gao et al. 2022; He et al. 2023; Pfaff et al. 2020), stress fields (Fu et al. 2023), and the structural dominant failure mode of beam or frame structures under various loading conditions such as point loads or contact between objects (Tian et al. 2024).

Graph representation methods

The graph representation methods can be categorised into continuum and frame-based structures, as illustrated in Fig. 5. The graph construction methods for continuum structures are based on FE meshes. The FE mesh nodes represent graph nodes, and the FE mesh edges represent graph edges. Input node features include external nodal forces (Deshpande et al. 2024; Fu et al. 2023; Tian et al. 2024), nodal coordinates (Fu et al. 2023; He et al.

Table 1 GNN application summary in the solid mechanics domain

Applications	Descriptions	GNN frameworks	References
Continuum and frame-based structures	Displacement field prediction; 2 Dimensional (2D) elastostatic problem of cantilever beam	Multi-fidelity MPNN with subgraph analysis and physics-informed training	Black and Najafi (2022)
	Displacement field prediction; 2D or 3 Dimensional (3D) linear elastic hollow cylinder	GCN with physics-informed loss functions	Gao et al. (2022)
	Displacement field prediction; 3D clothes deformation in contact with a rigid ball	Multigraph MPNN with learned adaptive remeshing	Pfaff et al. (2020)
	Displacement field prediction; 3D hyper-elastic plate	Spatial–temporal multigraph MPNN	Pfaff et al. (2020)
	Displacement field prediction; 3D elastic plate subject to an actuator	Multiscale MPNN	Cao et al. (2023)
	Displacement field prediction; 2D/3D beam	GCN model with input-independent learnable weighted multi-channel aggregation	Deshpande et al. (2024)
	Displacement field prediction; 3D linear elastic and hyper-elastic beam	GCN with physics-informed training	He et al. (2023)
	Stress field prediction; 2D cantilever beam	GCN with attention modules	Fu et al. (2023)
	Structural dominant failure modes classification; 2D truss structure	MPNN with attention modules	Tian et al. (2024)
	Dynamic responses prediction; 3D elastic beam and 2D elastic frame	GN combined with RNN with physics-informed training	Chen et al. (2024)
	Stiffness prediction; 3D post milled aircraft structural parts	GCN with graph level features	Chen et al. (2022)
	Buckling direction prediction; 2D asymmetric buckling columns	GCN	Prachaseree and Lejeune (2022)
	Structural design; 2D beam layout design	MPNN with GraphSAGE	Zhao et al. (2023)
	Structural design; 3D geometry feature designs	MPNN	Wong et al. (2022)

Table 1 (continued)

Applications	Descriptions	GNN frameworks	References
Mechanical metamaterials	Wave propagation and displacement prediction; 2D shell metamaterial	MPNN with physics-based edge update function	Xue et al. (2023)
	Dominant deformation mechanism classification; 3D truss metamaterial	MPNN classification model	Indurkar et al. (2022)
	Deflection prediction; 3D truss metamaterial	GCN	Ross and Hambleton (2021)
	Stiffness prediction; 3D shell metamaterial	MPNN	Meyer et al. (2022)
	Metamaterial design; 2D architected materials	Semi-supervised GCN	Guo and Buehler (2020)
	Generative modeling and inverse design; 3D truss metamaterial	Graph based VAE combined with a property predictor	Zheng et al. (2023)
	Metamaterial inverse design; 2D truss metamaterial	GCN (edgeconv) and MPNN	Dold and Aranguren van Egmond (2023)
Fracture mechanics and tool wear predictions	Fracture mechanics crack propagation prediction; 2D brittle materials	Spatial–temporal MPNN	Perera et al. (2022)
	Crack field and displacement field prediction	Multi-scale MPNN	Perera and Agrawal (2024)
	Tool wear field prediction; 3D york metal forging die	GCN	Shivaditya et al. (2022)

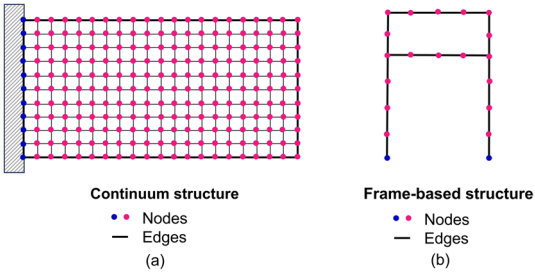

Fig. 5 Common graph representation methods for **a** Continuum and **b** frame-based structures

2023; Pfaff et al. 2020; Tian et al. 2024), material properties (Fu et al. 2023), and node types for heterogeneous graphs (Pfaff et al. 2020). Input edge features include the distance between each node (Black and Najafi 2022; Pfaff et al. 2020; Tian et al. 2024), and edge types represent different boundary conditions (Fu et al. 2023). Additionally, it is worth mentioning that node positional coordinates can differ under different global coordinate systems. Therefore, the GNN model trained with data from one global coordinate system might not be applicable to other coordinate systems. To apply a coordinate-free GNN model, a recent study from Chen et al. (2024) used the relative nodal displacements in node attributes and the change of edge length as edge features. Additionally, they also included gravitational acceleration and timestep interval as graph level features (Chen et al. 2024). While for the frame-based structure, graph nodes could represent joints or connection points in the frame, graph edges could represent the discrete elements connecting these graph nodes as shown in Fig. 5b. Geometric positions and constraints could be modelled as graph node features while material properties, length of the edge could be embedded in graph edge features.

Zhao et al. (2023) extensively studied the graph representation of frame-based structures based on different node and edge types. Three different graph representation methods named Case-Normal, Case-Inverse, and Case-AllNodes were adopted, as shown in Fig. 6. The Case-Normal method represented columns as graph nodes and represented both beam and non-beam links as graph edges. The Case-Inverse method represented columns as graph edges and represents beam and non-beam links as graph nodes. The Case-AllNodes method described the columns, beam links, and non-beam links all as graph nodes. They discovered that the Case-Inverse method achieved the best overall accuracy compared to other graph representation methods.

However, in FE simulations, stress and strain states are computed on mesh elements rather than mesh nodes. Some case studies with the primary aim of predicting the stress and strain fields in the mesh elements adopted a novel graph representation method by encoding information of the abovementioned node and edge features in a condensed element level node. Fu et al. (2023) proposed a novel graph embedding approach named

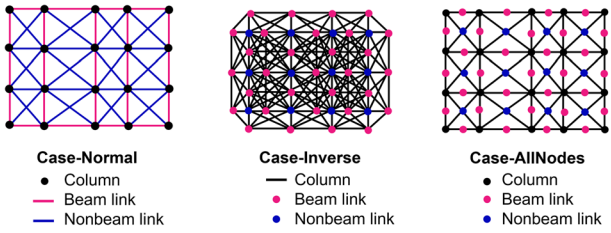

Fig. 6 Various graph representation methods explored for frame structure (Adapted from Zhao et al. (2023))

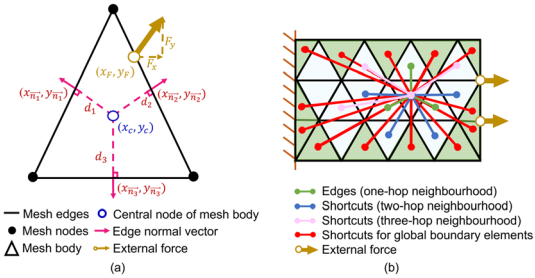

Fig. 7 Graph representation methods proposed by Fu et al. (2023). **a** Condensed graph representation features for one triangular mesh element. **b** An illustration for shortcut between the target mesh element with multi-hop neighbours and boundary elements (Adapted from Fu et al. (2023))

Boundary-oriented Graph Embedding (BOGE). In BOGE, the information of mesh nodes, edges, and elements was combined to form a condensed graph representation, as illustrated in Fig. 7. In the condensed graph representation, the graph nodes were the central nodes of mesh elements, and graph edges represented the connectivity between neighbouring mesh elements. Notably, instead of one-hop neighbouring edges, shortcuts were established between the target mesh element with both the global boundary mesh elements and local multi-hop mesh elements, as illustrated in Fig. 7. Therefore, fewer message-passing steps were required to accurately predict boundary value problems with long-range graph-vertex interactions. Furthermore, the condensed graph representation method significantly enhances flexibility for learning different loading conditions. Specifically, the external loading in the training dataset can be applied not only to the mesh nodes or edges but also to any arbitrary point within the mesh body. The location of the loading can be defined as the distance relative to the centre of the element mesh. This capability allows the model to learn various loading conditions, thereby substantially improving its generalisability across diverse scenarios.

Additionally, external loadings may not always be explicitly defined in FE simulations for direct inclusion in graph inputs. Instead, they can be implicitly represented based on the contact interactions between multiple objects. Pfaff et al. (2020) proposed a multigraph MPNN architecture named MeshGraphNet (MGN) that can implicitly learn mesh-based self-collisions or contact between two objects. The graph structure in MGN had the addition of world graph edges that were generated based on a predefined hyperparameter of connectivity radius, as displayed in Fig. 8. Therefore, the graph structure in MGN was referred to as a multigraph. The multigraph represented node features as nodal coordinates in the material mesh and global world spaces. It also represented edge features as nodal coordinate difference and Euclidean distance in both the material mesh space and 3D Cartesian world space (Pfaff et al. 2020). The mesh space focused more on message passing in the material space. In contrast, the world space analysis focused more on message passing

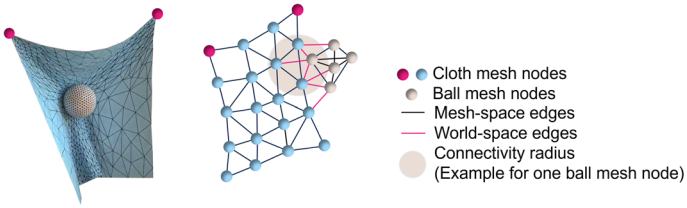

Fig. 8 Graph representation method for modelling the contact between objects (Adapted from Pfaff et al. (2020))

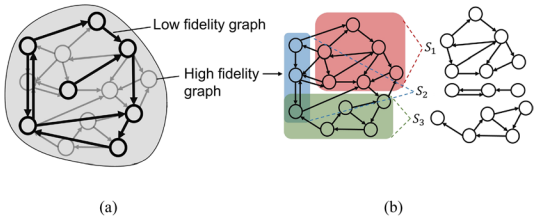

Fig. 9 The multi-fidelity graph representation method and subgraph analysis proposed by Black and Najafi (2022). **a** Illustration of low and high-fidelity graph. **b** The high-fidelity graph is separated into three subgraphs (S1, S2, S3) and analysed separately (Adapted from Black and Najafi (2022))

in the world space involving contact between objects and self-collisions. Additionally, they trained a dynamic and a sizing field model to perform learned dynamic re-meshing during rollout (Pfaff et al. 2020). All learned re-meshing variants could adeptly adjust the mesh resolution during each test time step and were, therefore, able to provide improved performance to the GNN model without the need to include a domain-specific re-mesher in the GNN model.

Moreover, the abovementioned graph construction methods often concern high-fidelity analysis with high-resolution mesh structures. A high-fidelity mesh may contain tens of thousands of mesh edges and mesh nodes with multiple channels of edge or node features. Thus, storing the nodes and edges features and training the GNN models can be highly time-consuming. Therefore, an analysis using a Multi-Fidelity GNN (MFGNN) is illustrated in Fig. 9 (Black and Najafi 2022). The MFGNN model took the inputs of FE simulation results of a low-fidelity mesh and outputs high-fidelity predictions. The input of the MFGNN model included a low-fidelity graph with simulation results assigned as node and edge features. A high-fidelity graph with unassigned node and edge features is also included in the graph input. The node and edge features in the low-fidelity graph were projected to the high-fidelity node and edge features. The high-fidelity graph was then subdivided into several subgraphs to update nodes and edge features in these subgraphs using message passing functions. Finally, the output of the high-fidelity graph was retained from these subgraphs via a feature recovery process. After training, Black and Najafi (2022) discovered that the MFGNN with subgraph analysis could accurately predict the displacement field near boundary conditions. This may be because the subgraph analysis function in the MFGNN model enabled more localised analysis within each subgraph to capture more specific physical relations near the boundary conditions (Black and Najafi 2022).

GNN algorithms

According to Table 1, MPNN with edge-based feature processing is popular (Black and Najafi 2022; Pfaff et al. 2020; Tian et al. 2024). GCN models are also widely adopted (Deshpande et al. 2024; Gao et al. 2022; He et al. 2023; Zhao et al. 2023). Some continuum and frame-based structures related GNN applications have extended from these fundamental architectures of MPNN and GCN, as described in Sect. 2.3, based on specific problems they aim to address.

To further enhance the GNN message passing capabilities in learning long-range dependencies of geometric features and boundary conditions, a multi-scale graph learning approach was developed by Deshpande et al. (2024) named Multi-channel Aggregation Network (MAgNET). The MAgNET framework is an encoder-decoder architecture inspired by the CNN U-Net (Ronneberger et al. 2015). During encoding, one or more Multichannel Aggregation layers (MAg) were applied to the mesh-based dataset, followed

by a pooling layer to downsample the graph. The aggregation and pooling processes were repeated until a desired coarse mesh level was obtained. In the decoding stage, unpooling operations on the graph data were performed and followed by one or more MAg layers. Compared to CNN U-Net benchmark results, the graph-based method was claimed to have improved scalability by operating on unstructured data (Deshpande et al. 2024).

