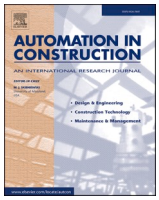
Automation in Construction 154 (2023) 104984

Contents lists available at ScienceDirect

Automation in Construction

journal homepage: www.elsevier.com/locate/autcon



Review

Graph neural networks for construction applications

Yilong Jia [a], Jun Wang [b,*], Wenchi Shou [b,*], M. Reza Hosseini [a], Yu Bai [c]

[a] School of Architecture and Built Environment, Faculty of Science, Engineering and Built Environment, Deakin University, Geelong, VIC 3220, Australia
[b] School of Engineering, Design and Built Environment, Western Sydney University, Penrith, NSW 2751, Australia
[c] Department of Civil Engineering, Faculty of Engineering, Monash University, Clayton, VIC 3800, Australia

ARTICLE INFO

Keywords:
Graph neural networks
Machine learning
Artificial intelligence
Architecture
Engineering

ABSTRACT

Graph Neural Networks (GNNs) have emerged as a promising solution for effectively handling non-Euclidean data in construction, including building information models (BIM) and scanned point clouds. However, despite their potential, there is a lack of comprehensive scholarly work providing a holistic understanding of the application of GNNs in the construction domain. This paper addresses this gap by conducting a thorough review of 34 publications on GNNs in construction, presenting a comprehensive overview of the current research landscape. By analyzing the existing literature, this paper aims to identify opportunities and challenges for further advancing the application of GNNs in construction. The findings from this review shed light on diverse approaches for constructing graph data from common construction data types and demonstrate the significant potential of GNNs for the industry. Moreover, this paper contributes to the existing body of knowledge by increasing awareness of the current state of GNNs in the construction industry and offering practical recommendations to overcome challenges in real-world practice.

1. Introduction

Followed by the success of AlexNet for image classification, deep learning has achieved significant success in various industries, such as chemistry, biology and transportation [1,2]. Like other industries, the application of deep learning in construction is growing. Correspondingly, research activity in the domain has increased – from three papers in 2016 to 103 papers in 2019 – and is expected to rise continuously [3]. Convolutional neural networks (CNNs), recurrent neural networks (RNNs) and deep neural networks (DNNs) are the most commonly used deep learning networks in the construction industry [4]. These networks have been applied to several aspects of construction. For example, CNNs have been used to automatically identify and classify various types of highway cracks from images [5]; RNNs can accurately extract semantic-rich information for advanced working packaging [6]; DNNs were adopted to forecast 24-h ahead building cooling load [7].

These traditional neural networks can successfully process data structures with fixed forms, such as text and images, but they cannot deal with graphs [8,9]. A graph typically consists of a variable number of unordered nodes and edges. Graphs can present many forms of data in various application areas. For example, road networks are modeled in traffic engineering as graphs consisting of crossings and roads [10]. In knowledge representation and reasoning, topological structures of knowledge bases are presented as knowledge graphs [11]. In construction, a variety of data can also be represented by graphs. For example, a floor plan can be abstracted as a graph where rooms are connected with walls [12]. A Gantt chart forms a graph of construction activities connected by their dependencies [13]. In recent years, graph neural networks (GNNs) have received considerable scholarly attention due to their convincing performance in processing graphs [8]. GNNs have demonstrated their potential to solve complex problems in many fields. For example, in medicine, GNNs are used to predict the side effects caused by combinations of multiple drugs [14]. Researchers in chemistry adopt GNNs to predict chemical properties based on molecule structures [15].

Graph neural networks (GNNs) have gained increasing attention in the field of construction, with their initial use being the clustering of building groups into regular and irregular patterns through unsupervised learning [16]. Since then, GNNs have been applied to a range of construction-related tasks, including floor plan design [17], bridge inspection [18], and BIM semantic enrichment [12], among others. The number of studies investigating the adoption of GNNs is on the rise [19].

* Corresponding authors.
E-mail addresses: y.jia@research.deakin.edu.au (Y. Jia), jun.wang@westernsydney.edu.au (J. Wang), w.shou@westernsydney.edu.au (W. Shou), reza.hosseini@deakin.edu.au (M.R. Hosseini), Yu.Bai@monash.edu (Y. Bai).

https://doi.org/10.1016/j.autcon.2023.104984
Received 12 January 2023; Received in revised form 5 June 2023; Accepted 11 June 2023
Available online 21 June 2023

Despite the desirability of such attention, understanding literature in the field highlights particular challenges. Indeed, the volume of work now available makes it difficult to evaluate the exact nature of the knowledge uncovered, its impact and contribution, and specifically, to identify critical areas that remain overlooked or neglected. In essence, where a field sees a burst of research activity, a rigorous, critical review of the body of output now available is warranted [20]. To date, this is lacking. Researchers in the field of computer science have conducted several surveys or reviews on GNNs [8,9,12,21–23]. These surveys and reviews suggest that GNNs can be a powerful tool to process graph-structured data, and their applications beyond the field of computer science are rapidly developing. However, these reviews primarily focused on computer science [24] and computer vision [25,26], with only a few covering other areas such as drug design [27,28], power systems [29], and traffic forecasting [10], which are not applicable to the construction field due to this domain's unique characteristics and challenges.

This review paper aims to comprehensively examine the applications of graph neural networks (GNNs) in the construction industry. The goal is to provide a critical analysis of the current state of GNNs in construction, raising awareness of their adoption in this field and serving as a reliable reference for designing GNN models for construction activities. The review will also explore the challenges and opportunities of using GNNs in construction, laying the groundwork for future research in this area. The paper is structured as follows: Section 2 introduces the basic concepts of graphs and GNNs. Section 3 describes the review methods and an overview of the retrieved literature. Section 4 analyzes the methods for constructing graphs from seven data types in construction. In Section 5, various GNN-enabled applications in construction are presented. Finally, Section 6 discusses the findings from a broad perspective.

2. Graph and graph neural networks

2.1. Graph

A graph G is a type of non-Euclidean structure consisting of a set of nodes (or vertices) V and a set of edges E. A graph can be denoted as $G = (V, E)$, and an edge can be denoted by its nodes as $e_{ij} = (v_i, v_j) \in E$, where $v_i, v_j \in V$. A graph can also be represented as a $n \times n$ matrix called adjacency matrix A for computational purposes. If $e_{ij} \in E$, then $A_{ij} = 1$; and if $e_{ij} \notin E$, then $A_{ij} = 0$. Each node, edge or the whole graph may have attributes that can be represented as vectors to contain more information.

A graph can be categorized as follows:

- a directed or an undirected graph, depending on if the edges are directed or undirected from a node to another node; or
- a homogeneous or a heterogeneous graph, depending on if all the nodes or edges represent the same class of information; or
- a static or dynamic graph, depending on if the attributes or the topologic structure will change with time.

Unlike the data in uniform structures, graphs can be different in size. In addition, the number of nodes and edges is not permanently fixed. A graph is also invariant to node ordering, which means that the graph will not be changed even if the order of nodes is changed. Because of the irregular form of graphs, it is difficult for traditional neural networks to process graphs: multilayer perceptrons (MLPs) connect all data points together, ignoring relationships between data points; CNNs expect data with a fixed size so that the convolutional kernel can operate on subpatches; RNNs require a fixed sequence of data as inputs. Graphs cannot meet the requirements of these traditional neural networks. To tackle this issue, researchers in the field of computer science proposed GNNs [30].