GNN models are mostly data-driven, meaning the model is trained on large datasets with common loss functions such as Mean Squared Error (MSE) or Root Mean Squared Error (RMSE), which compare the predicted output of GNN with the ground truth value from numerical simulations or experiments. Additionally, certain mechanical problems can be well described using fundamental physical principles. For instance, in the field of solid mechanics, the principles of equilibrium, compatibility, and constitutive laws are essential for defining the kinematics of continuum bodies. These physical laws also offer crucial temporal and spatial information that can be effectively incorporated into the training process. Consequently, by integrating the concept of physics-informed neural networks (PINNs) with GNN architectures, it becomes possible to address forward or inverse problems governed by physical laws, thus reducing the reliance on extensive datasets. For instance, He et al. (2023) combined GNN with the deep energy method (DEM) to model the linear elastic and hyperelastic material behaviours in a cantilever beam. DEM is a type of PINN that solves physical problems by minimising the total energy (Nguyen-Thanh et al. 2020). In the combined GNN-DEM method developed by He et al. (2023), the displacement fields were predicted first, which then helped compute the energy functional used as the loss function, minimised using standard optimisation techniques. Additionally, Gao et al. (2022) utilised a GCN model with Galerkin variational formulation to address both forward and inverse Poisson equation problems and linear elastic problems. The Galerkin variational formulation is crucial as it effectively incorporates the weak form of the PDE into the computational model. In the GCN framework, this weak form is adeptly integrated by representing the domain as a graph, where each node corresponds to a spatial point. This representation allows the network to focus on minimising the residuals of these integral equations, enhancing the network's ability to accurately resolve the underlying physical problems.

Instead of the abovementioned attempts that utilised standard PINN methods by adding physical laws to train the GNN model, several architectures have tried to embed additional physical terms in the loss function to guide the GNN model to implicitly learn some physical constraints via a data-driven approach. For instance, an internal potential energy-based objective term was adopted in the training process to model the elastostatic material behaviour in a cantilever beam (Black and Najafi 2022). In addition to comparing the predicted and ground truth displacement fields, the model also introduced a loss function term that compared the predicted and ground truth values of the internal potential energy. Additionally, Chen et al. (2024) composed a loss function, based on the predicted nodal acceleration and internal forces comparing with the simulation ground truth values, as these two criteria were essential for computing the second-order differential equations governed by Newton's second law.

Additionally, several GNN models are combined with RNN models in order to capture the spatial–temporal relationships in the mechanics problem. Chen et al. (2024) incorporated the rest states such as the internal forces of elements in a continuum deformable body as the initial hidden states of the gated recurrent unit (GRU) when developing a combined RNN and GNN surrogate model. It was discovered that the RNN architecture helped to reduce the accumulated long-term rollout error since it was trained to reduce the loss of multi-step sequence prediction during model training.

Attention-based GNN methods are also incorporated to facilitate a more nuanced understanding of graph structure and message passing processes. Tian et al. (2024) developed a hierarchical GAT with node and edge attention modules built inside the MPNN fundamental architecture. Compared to the genetic algorithm (GA) and the β-unzipping method, the proposed attention-based GNN algorithm exhibited higher computational efficiency.

Further discussions

Most GNN applications in this domain often exclude material properties from their input tensors and have minimal material variations in their training dataset. The absence of explicit material featurisation in the input tensor means that even though the model might implicitly learn the properties of specific materials used in the training set, its generalisation performance is limited when encountering new materials, thus constraining the capability of GNNs to learn and generalise across diverse material characteristics. This limitation likely stems from the challenges encountered in dataset preparation, particularly the difficulty of assembling datasets that represent a variety of material properties alongside complex and high-dimensional geometric parameterisation schemes and requires future investigations. Moreover, most GNN models focus on learning linear material behaviours such as linear elastic materials. Furthermore, the majority of GNN models are primarily designed to capture nonlinear dynamics under the assumption of linear material behaviours, such as that exhibited by elastic materials. However, their effectiveness in accurately modelling nonlinear material behaviours, such as those involving elastoplastic, remains limited and requires further investigation.

GNN performance is sensitive to the input variables' size and physical meanings (Fu et al. 2023). If very few input variables are embedded, the GNN model may tend to underfit due to a lack of effective physical embeddings. However, if the input variables size is too large, the GNN model may become highly computationally intensive and prone to overfitting. Therefore, future work is expected to investigate various input combinations of the GNN model to explore the potential of optimal input combinations.

Furthermore, the current models mainly focus on relatively small graphs with several thousands of nodes and edges involved. When modelling the dynamics of complex continuum structures, finer meshes are required to observe the localised features such as displacement or stress concentrations. This would result in significant increase of node and edge numbers in the graph embeddings. The training time may be considerable due to additional edges and nodes required to represent the system accurately. Although previous methods such as MFGNN is proposed to address this issue, the MFGNN is not an independent solver as it requires FE simulations to obtain low fidelity graph embeddings.

Moreover, when operating on large graphs, the message-passing layers may get deeper, resulting in gradient vanishing or gradient explosion (Lukovnikov et al. 2020). Therefore, novel GNN architectures addressing this issue could be adopted. For instance, Addanki et al. (2021) explored a GNN model for deep transudative node classification that leverages bootstrapping, and a very deep inductive graph regressor capable of handling up to 50 layers regularised by denoising techniques. Conversely, if message-passing layers are relatively shallow, the GNN model may lack the capacity to capture the long-range dependency in the system. For instance, the deformation of the cantilever beam next to the boundary wall may be influenced by the force applied to the cantilever beam at the other end. With shallow layers, the message propagation may not be able to capture this type of long-range dependencies. To mitigate this problem, multi-scale GNNs with down-sampling techniques can be explored in a hierarchical order by enhancing the message passing over graphs (Cao et al. 2023).

3.1.2 Mechanical metamaterials

Mechanical metamaterials (MMs) are micro or nano-scale materials designed with a hierarchical architecture (Surjadi et al. 2019). MMs exhibit unique mechanical properties such as high strength-to-density ratio and high resilience. Studies have extensively focused on predicting the deformation mechanisms (Indurkar et al. 2022; Xue et al. 2023) or other structure–property relations, such as elastic moduli and heat conductivity of MMs (Meyer et al. 2022). Additionally, it is worth noting that some inverse designs of MMs with targeted desired properties have emerged in recent years (Dold and Aranguren van Egmond 2023; Zheng et al. 2023).

Graph representation methods

Developing effective graph construction methods for MMs is essential to enhance input featurisation, which is crucial for the learning and performance of GNN models. For truss-based metamaterials, common graph construction methods include representing the nodes in the 3D metamaterial structure as graph nodes and the struts as graph edges, as shown in Fig. 10a. Node features often include nodal positions and load levels, while edge information is commonly edge length or strut orientations (Indurkar et al. 2022). Moreover, Ross and Hambleton (2021) encoded node types into the graph system to learn the boundary representation of the periodic unit cell structure.

Besides truss-based metamaterials, a shell-based metamaterial can be represented by a cross-spring system that contains rigid crosses connected by nonlinear springs, as illustrated by Fig. 10b. In the cross-spring system, the rigid crosses are represented as graph nodes, and the nonlinear springs are encoded as graph edges. The nodal features contain the position and orientation of the rigid elements, and the edge features contain elastic energy stored in each spring (Xue et al. 2023).

GNN algorithms

From Table 1, commonly applied GNN algorithms include MPNN and GCNs. In addition, several unique GNN architectures with tailored functions have been proposed (Guo and Buehler 2020; Xue et al. 2023). To improve prediction accuracy, Xue et al. (2023) incorporated a physics-based edge update function based on the functional form of elastic energy of the cross-spring system. This approach provided a quantitatively accurate solution to predict the dynamics of large-scale structures comprising over 200×200 metamaterial unit cells with hyper-elastic properties. Guo and Buehler (2020) adopted a semi-supervised learning approach to classify test node load levels to obtain the load levels of architecture metamaterials. The input graph contained load level data for only 1% of graph

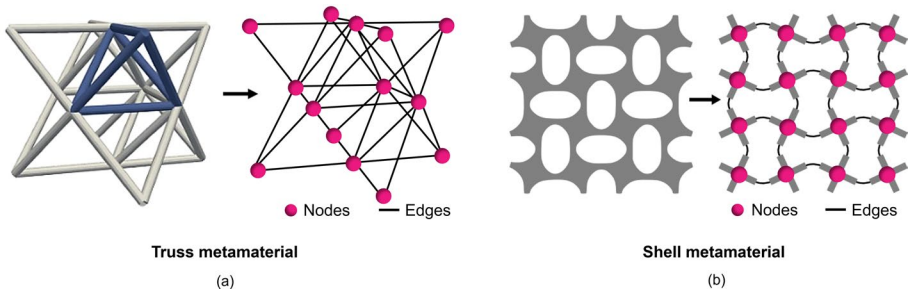

Fig. 10 Examples of graph representation methods for **a** truss metamaterial and **b** shell metamaterial (Adapted from Zheng et al. (2023) and Xue et al. (2023))

nodes, and the trained model was able to accurately predict the load levels for the remaining 99% of graph nodes. Notably, this model incorporated a predefined design approach that allowed for the creation of voids in low-load areas while maintaining the overall volume of the material. This approach ensured that in predicted high-load areas, the edge stiffness would increase, approaching a maximum stiffness value. Conversely, in predicted low-load areas, the edge stiffness would decrease, potentially reaching zero.

Most GNN applications in MMs mainly focus on forward prediction tasks such as predicting the structural–property relations. Regarding inverse design, the GNN architecture is often coupled with design algorithms using gradient descent. Moreover, some generative models were also proposed to generate various MMs with different geometric features. Zheng et al. (2023) developed a graph variational autoencoder-based generative model that enables continuous latent space exploration encompassing a vast array of truss metamaterials. The generative model also enabled the convenient generation of new designs via latent space traversal or structure interpolation. Based on the generative model, they further developed an optimisation framework for inverse truss design, targeting custom mechanical properties, including exceptionally stiff, auxetic, pentamode-like, and tailored nonlinear behaviours (Zheng et al. 2023). Moreover, Dold and Aranguren van Egmond (2023) developed an approach that utilises differentiable message-passing algorithms with a unique type of convolution layer named the EdgeConv layer. The EdgeConv layer updated node features by considering the differences between neighbouring nodes and applying linear transformations followed by max-pooling. Additionally, they integrated a GRU to iteratively update node features. They compared the GNN model to an MPNN baseline and gradient descent optimisation method to permit modifications to the geometric configurations and local attributes of individual lattice elements. They discovered that the EdgeConv surrogate model had enhanced material property prediction accuracy compared to the MPNN model.

Further discussions

In practical applications in mechanical metamaterials, visco-hyperelastic materials are often used (Xue et al. 2023). While Xue et al. (2023) investigated the dynamic response of a beam structure composed of shell-based metamaterials with a simplified hyperplastic behaviour, they did not account for viscosity and energy dissipation. To capture the rate-dependant behaviour due to viscosity, RNNs can be utilised to capture temporal variations or MGN networks can be employed to learn material or dynamic constitutive equations, facilitating accurate predictions of system dynamics (Xue et al. 2023). Additionally, when investigating the dynamic response of complex structures consisting of a large number of unit cell metamaterials, issues may arise due to the increasing number of unit cells and their long-range dependencies. Long-range dependency in metamaterials refers to the influence that the behaviour or state of one unit cell can have on distant unit cells, beyond the immediate vicinity. To mitigate this, advanced neural network architectures, such as hierarchical GNN models or networks with established shortcuts between boundary conditions and local unit cells, could be explored to enhance the model capability to learn dynamic responses on a global scale rather than local scale. However, how to effectively down-sample the model without losing information on the unit cell representation of the metamaterial is also a challenging topic to discover.

Finally, most applications focus on the forward prediction of metamaterial properties given a predefined structure, while very few explore the inverse design of metamaterials. The metamaterial structure is determined by various design parameters that can vary independently. Consequently, the solution to an optimal design is not unique, often due to the presence of local optima in gradient-based optimisation methods. Addressing

this non-uniqueness is critical for the success of the inverse design problem, requiring the development of robust optimisation strategies to effectively navigate the complex design space.

3.1.3 Fracture mechanics and tool wear predictions

Fracture mechanics and tool wear predictions involve studying the propagation of defects in materials, such as predicting tool wear conditions of manufacturing tooling, which is essential in solid mechanics applications to predict material or structural failures. Although there has not been much research conducted in this domain, the current GNN applications in this subdomain include modelling the crack propagation in brittle materials (Perera and Agrawal 2024; Perera et al. 2022), and tool wear field prediction of manufacturing tools (Shivaditya et al. 2022).

Graph representation methods

Regarding GNN applications in crack propagation prediction, crack tips could be represented by graph nodes, and graph edges could represent the connectivity between multiple crack tips within a zone of influence. Node features may include cartesian coordinate positions and orientations of the crack-tip (Perera et al. 2022). Meanwhile, for tool wear field prediction, the FE mesh of the manufacturing tooling could be constructed into graphs with mesh nodes and mesh edges representing graph nodes and graph edges, respectively. The node features may include temperature and friction coefficient, while the edge information may include the connectivity between node pairs (Shivaditya et al. 2022).