2.2. Graph neural networks

The basic idea of GNNs is that the representation of each node in a graph is determined by its own attributes and the aggregation of its neighboring nodes via edges. As a result, GNNs have better performance in processing relationships among nodes [31], while traditional deep learning approaches (CNNs and RNNs) assume that nodes are independent of each other and ignore the topological information [9]. There are three types of tasks that GNNs can perform:

- Node-level tasks: node classification, node clustering and node regression
- Edge-level tasks: edge classification and link prediction
- Graph-level tasks: graph classification, graph clustering, and graph regression

Based on the propagation modules, GNNs can be classified into three categories: recurrent GNNs (Rec-GNNs), convolutional GNNs (Conv-GNNs) and Skip-GNNs [8]. The early studies on GNNs started from Rec-GNNs. Based on recursive neural networks, Gori et al. [30] formally proposed the concept of GNN for the first time, and Scarselli et al. [32] further developed the concept. Then, inspired by the development of CNNs, Conv-GNNs were proposed, which generalizes convolution operators to GNNs [9]. The main difference between Rec-GNNs and Conv-GNNs is the weights among layers: the former uses the same weights across layers during propagation, while the latter uses different weights [28]. Based on Rec-GNNs and Conv-GNNs, Skip-GNNs add skip connections across layers to avoid over-smoothing problems and increased noise in deeper networks [8].

2.3. Message passing mechanism of GNNs

A basic GNN model operates as a message passing mechanism consisting of aggregation, update, and iteration. The message passing mechanism can be expressed as Eq. (1):

$$h_u^k = Update^{(k-1)}\left(h_u^{(k-1)}, Aggregate^{(k-1)}\left(h_v^{(k-1)} \mid \forall v \in N(u)\right)\right) \quad (1)$$

where $h_v^{(k-1)}$ are the embeddings of the node u's neighbors $N(u)$ at the *(k-1)*-th iteration, and $h_u^{(k-1)}$ are the embedding of node u itself at the *(k-1)*-th iteration. Throughout the two functions $Aggregate^{(k-1)}$ and $Update^{(k-1)}$, the embedding of node u at the *(k)*-th iteration is obtained. The *Aggregate* and *Update* functions are neural networks such as multilayer perceptron (MLP). A GNN with k iterations is also known as a k-layer GNN.

Take the graph shown in Fig. 1, for example. The target node is A, its direct neighbors are nodes B and C, and its two-hop neighbors are nodes E and D. If a single-layer GNN is applied to the graph ($k = 1$), only the information of nodes B and C is passed to node A. In the aggregation operation, the original embeddings of its neighbor node B $h_B^{(0)}$ and node C $h_C^{(0)}$ are aggregated. Next, in the update operation, the embedding of node A at the 1st iteration $h_A^{(1)}$ is updated based on its own original embedding and the aggregated embeddings of its neighbor nodes B and C. At this point, the message passing is completed. If there is a two-layer GNN applied to the graph ($k = 2$), the information of the two-hop neighbors can be passed to the target node. Specifically, the embeddings $h_B^{(1)}$ and $h_C^{(1)}$ are firstly obtained from $h_A^{(0)}$, $h_D^{(0)}$ and $h_E^{(0)}$, and then the embedding $h_A^{(2)}$ is obtained from $h_B^{(1)}$ and $h_C^{(1)}$. Therefore, the embedding $h_A^{(2)}$ contains the $h_A^{(0)}$, $h_D^{(0)}$ and $h_E^{(0)}$ through aggregating the $h_B^{(1)}$ and $h_C^{(1)}$. In general, the more layers a GNN has, the more iterations will have in one message passing process, and the information of farther nodes will be passed to the target node. A node's embedding can be learned from its k-hop neighbors after k iterations with a k-layer GNN. For example, Variants of GNNs can be made by changing the *Aggregate*