GNN algorithms

When modelling crack propagation, spatial–temporal MPNN could be adopted to predict crack propagation of multiple time steps. Due to the intricacies of microcrack mechanics, Perera et al. (2022) proposed that the MicrocrackGNN framework consisted of four sub-GNN surrogate models, each predicting an underlying physical component of microcrack mechanics. The first two GNN sub-models predicted Mode-I and Mode-II stress intensity factors, the third GNN sub-model predicted crack propagation type, and the final GNN sub-model predicted the position of microcrack tips. The trained MicrocrackGNN demonstrated its capability to simulate crack propagation across various initial microcrack configurations, ranging from 5 to 19 microcracks. Notably, the prediction speed of the MicrocrackGNN was 25 times faster than traditional FE simulations (Perera et al. 2022). Moreover, compared to CNN-RNN baseline models Recurrent Convolutional Neural Network (RCNN) and Recurrent Encoder–Decoder Neural Network (REDNN), the sub-GNN surrogate models exhibited an order of magnitude smaller error relative to the baseline. Specifically, the RCNN model, which includes convolutional layers, batch normalisation layers, ReLU activation functions, and a linear output layer, is designed to handle sequential data. In contrast, the REDNN model employed an encoder–decoder architecture with convolutional and transpose convolutional layers, allowing it to reconstruct input data for enhanced feature learning. The results showed that the MicrocrackGNN outperformed both RCNN and REDNN models, highlighting its superior accuracy and efficiency in predicting crack propagation. However, they discovered that conventional mesh-based GNNs often face oversmoothing issues, especially when dealing with large graphs with high-fidelity meshes. The oversmoothing issue was due to insufficient message passing capabilities. To address this, Perera and Agrawal (2024) further introduced a multiscale GNN framework with adaptive mesh refinement method. This framework reduced the number of required message-passing steps by sequentially coarsening and upscaling the mesh. The

trained framework demonstrated faster simulation while maintaining high accuracy across all cases compared to traditional physics-based phase-field fracture models (Perera and Agrawal 2024).

Regarding tool wear predictions, Shivaditya et al. (2022) introduced a GCN architecture to predict the wear field of metal forging tools. In the GCN surrogate model, the first GCN layer takes a graph with 5 node features and outputs a graph with 50 hidden node features. The second GCN layer processed the graph into 100 hidden node features, followed by a ReLU activation layer. The subsequent three GCN layers reduce the hidden node features from 100 to 50 and then to 1, targeting the wear feature. Dropout layers were applied for regularisation after each activation function. The final ReLU layer ensures that all model outputs are positive. They discovered that the GCN surrogate model was at least four times more accurate compared to other benchmark models, such as Point-Net (Qi et al. 2017) and the Dynamic Graph Convolutional Neural Network (DGCNN) model (Wang et al. 2019).

Further discussions

The development of GNN applications in fracture mechanics is still in its nascent stage. The applications primarily focused on relatively fewer number of cracks. It is suggested that future work could extend from the MicrocrackGNN architecture mentioned in (Perera et al. 2022) and study the prediction performance on larger numbers of initial microcracks. Additionally, when modelling spatial–temporal MPNN architectures, the error accumulation in long-term rollout may influence model prediction accuracy. Integrating RNN with MPNN may help reduce the error accumulation since RNNs are trained to reduce the loss of multi-step sequence prediction. Regarding tool wear field prediction, Shivaditya et al. (2022) mentioned that additional features, such as normal stress and flow stress, have yet to be included in the GNN model to reduce computational effort. More case studies to analyse different input features and their relations with wear field prediction results may be beneficial. Finally, it is advised that the generalisability performance of GNN models in the fracture mechanics and tool wear prediction subdomain could be evaluated, which is essential for industrial practices. Moreover, future work would be beneficial in expanding these GNN applications to damage mechanics such as predicting the dislocation in the crystal structure of a material.

3.2 Application of GNNs in the fluid mechanics domain

This section summarises various GNN applications within the fluid mechanics domain. In this section, the applications were categorised into hydrodynamics, aerodynamics, and complex fluid rheology subdomains. In the hydrodynamics subdomain, the fluid of focus is usually liquids such as water or oil. Therefore, it is widely applied to tasks such as hydraulic machinery, submarine design, and ocean simulations. Aerodynamics is often concerned with air or other types of gaseous fluids. Studying the principles of aerodynamics is crucial for the design of vehicles and for optimising sporting facilities. Complexity fluid rheology often involves complex fluids that do not follow Newton's law of viscosity. This fluid type is known as non-Newtonian fluid, including polymers, suspensions, and biological fluids like blood. The complexity of fluid rheology is often seen in its applications across various industries, including cosmetics, food, and biomedicine. In line with the previously outlined categorisation methods, the graph representations, GNN algorithms, and challenges and future works in each fluid mechanics subdomain are briefly discussed in this section. A

Table 2 GNN application summary in fluid mechanics domain

Applications	Descriptions	GNN frameworks	References
Hydrodynamics	Eulerian method; Ocean, river hydrodynamics	GCN with multiscale graph representation	Shi et al. (2022)
		Multiscale spatial–temporal MPNN with adaptive remeshing	Lino et al. (2022b)
	Lagrangian method; Fluid fall and box bath	Spatial–temporal MPNN	Sanchez-Gonzalez et al. (2020)
		GCN	Li and Farimani (2022)
	Lagrangian method; 3D watercourse-inverse design	Spatial–temporal MPNN	Allen et al. (2022b)
Aerodynamics	Eulerian method; Incompressible laminar flow over cylinder	GCN with GAT	Liu et al. (2022)
		GCN with ResBlock module for node aggregation, physics-informed training, and attention mechanism	He et al. (2022)
		MPNN with multiscale and graph level representation	Yang et al. (2022)
		Spatial–temporal MPNN	Li et al. (2023c)
	Eulerian method; Incompressible laminar flow over aerofoil	GCN with GraphSAGE	Ogoke et al. (2021)
		Spatial–temporal MPNN	Gao et al. (2024)
		Multiscale spatial–temporal MPNN	Fortunato et al. (2022)
		GCN combined with CFD	Belbute-Peres et al. (2020)
		GCN	Jessica et al. (2023)
	Eulerian method; Incompressible laminar flow over arbitrary shapes	GCN	Chen et al. (2021)
		Rotational equivariant multiscale MPNN	Lino et al. (2022a)
	Eulerian method; Fluid field prediction and optimisation of turbomachinery	GCN	Li et al. (2023a)
	Eulerian method; Turbulence modelling of the wake in flow over a cylinder	MPNN with physics informed training	Quattromini et al. (2023)

Table 2 (continued)

Applications	Descriptions	GNN frameworks	References
	Eulerian method; 3D turbulent flow predictions of urban wind fields	Spatial–temporal MPNN with physics-informed training	Shao et al. (2023)
	Eulerian method; 2D turbulent channel flow	MPNN	Dupuy et al. (2023)
	Lagrangian method; 3D laminar Taylor–Green Vortex, and 3D reverse Poiseuille Flow	Rotation equivariant spatial–temporal MPNN	Toshev et al. (2023)
Rheology of complex fluid	Eulerian method; Blood, yogurt, or polymer solutions	Rheology-informed multi-fidelity GNN	Mahmoudabadbozchelou et al. (2022)

comprehensive summary of GNN applications in the fluid mechanics domain is given in Table 2.

3.2.1 Hydrodynamics

Applications of GNNs in the hydrodynamics subdomain have been concerned with large graph predictions of temperature fields and fluid flow of oceans or rivers (Lino et al. 2022b; Shi et al. 2022). Additionally, various rollout tests of fluid particles interacting with each other inside a constrained space, such as the dynamics of fluid particles in water cubes and bathtubs (Allen et al. 2022b; Li and Farimani 2022; Sanchez-Gonzalez et al. 2020), are popular in this subdomain.

Graph representation methods

We have further divided the GNN applications in fluid mechanics subdomains based on the Eulerian and Lagrangian methods to provide a more precise distinction and navigation to readers interested in either method. The Eulerian method uses a fixed grid system to analyse fluid motion (Harlow and Hirt 1972). Through finite-difference methods, the mass, momentum, and energy conservation laws are applied to the fixed grids to calculate fluid behaviours. For instance, the 2D cylinder flow problem uses the Eulerian method because the mesh grid for the control domain is fixed. In contrast, the Lagrangian method studies fluid dynamics by tracking the movement of individual fluid particles over time (Wang 2023). It involves applying the mass, momentum, and energy conservation laws to a mesh of particles that move along with the fluid.

Because of the abovementioned distinct methods in analysing fluid behaviours, there are significant differences in graph construction techniques when applying GNN with Eulerian and Lagrangian methods. In the Eulerian method, people are commonly interested in the changes in physical properties over time at fixed spatial points. Therefore, the graph is usually static. In the static graph, each node is fixed in spatial location, and the interaction with its neighbourhood is constant (Shi et al. 2022). In the Lagrangian method, fluid particles were encoded as graph nodes and graph edges were added between particles within a connectivity radius (Lino et al. 2022b). The graph edges representing the current interactions between particles are also dynamic to accommodate changes in graph node interactions over time (Lino et al. 2022b; Sanchez-Gonzalez et al. 2020; Li and Farimani 2022). For instance, two fluid particles previously close to each other and interacting strongly may move relatively far away in the next time step, causing interactions between them to weaken or disappear.

Additionally, it is noticed that storing intermediate feature maps at full resolution can be challenging because of limited GPU memory. To solve this problem, Shi et al. (2022) used graph hierarchical tree cutting to build a graph hierarchy consisting of graphs at different resolution levels. Nevertheless, modelling hydrodynamics of oceans and rivers often requires effective message passing on large graphs with an excessive number of message-passing layers. Propagating the node and edge features between nodes separated by hundreds of hops would require an extensive number of message passing layers, which could be inefficient and unscalable (Lino et al. 2022b). Therefore, Lino et al. (2022b) proposed multiscale MPNN that created graphs with different levels. To illustrate, lower-resolution graph levels possess fewer nodes and edges and, therefore, require a smaller number of messages passing layers to propagate node and edge features over longer distances effectively.

GNN algorithms

Commonly applied GNN algorithms in the hydrodynamics domain are spatial–temporal MPNNs to model the evolution of fluid properties such as velocity and pressure field at incremental timesteps, mimicking the physics simulator (Sanchez-Gonzalez et al. 2020). A few GCN models also emerged (Li and Farimani 2022; Shi et al. 2022). To accurately predict the fluid motion with a far less parameterised model, Li and Farimani (2022) decomposed a single forward step in the fluid dynamics into multiple simpler sub-systems, namely node-focused network and edge-focused network. The node-focused network is a GCN with distance-weighted aggregation methods, and the edge-focused network could be regarded as a GCN with only edge information.

Further discussions

Overall, the GNN applications in hydrodynamics have showcased their improved training efficiency without compromising the accuracy of prediction results. Additionally, improved generalisability performance was discovered in the MPNN network (Sanchez-Gonzalez et al. 2020). However, the current spatial–temporal MPNN model is trained on next-step prediction and unrolled with a fixed time step (Sanchez-Gonzalez et al. 2020). Future work could potentially analyse adaptive time stepping, investigate sequence models and the role of training noise in error accumulations, or incorporate an RNN-based model for reducing error accumulations (Lino et al. 2022b).

3.2.2 Aerodynamics

From Table 2, GNN applications in the aerodynamics subdomain mainly involve predicting 2D aerofoil flow, cylinder flow, or flow over arbitrary shapes. The prediction outputs are primarily fluid field predictions such as velocity field, pressure field, and wall shear stress. Moreover, there are several GNN applications in the 3D domain, such as the 3D flow field in turbomachinery (Li et al. 2023a), turbulence modelling of wake region over a cylinder (Quattromini et al. 2023), turbulent flow predictions of urban wind fields (Shao et al. 2023), turbulent channel flow (Dupuy et al. 2023), as well as 3D laminar Taylor–Green Vortex and reverse Poiseuille Flow (Toshev et al. 2023).

Graph representation methods

Most GNN applications of aerodynamics problems adopted the Eulerian method, especially when studying laminar fluid dynamics. Meshes in computational fluid dynamics (CFD) can be naturally interpreted as graphs, where mesh nodes are represented as graph nodes and mesh edges are represented as graph edges. Node input features often include nodal coordinates, while edge features include node connectivity information and the relative distance between connecting node pairs (Belbute-Peres et al. 2020; Ogoke et al. 2021; Pfaff et al. 2020; Shao et al. 2023). Moreover, Toshev et al. (2023) included the external force field of the reverse Poiseuille flow as a node feature when studying 3D turbulent flow.

In addition to the abovementioned graph representation methods, a graph augmented representation method based on node-element hypergraph is proposed by Gao et al. (2024), as illustrated in Fig. 11. In the data representation stage, the node-element hypergraph encoded mesh nodes as graph nodes. The graph edges were constructed based on element-node edges, relative locations between element centres and nodes were encoded in the edge feature. The proposed graph representation method produced more stable and accurate results compared to the conventional graph construction method for a longer rollout time step within the training range (Gao et al. 2024).

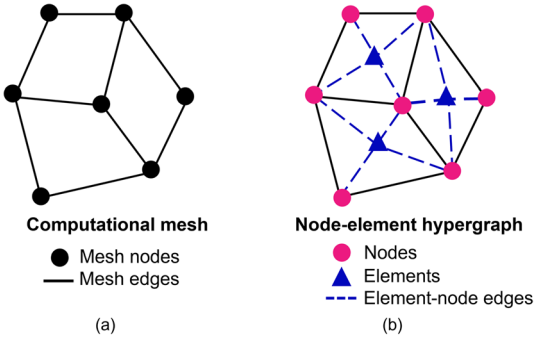

Fig. 11 Comparison of **a** computational mesh and **b** node-element hypergraph (Adapted from Gao et al. (2024))

Moreover, Lino et al. (2022a) applied a rotation-invariant representation of the input and output vector fields with directed angles between adjacent edges. By detailing the angles between edges and incorporating the lengths of these edges, this approach offers a comprehensive representation of the relative positions of nodes without explicitly requiring a coordinate system. This rotational equivariant data enabled higher prediction accuracy than the explicit representation method based on coordinate systems, which directly uses the coordinates of the nodes to define their positions.

GNN algorithms

MPNN and GCN have been applied to fluid dynamics problems, as in Table 2. Spatial–temporal MPNN models are commonly adopted (Gao et al. 2024; Shao et al. 2023; Toshev et al. 2023) to perform rollout tests that predict intermediate time steps rather than end-to-end predictions. Spatial–temporal MPNN models allow the GNN model to capture the complexity of flow evolutions such as turbulence and vortex formation. Additionally, several examples have studied multiscale MPNNs to improve message-passing capabilities (Fortunato et al. 2022; Lino et al. 2022a; Yang et al. 2022). Fortunato et al. (2022) proposed the Multiscale MGN as a hierarchical update of the MGN. The multiscale MGN model transferred 'high accuracy' labels from a high resolution simulation onto a coarser, lower resolution mesh, achieving more accurate and less time-consuming prediction results than the MGN using the same number of message passing layers (Fortunato et al. 2022). They also explored the handling of varying mesh resolutions and observed that GNN models generally manage different resolutions effectively, provided they are trained on datasets that encompass a range of resolutions. This adaptability demonstrates the flexibility of GNNs in addressing diverse input mesh conditions commonly encountered in mechanics-related tasks. However, when using meshes with edge lengths that are much longer or shorter than those seen during training, predictive performance may suffer. Moreover, at very high mesh resolutions, the ability of GNNs to propagate information effectively can become a bottleneck. Multi-scale approaches may be necessary to maintain both efficiency and accuracy when dealing with intricate mechanical systems. While this multiscale MGN model contained only two levels of graph hierarchies, Shao et al. (2023) and Toshev et al. (2023) explored more levels in the graph hierarchies. Shao et al. (2023) designed a multiscale GNN to deal with multiscale graph data following a CNN U-Net-inspired encoder-decoder architecture. Toshev et al. (2023) developed an algebraic multigrid-based coarsening layer that coarsened the graph to smaller scales. They then summarised and extracted features through message passing at smaller graph scales. Finally, a graph recovery layer was applied to recover the coarse graph to the high-fidelity graph of the initial mesh scale.

Regarding GCN applications, attention-based GCN models were commonly adopted (He et al. 2022; Liu et al. 2022). He et al. (2022) developed a GCN model consisting three graph convolution layers and two spatial gradient attention layers with a residual learning block (ResBlock) module and a ResBlock with squeeze and excitation (ResSE) module. Additionally, Liu et al. (2022) used a multi-head attention mechanism to learn node relations in different representation subspaces. Experimental results showed that the ability of the attention mechanism to aggregate physical information of adjacent nodes was much better than that of average aggregation (Liu et al. 2022). Belbute-Peres et al. (2020) developed a CFD–GCN model that integrated a CFD solver with the GCN architecture. The CFD–GCN model operates on two types of graphs: a fine graph and a coarsened version of the fine graph. The coarsened graph was processed with a CFD simulator, while the fine mesh was processed simultaneously by GCNs. The results from the coarsened mesh simulation results were then upsampled and combined with the intermediate outputted from the fine graph. Additional layers of GCNs were then applied to the combined features, to output the final desired output values such as velocity and pressure fields at each node in the fine graph. In the CFD–GCN framework, the CFD simulator can be considered an additional 'layer' attached to the GCN architecture. An interface was built to enable the CFD solver to be integrated seamlessly as any other layer within the PyTorch library (Belbute-Peres et al. 2020). The results demonstrated that the CFD–GCN framework can improve the accuracy of CFD simulation results on the coarsened mesh (Belbute-Peres et al. 2020). Moreover, they discovered enhanced generalisability when compared to the conventional GCN baseline without the upsampling technique. CFD–GCN was able to approximate the characteristics of the unseen shockwave behaviour.

Moreover, some of these existing GNN-based models applied the laws of physics to the flow field prediction to further improve prediction accuracy. He et al. (2022) incorporated physics-based training utilising Navier–Stokes loss functions, and Quattromini et al. (2023) ensured that the predictions result satisfied the Reynolds–Averaged Navier–Stokes (RANS) equations, which were fundamental in fluid dynamics simulations.

Further discussions

It is noticed that current studies are mainly based on 2D laminar flow. Future work is expected to study large 3D simulations using GNNs as they often provide more accurate and realistic fluid dynamics results that satisfy industrial applications. For instance, 3D models can simulate wingtip vortices and induced drag, providing a more accurate and realistic analysis. Also, experimental fluid simulation results utilising particle image velocimetry (PIV) or digital image correlation (DIC) methods to track the movement of fluid particles could be adopted as a training dataset for GNNs to learn from experimental data.

Moreover, many GNN applications in the aerodynamics domain are predicting spatial–temporal properties of fluid fields. Although these incremental GNN models can capture the dynamics of mechanical systems, the inherent challenge arises regarding accumulated time-step errors, especially in long-term rollouts. At training time, the spatial–temporal MPNN network never sees unphysical states, such as the non-zero divergence in incompressible Navier–Stokes equations. However, those unphysical states can appear through error accumulation. One method to mitigate error accumulation includes introducing training noise, such as Gaussian noise, to perturb the input data. However, this approach often has to compensate for relatively lower prediction accuracy and needs to be more principled. Therefore, a better understanding of rollout error accumulations could be studied, and the role of training noise in error accumulations could be further explored.

Combining RNNs with GNN models may be an alternative approach that reduces the accumulated error.

3.2.3 Rheology of complex fluid

Another fluid mechanics problem was the rheological behaviour of complex fluids. Mahmoudabadbozchelou et al. (2022) proposed the Rheology-informed Graph Neural Networks (RhiGNets) which is capable of learning the stress response of a complex fluid with applied deformation through a limited number of experiments. In the RhiGNets model, the graph nodes represent particles, and the graph edges represent the interactions or forces between particles. The architecture of the RhiGNets model comprises three sections, encoder, decoder, and graph predictor. It can efficiently achieve accurate results for predicting the shear, elongation, or oscillation of complex fluids such as blood, yoghurt, or polymer solutions.

3.3 Application of GNNs in interdisciplinary mechanics-related domain

The interdisciplinary mechanics-related GNN applications are thoroughly reviewed in this section. We have categorised the literature into interdisciplinary mechanics-related domains by their commonalities into thermo-mechanics, biomechanics, fluid–structural interaction, and complex system dynamics subdomains. The thermo-mechanics subdomain involves applications that study the interaction between mechanical and thermal effects in mechanical systems. The biomechanics subdomain integrates mechanics and biology principles to effectively understand the mechanical behaviours of biological systems. The fluid structural interactions subdomain studies the coupling effects of deformable structures and fluids. Lastly, the complex system subdomain is categorised by multiple interacting systems with nonlinear dynamics. These systems frequently display chaotic dynamics, making predictions particularly challenging.

GNNs are exceptionally well-suited for modelling the intricate, interconnected, and nonlinear nature of the high-dimensional systems. There is a growing interest in adopting GNN applications to understand, predict, and control mechanical systems within these subdomains. A summary of the GNN applications in these subdomains can be found in Table 3. This section concisely discusses the data representations, GNN architecture, and model performance of each proposed GNN framework in these subdomains.

3.3.1 Thermo-mechanics

The existing studies in the thermos-mechanics subdomain are mainly based on predicting spatial–temporal thermal responses for additive manufacturing processes (Mozaffar et al. 2021; Xue et al. 2022).

Graph representation methods

For additive manufacturing applications, graph structures commonly represent FE meshes, with graph nodes representing mesh nodes and graph edges representing mesh edges. Node features included node types that indicated whether a node is an active node, layer height that represents the distance from the bottom of the substrate, and the inverse of the distance between nodes and the laser beam. Edge information was stored in the adjacency matrix, with distances between nodes encoded as edge features.

GNN algorithms

Table 3 GNN application summary in interdisciplinary mechanics-related domain

Applications	Descriptions	GNN frameworks	References
Thermo-mechanics	Thermal profile prediction; Additive manufactured objects	MPNN combined with RNN	Mozaffar et al. (2021)
	Thermal behaviour prediction; 3D electronics objects	Spatial–temporal MPNN	Sanchis-Alepuz and Stipsitz (2022)
Biomechanics	Tissue deformation prediction; Liver and brain	MPNN with physics-informed training (liver)	Dalton et al. (2023)
		GCN GraphSAGE and GraphConv (brain)	Salehi and Giannacopoulos (2022)
	Cardiac mechanics prediction; Left ventricle of the heart	MPNN (MGN and DeepEmulator)	Dalton et al. (2021)
		Multiscale MPNN	Dalton et al. (2022)
		MPNN with physics-informed training	Dalton et al. (2023)
Fluid–structure interaction	Dynamics prediction; Flag blown by the wind	Spatial–temporal multigraph MPNN	Pfaff et al. (2020)
Complex system dynamics	Dynamics Inference for complex systems such as ropes, bouncing balls, and splashing fluids	Spatial–temporal MPNN	Sanchez-Gonzalez et al. (2018)
		Constraint-based spatial–temporal MPNN	Rubanova et al. (2021)
	Motion inference of multiple rigid bodies colliding with one another	Spatial–temporal MPNN	Allen et al. (2022a)
	Complex 3D granular flow	Spatial–temporal MPNN	Mayr et al. (2023)

Sanchis-Alepuz and Stipsitz (2022) studied the spatial–temporal thermal behaviour of 3D electronics designs such as heat sinks and printed circuit boards using a spatial–temporal MPNN architecture. They discovered that the MPNN model was more computationally efficient than traditional FE simulations. However, they also discovered that although the one-step prediction error was very low, the accumulated error of the MPNN surrogate model increased significantly (Sanchis-Alepuz and Stipsitz 2022). To mitigate these accumulated errors in rollout tests, Mozaffar et al. (2021) proposed a recurrent graph neural network (RGNN) to model the spatial–temporal thermal profile of the laser powder bed fusion (LPBF) additive manufacturing process. The RGNN surrogate model comprised DeeperGCN layers, GRU modules that took the update of the DeeperGCN layer and a hidden stage as input to predict the temperature field at incremental test time. DeeperGCN proposed by Li et al. (2020) employed a residual-based design with skip connections, which empowered deeper and more efficient training of the GCN model compared to conventional GCN architectures. The performance of RGNN was compared to the conventional GNN framework. Results revealed that the RGNN model excelled in longer-term predictions of over 1000 time steps compared to the baseline GNN architecture.

Further discussions

The abovementioned GNN applications in the thermo-mechanics domain are all based on simulation results. With the continuous development of high-quality shared repositories of additive manufacturing experimental data, it is promising that more GNN applications could be developed by incorporating the experimental data to enhance the prediction accuracy and become more practical for real-world usage. Furthermore, traditional loss functions like MSE, commonly used in GNNs, are often limited in capturing the dynamics of complex systems, especially in interdisciplinary mechanics-related tasks. A promising future direction could involve incorporating different loss functions by integrating physically informed metrics. For instance, Andrieux (2023) expanded the use of Bregman divergence applied in the thermo-mechanics domain, showcasing a systematic construction of discrepancy measures between the fields of thermodynamic variables within the large domain covered by the convex framework.

3.3.2 Biomechanics

Current applications in the biomechanics domain are focused on high-fidelity soft tissue deformations such as the brain (Salehi and Giannacopoulos 2022), liver (Dalton et al. 2023) and emulating the cardiac mechanics of the left ventricle of the heart (Dalton et al. 2022, 2021).

Graph representation methods

The graph representation method for the biomechanics domain is mainly based on high-fidelity meshes. The graph nodes represent mesh nodes, graph edges represent mesh edges. Node features commonly include node types indicating boundary nodes and kinematic nodes subject to deformations, nodal coordinates, and physical properties. Edge features often include distance between two nodes (Dalton et al. 2021; Salehi and Giannacopoulos 2022). Moreover, to ensure the model input is independent of different coordinate systems, Dalton et al. (2023) used the unit vector in the fibre direction instead of nodal coordinates as node features. The geometric information was mainly stored in the edge features by incorporating the relative positions (Dalton et al. 2023). However, many message-passing steps may be required to propagate information within a dense set of FE nodes. To allow

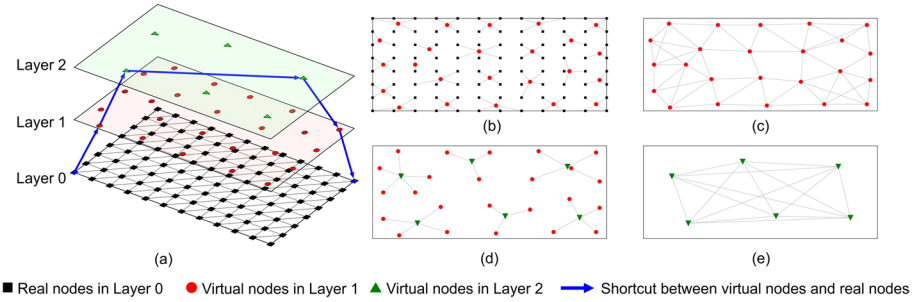

Fig. 12 A graph representation method proposed by Dalton et al. (2022). **a** Illustration of the hierarchical graph levels generated by adding virtual graph nodes and additional node elements. Blue arrows show how message passing is achieved based on the hierarchical graphs. (**b**–**e**) Illustration of the addition of virtual nodes and the generation of different graph hierarchies (layers) (Adapted from Dalton et al. (2022))

for more efficient message passing, Dalton et al. (2022) introduced a multi-scale data augmentation method to generate additional layers of coarse, virtual nodes that allow rapid message passing using a 'shortcut' through the augmented multi-scale spaces as illustrated in Fig. 12. Additional virtual nodes were generated based on the k-means algorithm. Corresponding edges were added between the real and virtual nodes. As shown in Fig. 12 below, the black squared nodes are real nodes from the original FE meshes, and the red circular nodes are virtual nodes created based on the positions of the real nodes. The green triangular nodes are the next graph hierarchy level of virtual nodes created based on the red circular virtual nodes (Dalton et al. 2022). Figure 12b–e shows the additional edges generated based on the virtual nodes and original nodes.

GNN algorithms

According to Table 3, MPNN and GCN models are the common GNN algorithms adopted by the biomechanics subdomain applications. Regarding MPNN-based architectures, Dalton et al. (2021) conducted a comparison study between the DeepEmulator and the MGN to investigate their performances in emulating the cardiac mechanics of the left ventricle (LV) of the heart. The DeepEmulator framework is a GNN framework proposed by Zheng et al. (2021), initially developed for the application of skinning-based animations of 3D characters. They discovered that the MGN model was two times more accurate compared to the DeepEmulator model. Driven by this discovery, Dalton et al. (2022) proposed a GNN framework named DeepGraphEmulator. The DeepGraphEmulator model adopted a similar encoder–processor–decoder architecture to MGN (Pfaff et al. 2020), but with an enhanced multi-scale data augmentation method (as previously introduced in Fig. 12). They discovered that the proposed DeepGraphEmulator could generalise to different systems of various LV geometries and material constitutive properties, making it promising for real-world practices. To further enhance the prediction accuracy, they combined physics-informed training with the proposed DeepGraphEmulator architecture by incorporating a potential energy function in the loss function (Dalton et al. 2023).

Regarding GCN models, Salehi and Giannacopoulos (2022) proposed the PhysGNN surrogate model combining two GCN modules, including GraphCONV and GraphSAGE, with Jumping Knowledge connections. The results showed that the proposed PhysGNN surrogate model could achieve an absolute displacement error of under 1 mm for 97% of the graph nodes, which satisfies the precision requirement in neurosurgery (Salehi and Giannacopoulos 2022).

Further discussions

Firstly, one significant challenge lies in accurately representing the unique muscle fibre arrangements in the LV of different patients. Each patient's LV geometry will have a unique myofibre field, which significantly impacts the mechanical response of the LV (Dalton et al. 2022). The current GNN models for cardiac mechanics often assume uniform material properties across all nodes. However, this assumption does not accurately reflect the variations in myofibre fields among different patients. It also does not account for regional differences of LV within a single patient, especially in cases of myocardial infarction (MI), where the myocardium exhibits higher stiffness levels in the affected regions (Dalton et al. 2022). To address this, future work should allow stiffness values to vary across different regions of the LV by inputting local values in the decoder stage rather than using a global value. Additionally, existing GNN models primarily focus on the passive phase of the cardiac cycle and neglect the active phase involving myocardial contraction. To provide a more comprehensive simulation of cardiac mechanics, future research should extend the modelling to include the active phase (Dalton et al. 2022).

Incorporating physics-based training in biomechanics is a relatively new topic requiring further exploration. The commonly applied strain energy density functions of the myocardium are considered slightly compressive (Dalton et al. 2023). To comprehensively address incompressibility, future work should incorporate multi-field variational principles, introducing additional variables to enforce the incompressibility constraint (Dalton et al. 2023). Additionally, preliminary studies have shown that physics-informed GNNs can match FE results across different mesh densities, suggesting that increasing mesh density can lead to more accurate and reliable results (Dalton et al. 2023). However, constructing and managing large graphs with increased mesh density can be resource-intensive, and this requires future research on optimising the GNN models and graph representation methods to achieve better computational efficiency.

3.3.3 Fluid–structure interaction

One spatial–temporal MPNN architecture named the MGN, mentioned in Sect. 3.1.1, was also adopted in various interdisciplinary mechanics-related domains, such as fluid–structure interactions (Pfaff et al. 2020).

Graph representation methods

The fundamental graph representation method is similar to those explained in Sect. 3.1.1. Additionally, Pfaff et al. (2020) also developed a learned adaptive re-meshing method across different time steps. This method involves learning a sizing field from the FE simulations that specifies the desired local resolution. The GNN model dynamically adjusts the mesh resolution during iterative rollouts by splitting, collapsing, and flipping edges based on the predicted sizing field at each timestep. This ensures a finer mesh resolution in areas with rapid changes or high gradients, such as regions with high stress in the deformed material, and a coarser mesh resolution in areas with low stress.

GNN algorithms

A comprehensive explanation of the MGN framework can be found in Sect. 3.1.1. The proposed model was applied to predict the evolution of fluid–structure interactions with the learned adaptive re-meshing method to improve prediction accuracy in small-scale features. It is noted that the MGN model had a robust capability to accurately predict out-of-distribution test datasets. For example, when trained on flat flag geometries containing

2000 nodes, the model demonstrated commendable adaptability by successfully extending their understanding to unseen cylindrical flag geometries comprising 20,000 nodes under identical air excitation conditions.

Further discussions

Regarding future work directions, the learned adaptive re-meshing method could be further investigated. Currently, the GNN surrogate model learns the sizing field based on training data generated by FE adaptive re-meshing algorithms. These methods are designed to improve numerical approximations of physical processes by increasing grid density in areas requiring higher resolution. However, GNNs excel in identifying and learning from structural and relational patterns within graph data, which is a fundamentally different approach from FE analysis. The FE adaptive mesh refinement method is tailored for numerical precision in physical simulations but may not optimise the graph structure for GNN processing and analysis. Therefore, a potential future direction is to explore optimal adaptive re-meshing methodologies specifically tailored for GNNs. A GNN-specific adaptive mesh might focus on optimising the graph topology to improve the efficiency of message passing and the prediction accuracy of the GNN model, rather than mimicking the mesh refinement methodologies used in numerical simulations. This could involve dynamic adjustments that prioritise connectivity and information flow within the graph, which are critical for GNN performance.

3.3.4 Complex systems dynamics

According to Table 3, GNN applications in complex systems dynamics primarily involve the dynamics inference for rigid body systems, including the movement of robot arm and pendulum (Sanchez-Gonzalez et al. 2018), contact between multiple rigid bodies (Allen et al. 2022a), movement of bouncing balls (Rubanova et al. 2021), and evolution of complex 3D granular flow (Mayr et al. 2023).

Graph representation methods

In these examples, the graph representation varied based on different shapes of geometries. For instance, the bodies and joints were depicted as graph nodes and edges when modelling the pendulum movement (Sanchez-Gonzalez et al. 2018). Additionally, accurately representing the boundary conditions when modelling the contact between multi-objects is crucial to ensure accurate predictions. Common approaches to model contacts between multi-objects are mainly based on node to node interactions; however, this may not always be effective, especially when the contact point occurs on mesh faces far from nodes. To compensate for this, Mayr et al. (2023) proposed the Boundary Graph Neural Networks (BGNNs) that dynamically alter the graph structure by adding virtual graph nodes or graph edges at the boundary surface to minimise the distance between the virtual graph nodes and the real fluid particle, to accurately depict the contact between real fluid particles and boundary surfaces. It was demonstrated that the BGNN model could accurately predict the motion of material particles over hundreds of time steps without applying physical constraints (Mayr et al. 2023).

Additionally, although incorporating additional nodes and edges in the augmented graph can improve accuracy, it also comes with high computational costs. To accurately depict the boundary conditions during collision without adding more nodes and edges, Allen et al. (2022a) introduced the Face Interaction Graph Network (FIGNet) that processed inputs on mesh faces rather than mesh nodes. The graph construction method was similar to the MGN (Pfaff et al. 2020), but a major modification was applied in the encoding

stage. During the encoding stage, Allen et al. (2022a) replaced node-to-node 'world edges' with face-to-face edges connecting the sender to the receiver face. By doing so, the FIGNet model outperformed the MGN regarding accuracy and computational efficiency.

GNN algorithm

Spatial–temporal MPNNs were commonly adopted for complex system dynamics to perform rollout tests (Allen et al. 2022a; Mayr et al. 2023; Rubanova et al. 2021; Sanchez-Gonzalez et al. 2018). In these GNN models, the encoder–processor–decoder architecture was employed by combining the node features from both the source and target nodes with the original edge features. The updated edge features are aggregated and combined with the target node features to output the updated node features. In the decoder layer, the node or graph-level features are concatenated through a fully connected neural network model to output final node prediction values.

Rubanova et al. (2021) introduced an MPNN-based physics simulator utilising an implicit constraint-based approach named C-GNS. This approach ensured that physical constraints were rigorously satisfied by integrating a scalar constraint function into an iterative optimisation process. The architecture employed an encoder–processor–decoder architecture based on the Graph Network-based Simulators (GNS) models proposed by Sanchez-Gonzalez et al. (2020). The GNS can be considered a predecessor of the MGN but was primarily designed for particle-based simulations, such as simulating the movement of fluid particles in a container. Compared to the GNS model, the C-GNS model had an additional scalar constraint function which quantified the degree to which a proposed state update violated physical constraints. The constraint function was employed within an iterative optimisation loop. The process began with an initial guess for the node positions. At each iteration, the model adjusted the node positions using the gradient of the constraint function with respect to the positions. The C-GNS model outperformed the GNS baseline models in terms of prediction accuracy and capability to generalise to novel and hand-designed constraints in various simulation scenarios.

Further discussions

It is noticed that the GNN applications in complex system dynamics were mainly based on examples with a relatively small number of nodes and edges. Future work could expand the dataset to incorporate diverse and larger graphs of more complex interactions.

3.3.5 Open data and source code

The open data and source codes for the applications of GNN models in the mechanics-related domains are organised in Table 4 to assist readers with similar research interests in reproducing previous studies and promote communication and future research on potential fields. These open data and source codes were originally provided by the authors of the papers discussed in Sect. 3.

4 Conclusions

To conclude, GNNs have gained significant traction in mechanics-related domains due to their unique ability to learn from graph-structured data. This review paper summarised various fundamental GNN algorithms, such as MPNN, GN, GCN, GAT, and GraphSAGE, and their applications in various domains. We systematically reviewed graph representation methods and GNN architectures across different mechanics-related domains,

Table 4 Summary table of open data and/or source codes

Descriptions	GNN frameworks	Open data and/or source codes	References
Solid mechanics applications			
Displacement field prediction: 2D elastostatic problem of cantilever beam	Multi-fidelity MPNN with subgraph analysis and physics-informed training	github.com/MCMB-Lab/MFGNNs	Black and Najafi (2022)
Displacement field prediction: 3D linear elastic and hyper-elastic beam	GCN with physics-informed training	github.com/Jasiuk-Research-Group	He et al. (2023)
Displacement field prediction: 3D Elastic plate subject to an actuator	Multiscale MPNN	github.com/Eydcao/BSMS-GNN	Cao et al. (2023)
Displacement field prediction: 2D/3D beam	GCN model with input-independent learnable weighted multi-channel aggregation	github.com/saurabhdeshpande93/convolution–aggregation–attention	Deshpande et al. (2024)
Displacement field prediction: 3D Hyper-elastic plate	Spatial–temporal multigraph MPNN	github.com/google-deepmind/deepmind-research/tree/master/meshgraphnets	Pfaff et al. (2020)
Wave propagation prediction: 2D shell metamaterial	MPNN with physics-based edge update function	github.com/tianjuxue/gnmm	Xue et al. (2023)
Generative modeling and inverse design: 3D truss metamaterial	Graph based VAE combined with a property predictor	zenodo.org/records/8255658	Zheng et al. (2023)
Crack field and displacement field prediction	Multi-scale MPNN	github.com/rperera12/Adaptive-mesh-based-Multiscale-Graph-Neural-Network	Perera and Agrawal (2024)
Fluid mechanics applications			
Eulerian method—Ocean, river simulations	Multiscale spatial–temporal MPNN with adaptive remeshing	github.com/afansi/multiscalegnn	Lino et al. (2022b)
	GCN with multiscale graph representation	github.com/trainsn/GNN-Surrogate	Shi et al. (2022)
Eulerian method: Incompressible laminar flow over aerofoil	Spatial–temporal MPNN	github.com/garrygale/NodeElementMessagePassing	Gao et al. (2024)
	GCN with GraphSAGE	github.com/BaratiLab/Airfoil-GCNN	Ogoke et al. (2021)
	GCN combined with CFD	github.com/locuslab/cfd-gcn	Belbute-Peres et al. (2020)
Eulerian method: Incompressible laminar flow over cylinder	MPNN with multiscale and graph level representation	github.com/baoshiaijhin/amgnet	Yang et al. (2022)
	Spatial–temporal MPNN	github.com/Litianyu141/My-CODE	Li et al. (2023c)

Table 4 (continued)

Descriptions	GNN frameworks	Open data and/or source codes	References
Eulerian method: Incompressible laminar flow over arbitrary shapes	GCN	github.com/jviquerat/gnn_laminar_flow	Chen et al. (2021)
Lagrangian method: Fluid fall and box bath	GCN	github.com/BaratiLab/FGN	Li and Farimani (2022)
	Spatial–temporal MPNN	github.com/deepmind/deepmind-research/tree/master/learning_to_simulate	Sanchez-Gonzalez et al. (2020)
Interdisciplinary mechanics-related applications			
Soft tissue deformation prediction	MPNN with physics-informed training	github.com/YasminSalehi/PhysGNN	Salehi and Giannacopoulos (2022)
Thermal profile prediction: Additive manufactured objects	MPNN combined with RNN	github.com/AMPL-NU/Graph_AM_Modeling	Mozaffar et al. (2021)
Complex 3D granular flow prediction	Spatial–temporal MPNN	github.com/ml-jku/bgnn	Mayr et al. (2023)
Dynamics prediction: flag blown by the wind	Spatial–temporal multigraph MPNN	github.com/google-deepmind/deepmind-research/tree/master/meshgraphnets	Pfaff et al. (2020)

Accessed 20 June 2024

specifically solid mechanics, fluid mechanics, and interdisciplinary mechanics-related domain. In the solid mechanics domain, we categorised GNN applications into continuum and frame structure, mechanical metamaterials, and fracture mechanics and tool wear predictions subdomains. In the fluid mechanics domain, we covered hydrodynamics, aerodynamics, and rheology subdomains. For the interdisciplinary mechanics-related domain, we explored thermo-mechanics, biomechanics, fluid–structural interaction, and complex system dynamics.

Our review highlights that in solid mechanics, most GNN applications focus on predicting displacement or stress fields of continuum structures, and structural response of mechanical metamaterials. Few studies have explored fracture mechanics, which represents a potential area for further research. In fluid mechanics, GNN applications are predominantly centred on 2D laminar flow, with fewer studies addressing turbulent modelling and rheology, both of which could benefit from more research. Compared to the solid mechanics and fluid dynamics domains, interdisciplinary mechanics-related domain that include multi-physics coupling are relatively underexplored. This is likely due to the complexity of problem formulation and the increased non-linearity that challenges GNN performance in this domain. However, with the continuous development of GNN architectures and novel graph construction methods, there is significant potential for applying GNNs in the interdisciplinary mechanics-related domain, making them worth exploring in future research.

Most of the reviewed GNN models outperformed traditional DNN baselines in terms of prediction accuracy and scalability. Additionally, some case studies demonstrated the generalisability potential of GNNs, showing their ability to handle previously unseen test data that exhibit similar physical behaviours to the training dataset. However, challenges that hinder the practical deployment of GNN models in the mechanics-related domains still need to be addressed. Therefore, challenges and future works tailored to specific types of mechanics-related tasks have been discussed in each subdomain. Common challenges in these subdomains include defining graph structures and mapping features more effectively, improving message-passing efficiencies in high-fidelity graphs, embedding material non-linearity, rollout for different size of timesteps. Additionally, addressing time series error accumulations, incorporating experimental data into the training loop to enhance prediction accuracy, integrating physics-based constraints into the training process, and investigating generalisability for more complex, high-dimensional, and large-scale real-world applications are essential areas for future research.

Most trained GNN models are capable of achieving rapid predictions within a few seconds, which is significantly faster than traditional FE simulations. Regarding training costs, researchers have employed various techniques, such as multi-scale graph training, to improve the computational efficiency of GNN models. However, more comprehensive research comparing the overall computational efficiency between GNN frameworks and FE simulations has yet to be conducted. The computational costs of graph data generation and training vary depending on the representation methods and GNN architectures used. These could be compared to the FE simulation time to comprehensively evaluate the efficiencies of GNN frameworks. Therefore, it would be beneficial for future research to focus more on a global investigation of the computational efficiency of GNN frameworks including both prediction costs as well as training and data processing costs.

Overall, this review paper compiles recent advancements at the intersection of graph theory and fundamental mechanics principles. It offers insights and a comprehensive overview for researchers looking to explore the capabilities of GNNs in addressing complex physics and mechanics-related tasks.

Author contributions Y.Z. and H.L. wrote the main manuscript, prepared figures, reviewed and revised the manuscript. H.Z., H.R.A., and T.P. reviewed the manuscript and provided valuable insights for editing. N.L. supervised and organised the work, reviewed and provided critical insights for editing the drafts.

Funding Partial financial support was received from EPSRC for the CASE Conversion DTP training grant (EP/R513052/1), UKRI Impact Acceleration Accounts (EP/X52556X/1), and Innovate UK smart grant (10083425).

Data Availability No datasets were generated or analysed during the current study.

Declarations

Conflict of interest The authors declare no competing interests. The author has applied a Creative Commons Attribution (CC BY) license to any Author Accepted Manuscript version arising.

References

Abadal S, Jain A, Guirado R, López-Alonso J, Alarcón E (2022) Computing graph neural networks: a survey from algorithms to accelerators. ACM Comput Surv 54:1–38. https://doi.org/10.1145/3477141

Adamczyk J (2022) Application of Graph Neural Networks and graph descriptors for graph classification. Preprint at https://arxiv.org/abs/2211.03666 (2022)

Addanki R, Battaglia PW, Budden D, Deac A, Godwin J, Keck T, Li WLS, Sanchez-Gonzalez A, Stott J, Thakoor S (2021) Large-scale graph representation learning with very deep gnns and self-supervision. Preprint at https://arxiv.org/abs/2107.09422 (2021)

Afshar Y, Bhatnagar S, Pan S, Duraisamy K, Kaushik S (2019) Prediction of aerodynamic flow fields using convolutional neural networks. Comput Mech 64:525–545

Al-Jarrah R, AL-Oqla FM (2022) A novel integrated BPNN/SNN artificial neural network for predicting the mechanical performance of green fibers for better composite manufacturing. Compos Struct 289:115475. https://doi.org/10.1016/j.compstruct.2022.115475

Allen KR, Rubanova Y, Lopez-Guevara T, Whitney W, Sanchez-Gonzalez A, Battaglia P, Pfaff T (2022a) Learning rigid dynamics with face interaction graph networks. Preprint at https://arxiv.org/abs/2212.03574 (2022a)

Allen KR, Lopez-Guevara T, Stachenfeld K, Sanchez-Gonzalez A, Battaglia P, Hamrick J, Pfaff T (2022b) Physical design using differentiable learned simulators. Preprint at https://arxiv.org/abs/2202.00728 (2022b)

Andrieux S (2023) Bregman divergences for physically informed discrepancy measures for learning and computation in thermomechanics. Comptes Rendus Mécanique 351:59–81. https://doi.org/10.5802/crmeca.164

Asif NA, Sarker Y, Chakrabortty RK, Ryan MJ, Ahamed MH, Saha DK, Badal FR, Das SK, Ali MF, Moyeen SI, Islam MR, Tasneem Z (2021) Graph Neural network: a comprehensive review on non-euclidean space. IEEE Access 9:60588–60606. https://doi.org/10.1109/ACCESS.2021.3071274

Attar H, Zhou H, Foster A, Li N (2021a) Rapid feasibility assessment of components to be formed through hot stamping: a deep learning approach. J Manuf Process 68:1650–1671. https://doi.org/10.1016/j.jmapro.2021.06.011

Attar HR, Zhou H, Li N (2021b) Deformation and thinning field prediction for HFQ® formed panel components using convolutional neural networks. IOP Conference Series: Materials Science and Engineering 1157:012079. https://doi.org/10.1088/1757-899X/1157/1/012079
Attar H, Foster A, Li N (2022) Optimisation of panel component regions subject to hot stamping constraints using a novel deep-learning-based platform. IOP Conference Series: Materials Science and Engineering 1270:012123. https://doi.org/10.1088/1757-899X/1270/1/012123
Attar HR, Foster A, Li N (2023) Development of a deep learning platform for sheet stamping geometry optimisation under manufacturing constraints. Eng Appl Artif Intell 123:106295. https://doi.org/10.1016/j.engappai.2023.106295
Bacciu D, Errica F, Micheli A, Podda M (2020) A gentle introduction to deep learning for graphs. Neural Netw 129:203–221. https://doi.org/10.1016/j.neunet.2020.06.006
Battaglia PW, Hamrick JB, Bapst V, Sanchez-Gonzalez A, Zambaldi V, Malinowski M, Tacchetti A, Raposo D, Santoro A, Faulkner R (2018) Relational inductive biases, deep learning, and graph networks. Preprint at https://arxiv.org/abs/1806.01261(2018).
Behbahani M, Behr M, Hormes M, Steinseifer U, Arora D, Coronado O, Pasquali M (2009) A review of computational fluid dynamics analysis of blood pumps. Eur J Appl Math 20:363–397
Belbute-Peres FDA, Economon T, Kolter Z (2020) Combining Differentiable PDE Solvers and Graph Neural Networks for Fluid Flow Prediction. In: Hal D, III, Aarti S (eds) Proceedings of the 37th international conference on machine learning, PMLR, Proceedings of Machine Learning Research, pp 2402–2411
Black N, Najafi AR (2022) Learning finite element convergence with the multi-fidelity graph neural network. Comput Methods Appl Mech Eng 397:115120. https://doi.org/10.1016/j.cma.2022.115120
Bolandi H, Li X, Salem T, Boddeti V, Lajnef N (2022a) Deep learning paradigm for prediction of stress distribution in damaged structural components with stress concentrations. Adv Eng Softw 173:103240. https://doi.org/10.1016/j.advengsoft.2022.103240
Bolandi H, Li X, Salem T, Boddeti V, Lajnef N (2022b) Bridging finite element and deep learning: High-resolution stress distribution prediction in structural components. Fron Struct Civil Eng, 16. https://doi.org/10.1007/s11709-022-0882-5
Brunton SL, Noack BR, Koumoutsakos P (2020) Machine learning for fluid mechanics. Annu Rev Fluid Mech 52:477–508
Bui K-HN, Cho J, Yi H (2022) Spatial-temporal graph neural network for traffic forecasting: an overview and open research issues. Appl Intell 52:2763–2774. https://doi.org/10.1007/s10489-021-02587-w
Cao Y, Chai M, Li M, Jiang C (2023) Efficient learning of mesh-based physical simulation with bi-stride multi-scale graph neural network. In: Proceedings of the 40th international conference on machine learning, JMLR.org, Honolulu, Hawaii, USA, pp Article 145
Chen J, Hachem E, Viquerat J (2021) Graph neural networks for laminar flow prediction around random two-dimensional shapes. Phys Fluids 33:123607
Chen J, Li Y, Liu X, Deng T (2022) A data-driven minimum stiffness prediction method for machining regions of aircraft structural parts. Int J Adv Manufact Technol 120:3609–3623. https://doi.org/10.1007/s00170-022-08991-x
Chen Q, Cao J, Lin W, Zhu S, Wang S (2024) Predicting dynamic responses of continuous deformable bodies: a graph-based learning approach. Comput Methods Appl Mech Eng 420:116669. https://doi.org/10.1016/j.cma.2023.116669
Dalton D, Husmeier D, Gao H (2023) Physics-informed graph neural network emulation of soft-tissue mechanics. Comput Methods Appl Mech Eng 417:116351. https://doi.org/10.1016/j.cma.2023.116351
Dalton D, Lazarus A, Rabbani A, Gao H, Husmeier D (2021) Graph neural network emulation of cardiac mechanics. In: 3rd International conference on statistics: theory and applications (ICSTA'21), pp. Article 127.
Dalton D, Gao H, Husmeier D (2022) Emulation of cardiac mechanics using Graph Neural Networks. Comp Methods Appl Mech Eng, p 401.https://doi.org/10.1016/j.cma.2022.115645.
Defferrard M, Bresson X, Vandergheynst P (2016) Convolutional neural networks on graphs with fast localized spectral filtering. In: Proceedings of the 30th international conference on neural information processing systems. Curran Associates Inc., Barcelona, Spain, pp 3844–3852
Deng Y (2022) Recommender systems based on graph embedding techniques: a review. IEEE Access 10:51587–51633. https://doi.org/10.1109/access.2022.3174197
Deshpande S, Bordas SPA, Lengiewicz J (2024) MAgNET: a graph U-Net architecture for mesh-based simulations. Eng Appl Artif Intell 133:108055. https://doi.org/10.1016/j.engappai.2024.108055
Dold D, Aranguren van Egmond D (2023) Differentiable graph-structured models for inverse design of lattice materials. Cell Reports Phys Sci 4:101586. https://doi.org/10.1016/j.xcrp.2023.101586

Dupuy D, Lapeyre C, Odier N, Papadogiannis D (2023) Modeling the wall shear stress in large-eddy simulation using graph neural networks. Data-Centric Eng 4:e7. https://doi.org/10.1017/dce.2023.2
Fortunato M, Pfaff T, Wirnsberger P, Pritzel A, Battaglia P (2022) Multiscale meshgraphnets. Preprint at https://arxiv.org/abs/2210.00612
Fu X, Zhou F, Peddireddy D, Kang Z, Jun MB-G, Aggarwal V (2023) An finite element analysis surrogate model with boundary oriented graph embedding approach for rapid design. J Comput Des Eng 10:1026–1046. https://doi.org/10.1093/jcde/qwad025
Gao H, Zahr MJ, Wang J-X (2022) Physics-informed graph neural galerkin networks: a unified framework for solving pde-governed forward and inverse problems. Comput Methods Appl Mech Eng 390:114502
Gao C, Zheng Y, Li N, Li Y, Qin Y, Piao J, Quan Y, Chang J, Jin D, He X, Li Y (2023) A survey of graph neural networks for recommender systems: challenges, methods, and directions. ACM Trans Recommender Syst. https://doi.org/10.1145/3568022
Gao R, Deo IK, Jaiman RK (2024) A finite element-inspired hypergraph neural network: application to fluid dynamics simulations. J Comput Phys 504:112866. https://doi.org/10.1016/j.jcp.2024.112866
Georgousis S, Kenning MP, Xie X (2021) Graph deep learning: state of the art and challenges. IEEE Access 9:22106–22140. https://doi.org/10.1109/ACCESS.2021.3055280
Gilmer J, Schoenholz SS, Riley PF, Vinyals O, Dahl GE (2017) Neural Message Passing for Quantum Chemistry. In: Doina P, Yee Whye T (eds) Proceedings of the 34th international conference on machine learning, PMLR, Proceedings of Machine Learning Research, pp 1263--1272
Goodfellow I, Bengio Y, Courville A (2016) Deep learning. MIT Press, Cambridge
Groen D, Zasada SJ, Coveney PV (2012) Survey of multiscale and multiphysics applications and communities. Comput Sci Eng 16:34–43
Guo S, Agarwal M, Cooper C, Tian Q, Gao RX, Grace WG, Guo Y (2022) Machine learning for metal additive manufacturing: Towards a physics-informed data-driven paradigm. J Manuf Syst 62:145–163
Guo K, Buehler MJ (2020) A semi-supervised approach to architected materials design using graph neural networks. Extreme Mech Lett, p 41. https://doi.org/10.1016/j.eml.2020.101029
Gupta A, Matta P, Pant B (2021) Graph neural network: current state of art, challenges and applications. Mater Today Proc 46:10927–10932. https://doi.org/10.1016/j.matpr.2021.01.950
Hamilton WL, Ying R, Leskovec J (2017) Inductive representation learning on large graphs. In: Proceedings of the 31st international conference on neural information processing systems. Curran Associates Inc., Long Beach, pp 1025–1035
Harlow FH, Hirt CW (1972) Recent extensions to Eulerian methods for numerical fluid dynamics. USSR Comput Math Math Phys 12:123–141. https://doi.org/10.1016/0041-5553(72)90038-9
He X, Wang Y, Li J (2022) Flow completion network: Inferring the fluid dynamics from incomplete flow information using graph neural networks. Phys Fluids 34:087114
He J, Abueidda D, Koric S, Jasiuk I (2023) On the use of graph neural networks and shape-function-based gradient computation in the deep energy method. Int J Numer Meth Eng 124:864–879. https://doi.org/10.1002/nme.7146
Hou W, Darakananda D, Eldredge J (2019) Machine learning based detection of flow disturbances using surface pressure measurements.
Iman RL, Davenport JM, Zeigler DK (1980) Latin hypercube sampling (program user's guide). United States, pp Medium: X; Size: Pages: 77
Indurkar PP, Karlapati S, Shaikeea AJD, Deshpande VS (2022) Predicting deformation mechanisms in architected metamaterials using GNN. Preprint at https://arxiv.org/abs/2202.09427 (2022)
James G, Witten D, Hastie T, Tibshirani R (2021) An introduction to statistical learning. Springer, New York
Janiesch C, Zschech P, Heinrich K (2021) Machine learning and deep learning. Electron Mark 31:685–695. https://doi.org/10.1007/s12525-021-00475-2
Jessica LSE, Arafat NA, Lim WX, Chan WL, Kong AWK (2023), Finite volume features, global geometry representations, and residual training for deep learning-based CFD simulation. Preprint at https://arxiv.org/abs/2311.14464 (2023),
Jiang W, Luo J (2022) Graph neural network for traffic forecasting: a survey. Expert Syst Appl 207:117921. https://doi.org/10.1016/j.eswa.2022.117921
Kantzos C, Lao J, Rollett A (2019) Design of an interpretable convolutional neural network for stress concentration prediction in rough surfaces. Mater Charact 158:109961. https://doi.org/10.1016/j.matchar.2019.109961
Khadilkar A, Wang J, Rai R (2019) Deep learning–based stress prediction for bottom-up SLA 3D printing process. Int J Adv Manufact Technol, 102. https://doi.org/10.1007/s00170-019-03363-4

Kipf TN, Welling M (2016) Semi-supervised classification with graph convolutional networks. Preprint at https://arxiv.org/abs/1609.02907 (2016)
Kumar S, Kochmann DM (2022) What machine learning can do for computational solid mechanics. In: Current trends and open problems in computational mechanics. Springer, Cham, pp 275–285
Kutz JN (2017) Deep learning in fluid dynamics. J Fluid Mech 814:1–4. https://doi.org/10.1017/jfm.2016.803
Lebon F, Ramière I (2023) Advanced numerical methods in computational solid mechanics. Mathematics 11:1512. https://doi.org/10.3390/math11061512
Li Z, Farimani AB (2022) Graph neural network-accelerated Lagrangian fluid simulation. Comput Graph 103:201–211. https://doi.org/10.1016/j.cag.2022.02.004
Li J, Liu T, Zhu G, Li Y, Xie Y (2023a) Uncertainty quantification and aerodynamic robust optimization of turbomachinery based on graph learning methods. Energy 273:127289. https://doi.org/10.1016/j.energy.2023.127289
Li R, Yuan X, Radfar M, Marendy P, Ni W, O'Brien TJ, Casillas-Espinosa PM (2023b) Graph signal processing, graph neural network and graph learning on biological data: a systematic review. IEEE Rev Biomed Eng 16:109–135. https://doi.org/10.1109/RBME.2021.3122522
Li G, Xiong C, Thabet A, Ghanem B (2020) Deepergcn: all you need to train deeper gcns. Preprint at https://arxiv.org/abs/2006.07739 (2020)
Li T, Zhou S, Chang X, Zhang L, Deng X (2023c) Finite Volume Graph Network (FVGN): Predicting unsteady incompressible fluid dynamics with finite volume informed neural network. Preprint at https://arxiv.org/abs/2309.10050 (2023c)
Liang F, Qian C, Yu W, Griffith D, Golmie N (2022) Survey of graph neural networks and applications. Wirel Commun Mob Comput 2022:9261537. https://doi.org/10.1155/2022/9261537
Liang L, Liu M, Martin C, Sun W (2018) A deep learning approach to estimate stress distribution: a fast and accurate surrogate of finite-element analysis. J R Soc Interface 15. https://doi.org/10.1098/rsif.2017.0844
Liao W, Bak-Jensen B, Pillai JR, Wang Y, Wang Y (2022) A review of graph neural networks and their applications in power systems. J Modern Power Syst Clean Energy 10:345–360. https://doi.org/10.35833/MPCE.2021.000058
Lino M, Fotiadis S, Bharath AA, Cantwell CD (2022a) Multi-scale rotation-equivariant graph neural networks for unsteady Eulerian fluid dynamics. Phys Fluids, 34. https://doi.org/10.1063/5.0097679
Lino M, Fotiadis S, Bharath AA, Cantwell CD (2022b) Towards fast simulation of environmental fluid mechanics with multi-scale graph neural networks. Preprint at https://arxiv.org/abs/2205.02637.
Lipton ZC, Berkowitz J, Elkan C (2015) A critical review of recurrent neural networks for sequence learning. Preprint at https://arxiv.org/abs/1506.00019 (2015)
Liu Q, Zhu W, Jia X, Ma F, Gao Y (2022) Fluid simulation system based on graph neural network. Preprint at https://arxiv.org/abs/2202.12619 (2022)
Lopera DS, Servadei L, Kiprit GN, Hazra S, Wille R, Ecker W (2021) A survey of graph neural networks for electronic design automation. In: 2021 ACM/IEEE 3rd workshop on machine learning for CAD (MLCAD), pp 1–6
Lukovnikov D, Lehmann J, Fischer A (2020) Improving the long-range performance of gated graph neural networks. Preprint at https://arxiv.org/abs/2007.09668 (2022)
Luo Y, Mesgarani N (2019) Conv-TasNet: surpassing ideal time–frequency magnitude masking for speech separation. IEEE/ACM Trans Audio Speech Language Process 27:1256–1266. https://doi.org/10.1109/TASLP.2019.2915167
Mahmoudabadbozchelou M, Kamani KM, Rogers SA, Jamali S (2022) Digital rheometer twins: Learning the hidden rheology of complex fluids through rheology-informed graph neural networks. Proc Natl Acad Sci 119:e2202234119. https://doi.org/10.1073/pnas.2202234119
Malekzadeh M, Hajibabaee P, Heidari M, Zad S, Uzuner O, Jones JH (2021) Review of graph neural network in text classification. In: 2021 IEEE 12th annual ubiquitous computing, electronics & mobile communication conference (UEMCON), pp 0084–0091
Mandelli L, Berretti S (2022) CAD 3D Model classification by graph neural networks: a new approach based on STEP format. Preprint at https://arxiv.org/abs/2210.16815 (2022)
Mayr A, Lehner S, Mayrhofer A, Kloss C, Hochreiter S, Brandstetter J (2023) Boundary graph neural networks for 3D simulations. In: Proceedings of the thirty-seventh AAAI conference on artificial intelligence and thirty-fifth conference on innovative applications of artificial intelligence and thirteenth symposium on educational advances in artificial intelligence. AAAI Press, Cambridge, pp Article 1023

Meyer PP, Bonatti C, Tancogne-Dejean T, Mohr D (2022) Graph-based metamaterials: deep learning of structure-property relations. Mater Des 223:111175. https://doi.org/10.1016/j.matdes.2022.111175
Monti F, Boscaini D, Masci J, Rodola E, Svoboda J, Bronstein MM (2017) Geometric deep learning on graphs and manifolds using mixture model cnns. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 5115–5124
Morimoto M, Fukami K, Fukagata K (2021) Experimental velocity data estimation for imperfect particle images using machine learning. Phys Fluids 33. https://doi.org/10.1063/5.0060760
Mozaffar M, Liao S, Lin H, Ehmann K, Cao J (2021) Geometry-agnostic data-driven thermal modeling of additive manufacturing processes using graph neural networks. Addit Manufact, p 48. https://doi.org/10.1016/j.addma.2021.102449
Nakamura T, Fukami K, Fukagata K (2022) Identifying key differences between linear stochastic estimation and neural networks for fluid flow regressions. Sci Rep 12:3726. https://doi.org/10.1038/s41598-022-07515-7
Narayanasamy R, Padmanabhan P (2012) Comparison of regression and artificial neural network model for the prediction of springback during air bending process of interstitial free steel sheet. J Intell Manuf 23:357–364. https://doi.org/10.1007/s10845-009-0375-6
Nguyen-Thanh VM, Zhuang X, Rabczuk T (2020) A deep energy method for finite deformation hyperelasticity. Eur J Mech A Solids 80:103874. https://doi.org/10.1016/j.euromechsol.2019.103874
Nie Z, Jiang H, Kara LB (2019) Stress field prediction in cantilevered structures using convolutional neural networks. J Comput Inform Sci Eng, 20. https://doi.org/10.1115/1.4044097
Ogoke F, Meidani K, Hashemi A, Farimani AB (2021) Graph convolutional networks applied to unstructured flow field data. Mach Learn Sci Technol, 2. https://doi.org/10.1088/2632-2153/ac1fc9
Perera R, Agrawal V (2024) Multiscale graph neural networks with adaptive mesh refinement for accelerating mesh-based simulations. Comput Methods Appl Mech Eng 429:117152. https://doi.org/10.1016/j.cma.2024.117152
Perera R, Guzzetti D, Agrawal V (2022) Graph neural networks for simulating crack coalescence and propagation in brittle materials. Comput Methods Appl Mech Eng 395:115021. https://doi.org/10.1016/j.cma.2022.115021
Pfaff T, Fortunato M, Sanchez-Gonzalez A, Battaglia PW (2020) Learning mesh-based simulation with graph networks. Preprint at https://arxiv.org/abs/2010.03409 (2020)
Pietrzyk M (2000) Finite-element simulation of large plastic deformation. J Mater Process Technol 106:223–229. https://doi.org/10.1016/S0924-0136(00)00618-X
Pinto RN, Afzal A, D'Souza LV, Ansari Z, Mohammed Samee A (2017) Computational fluid dynamics in turbomachinery: a review of state of the art. Arch Comput Methods Eng 24:467–479
Prachaseree P, Lejeune E (2022) Learning mechanically driven emergent behavior with message passing neural networks. Comput Struct 270:106825. https://doi.org/10.1016/j.compstruc.2022.106825
Provost ML, Hou W, Eldredge J (2020) Deep learning and data assimilation approaches to sensor reduction in estimation of disturbed separated flows. In: AIAA Scitech 2020 Forum.
Qi CR, Su H, Mo K, Guibas LJ (2017) Pointnet: deep learning on point sets for 3d classification and segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 652–660
Quattromini M, Bucci MA, Cherubini S, Semeraro O (2023) Operator learning of RANS equations: a Graph Neural Network closure model. Preprint at https://arxiv.org/abs/2303.03806 (2023)
Rabault J, Kolaas J, Jensen A (2017) Performing particle image velocimetry using artificial neural networks: a proof-of-concept. Meas Sci Technol 28:125301. https://doi.org/10.1088/1361-6501/aa8b87
Reiser P, Neubert M, Eberhard A, Torresi L, Zhou C, Shao C, Metni H, van Hoesel C, Schopmans H, Sommer T, Friederich P (2022) Graph neural networks for materials science and chemistry. Commun Mater 3:93. https://doi.org/10.1038/s43246-022-00315-6
Ronneberger O, Fischer P, Brox T, Navab N, Hornegger J, Wells WM, Frangi AF (2015) Medical image computing and computer-assisted intervention – MICCAI 2015 18th International Conference Munich Germany October 5-9 2015 Proceedings Part III U-Net: Convol Net Bio Image Segmentation Springer Int Publishing Cham 234–241
Ross E, Hambleton D (2021) Using graph neural networks to approximate mechanical response on 3D lattice structures.In: Proceedings of AAG2020-advances in architectural geometry, vol 24, pp 466–485.
Rubanova Y, Sanchez-Gonzalez A, Pfaff T, Battaglia P (2021) Constraint-based graph network simulator. Preprint at https://arxiv.org/abs/2112.09161 (2021)
Salehi Y, Giannacopoulos D (2022) PhysGNN: a physics—driven graph neural network based model for predicting soft tissue deformation in image-guided neurosurgery. Adv Neural Inf Process Syst 35:37282–37296

Sanchez-Gonzalez A, Heess N, Springenberg JT, Merel J, Riedmiller M, Hadsell R, Battaglia P (2018) Graph networks as learnable physics engines for inference and control. In: International conference on machine learning, PMLR, pp 4470–4479
Sanchez-Gonzalez A, Godwin J, Pfaff T, Ying R, Leskovec J, Battaglia PW (2020) Learning to simulate complex physics with graph networks. In: Proceedings of the 37th international conference on machine learning, JMLR.org, pp Article 784
Sanchis-Alepuz H, Stipsitz M (2022) Towards real time thermal simulations for design optimization using graph neural networks. In: 2022 IEEE design methodologies conference (DMC). IEEE, New York, pp 1–6
Scarselli F, Gori M, Tsoi AC, Hagenbuchner M, Monfardini G (2009) The graph neural network model. IEEE Trans Neural Networks 20:61–80. https://doi.org/10.1109/TNN.2008.2005605
Schmidhuber J (2015) Deep learning in neural networks: an overview. Neural Netw 61:85–117. https://doi.org/10.1016/j.neunet.2014.09.003
Shao X, Liu Z, Zhang S, Zhao Z, Hu C (2023) PIGNN-CFD: a physics-informed graph neural network for rapid predicting urban wind field defined on unstructured mesh. Build Environ 232:110056. https://doi.org/10.1016/j.buildenv.2023.110056
Sharma SK, Kalamkar VR (2016) Computational fluid dynamics approach in thermo-hydraulic analysis of flow in ducts with rib roughened walls—a review. Renew Sustain Energy Rev 55:756–788. https://doi.org/10.1016/j.rser.2015.10.160
Shi N, Xu J, Wurster SW, Guo H, Woodring J, Van Roekel LP, Shen H-W (2022) GNN-Surrogate: a hierarchical and adaptive graph neural network for parameter space exploration of unstructured-mesh ocean simulations. IEEE Trans Visual Comput Graphics 28:2301–2313
Shivaditya MV, Alves J, Bugiotti F, Magoulès F (2022) Graph neural network-based surrogate models for finite element analysis. In: 2022 21st International symposium on distributed computing and applications for business engineering and science (DCABES). IEEE, New York, pp 54–57
Shuman DI, Narang SK, Frossard P, Ortega A, Vandergheynst P (2013) The emerging field of signal processing on graphs: extending high-dimensional data analysis to networks and other irregular domains. IEEE Signal Process Mag 30:83–98. https://doi.org/10.1109/MSP.2012.2235192
Sperduti A, Starita A (1997) Supervised neural networks for the classification of structures. IEEE Trans Neural Netw 8:714–735. https://doi.org/10.1109/72.572108
Spruegel TC, Schröppel T, Wartzack S (2017) Generic approach to plausibility checks for structural mechanics with deep learning. In: 21st international conference On engineering design, ICED17, Vancouver, Canada.
Surjadi JU, Gao L, Du H, Li X, Xiong X, Fang NX, Lu Y (2019) Mechanical metamaterials and their engineering applications. Adv Eng Mater 21:1800864. https://doi.org/10.1002/adem.201800864
Tamaddon-Jahromi HR, Chakshu NK, Sazonov I, Evans LM, Thomas H, Nithiarasu P (2020) Data-driven inverse modelling through neural network (deep learning) and computational heat transfer. Comput Methods Appl Mech Eng 369:113217. https://doi.org/10.1016/j.cma.2020.113217
Tian Y, Guan X, Sun H, Bao Y (2024) An adaptive structural dominant failure modes searching method based on graph neural network. Reliab Eng Syst Saf 243:109841. https://doi.org/10.1016/j.ress.2023.109841
Toshev AP, Galletti G, Brandstetter J, Adami S, Adams NA (2023) E(3) Equivariant graph neural networks for particle-based fluid mechanics. Preprint at https://arxiv.org/abs/2304.00150 (2023).
Veličković P (2023) Everything is connected: graph neural networks. Curr Opin Struct Biol 79:102538. https://doi.org/10.1016/j.sbi.2023.102538
Veličković P, Cucurull G, Casanova A, Romero A, Lio P, Bengio Y (2017) Graph attention networks. Preprint at https://arxiv.org/abs/1710.10903 (2017).
Viquerat J, Hachem E (2020) A supervised neural network for drag prediction of arbitrary 2D shapes in laminar flows at low Reynolds number. Comput Fluids 210:104645. https://doi.org/10.1016/j.compfluid.2020.104645
Wang H (2023) Description of fluid motion. A guide to fluid mechanics. Cambridge University Press, Cambridge, pp 34–56
Wang H, Qin Q-H (2020) Methods of Fundamental Solutions in Solid Mechanics. Elsevier, Amsterdam
Wang Y, Sun Y, Liu Z, Sarma SE, Bronstein MM, Solomon JM (2019) Dynamic graph cnn for learning on point clouds. ACM Trans Graphics (tog) 38:1–12
Wang S, Hu L, Wang Y, He X, Sheng QZ, Orgun M, Cao L, Wang N, Ricci F, Yu PS (2020) Graph learning approaches to recommender systems: a review. Preprint at https://arxiv.org/abs/2004.11718 (2020)
Warey A, Chakravarty R (2022) Classification of computer aided engineering (CAE) parts using graph convolutional networks.Preprint at https://arxiv.org/abs/2202.11289 (2022)
Wieder O, Kohlbacher S, Kuenemann M, Garon A, Ducrot P, Seidel T, Langer T (2020) A compact review of molecular property prediction with graph neural networks. Drug Discov Today Technol 37:1–12. https://doi.org/10.1016/j.ddtec.2020.11.009

Wong JC, Ooi CC, Chattoraj J, Lestandi L, Dong G, Kizhakkinan U, Rosen DW, Jhon MH, Dao MH (2022) Graph neural network based surrogate model of physics simulations for geometry design. In: 2022 IEEE symposium series on computational intelligence (SSCI), pp 1469–1475
Wu Z, Pan S, Chen F, Long G, Zhang C, Yu PS (2021) A comprehensive survey on graph neural networks. IEEE Trans Neural Netw Learn Syst 32:4–24. https://doi.org/10.1109/TNNLS.2020.2978386
Wu S, Sun F, Zhang W, Xie X, Cui B (2023) Graph neural networks in recommender systems: a survey. ACM Comput Surv 55:1–37. https://doi.org/10.1145/3535101
Xiong J, Xiong Z, Chen K, Jiang H, Zheng M (2021) Graph neural networks for automated de novo drug design. Drug Discovery Today 26:1382–1393. https://doi.org/10.1016/j.drudis.2021.02.011
Xu Q, Nie Z, Xu H, Zhou H, Attar HR, Li N, Xie F, Liu X-J (2021) SuperMeshing: a new deep learning architecture for increasing the mesh density of physical fields in metal forming numerical simulation. J Appl Mech 89. https://doi.org/10.1115/1.4052195
Xue T, Adriaenssens S, Mao S (2023) Learning the nonlinear dynamics of mechanical metamaterials with graph networks. Int J Mech Sci 238:107835. https://doi.org/10.1016/j.ijmecsci.2022.107835
Xue T, Gan Z, Liao S, Cao J (2022) Physics-embedded graph network for accelerating phase-field simulation of microstructure evolution in additive manufacturing. npj Comput Mate 8:201. https://doi.org/10.1038/s41524-022-00890-9
Yang Z, Dong Y, Deng X, Zhang L (2022) AMGNET: multi-scale graph neural networks for flow field prediction. Connect Sci 34:2500–2519. https://doi.org/10.1080/09540091.2022.2131737
Ye J, Zhao J, Ye K, Xu C (2022) How to build a graph-based deep learning architecture in traffic domain: a survey. IEEE Trans Intell Transp Syst 23:3904–3924. https://doi.org/10.1109/tits.2020.3043250
Yi H-C, You Z-H, Huang D-S, Kwoh CK (2022) Graph representation learning in bioinformatics: trends, methods and applications. Briefings Bioinform 23:bbab340. https://doi.org/10.1093/bib/bbab340
Zhang Z, Chen L, Zhong F, Wang D, Jiang J, Zhang S, Jiang H, Zheng M, Li X (2022) Graph neural network approaches for drug-target interactions. Curr Opin Struct Biol 73:102327. https://doi.org/10.1016/j.sbi.2021.102327
Zhang Y, Sung W, Mavris DN (2017) Application of convolutional neural network to predict airfoil lift coefficient. Preprint at https://arxiv.org/abs/1712.10082 (2017)
Zhang X-M, Liang L, Liu L, Tang M-J (2021) Graph neural networks and their current applications in bioinformatics. Front Genet 12. https://doi.org/10.3389/fgene.2021.690049
Zhao P, Liao W, Huang Y, Lu X (2023) Intelligent beam layout design for frame structure based on graph neural networks. J Build Eng 63:105499. https://doi.org/10.1016/j.jobe.2022.105499
Zheng L, Karapiperis K, Kumar S, Kochmann DM (2023) Unifying the design space and optimizing linear and nonlinear truss metamaterials by generative modeling. Nat Commun 14:7563. https://doi.org/10.1038/s41467-023-42068-x
Zheng M, Zhou Y, Ceylan D, Barbic J (2021) A deep emulator for secondary motion of 3d characters. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 5932–5940
Zheng X, Liu Y, Pan S, Zhang M, Jin D, Yu PS (2022) Graph neural networks for graphs with heterophily: a survey. Preprint at https://arxiv.org/abs/2202.07082 (2022)
Zhou JM, Dong L, Guan W, Yan J (2019) Impact load identification of nonlinear structures using deep recurrent neural network. Mech Syst Signal Process 133:106292. https://doi.org/10.1016/j.ymssp.2019.106292
Zhou J, Cui G, Hu S, Zhang Z, Yang C, Liu Z, Wang L, Li C, Sun M (2020) Graph neural networks: a review of methods and applications. AI Open 1:57–81. https://doi.org/10.1016/j.aiopen.2021.01.001
Zhou H, Xu Q, Nie Z, Li N (2021) A study on using image-based machine learning methods to develop surrogate models of stamp forming simulations. J Manuf Sci Eng 144:1–41. https://doi.org/10.1115/1.4051604
Zhou Y, Zheng H, Huang X, Hao S, Li D, Zhao J (2022) Graph neural networks: taxonomy, advances, and trends. ACM Trans Intell Syst Technol 13:1–54. https://doi.org/10.1145/3495161
Zhu L, Li N (2023) Springback prediction for sheet metal cold stamping using convolutional neural networks. In: 2022 workshop on electronics communication engineering, SPIE, pp 278–283
Zijian Wang RS, Timson Y (2022) Exploring graph neural networks for semantic enrichment: room type classification. Automat Construct 134. https://doi.org/10.1016/j.autcon.2021.104039.