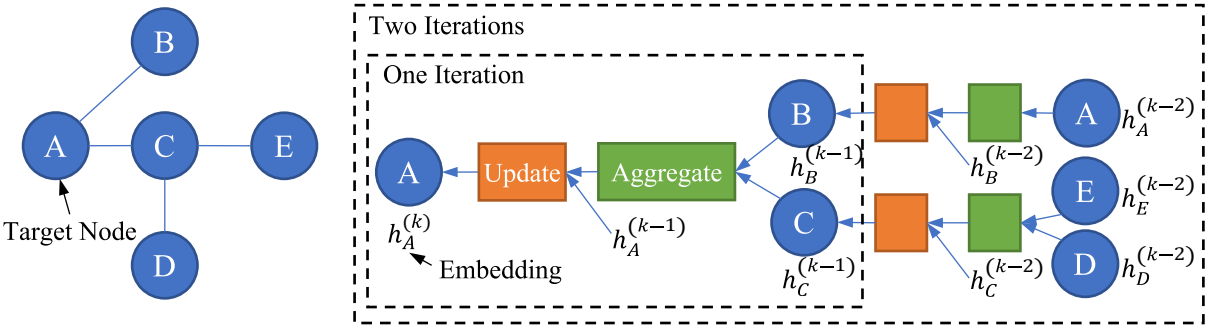


Fig. 1. Message Passing of GNN.

and *Update* functions and even adding sampling and pooling modules, such as graph convolutional networks (GCNs) [25], graph attention networks (GATs) [24], GraphSAGE [33], etc.

2.4. Pipeline for designing a GNN model

Zhou et al. [8] summarised a four-step pipeline of designing a GNN model as shown in Fig. 2, including i) finding graph structure, ii) specifying graph type and scale, iii) designing loss function, and iv) building model using computational modules.

First, a graph structure should be linked to the application filed. Explicit graph structures are easier to find, but some latent data structures can also be formatted as graphs. For example, the interrelation of words in natural languages can be mapped into a graph [34]. After obtaining a graph structure from the application field, it is necessary to identify its graph type, define its adjacency matrix, and code attributes as vectors to its nodes, edges or the entire graph. The task type and learning setting should be determined in the third step. On this basis, a proper loss function can be selected. There is no difference in loss functions between training GNNs and other deep neural networks. For example, the mean square error loss is an option for node regression tasks under supervised learning. The fourth step is to build the proposed GNN model by using GNN modules, including propagation modules, sampling modules, and pooling modules. The first and second steps of this pipeline will be addressed through this review in the context of construction in Section 0 and illustrate the third and fourth steps with existing applications of GNNs in construction in Section 5.

3. Research methods

A systematic review following the PRISMA framework [35] was conducted to obtain a comprehensive understanding of the status quo of GNNs in the construction context. Fig. 3 illustrates the PRISMA flow. Data were mainly retrieved from Scopus because of its extensive interdisciplinary repository of construction management and information technology [36]. Google Scholar was also employed as a supplementary search engine to ensure thorough coverage.

The full search string, which consists of three parts, is detailed in Appendix A. The first part narrows the search to focus on the term "graph", while the second part specifically targets AI-related content. The third part restricts the search to a comprehensive list of construction-related journals, drawing from previous interdisciplinary studies on construction and IT [37,38]. No time-frame limitation was imposed, with the date range set as all years to present. In addition, there was no constraint on the type of publication, thus minimizing potential "publication bias", as argued by Hopewell et al. [39].

The search was conducted in May 2022, yielding a total of 654 articles from Scopus and Google Scholar after removing duplicates. A two-stage manual screening was employed to identify relevant articles. In the first stage, titles and abstracts were assessed to exclude unrelated articles. If an article's relevance could not be determined from its title and abstract alone, the research method section was evaluated during the second stage to ascertain whether it adopted the GNN model to address the research question.

To further ensure the quality and relevance of the selected articles, the following inclusion and exclusion criteria were applied.

Inclusion criteria:

1) Articles that focused on the application of GNNs in the construction industry.
2) Articles that provided a clear description of the GNN model and its implementation in addressing the research question.
3) Research-based articles from various sources, including peer-reviewed journals, conference proceedings, preprint repositories (e.g., arXiv), and other relevant publications.

Exclusion criteria:

1) Articles that mentioned GNNs but did not directly apply them in the construction context.

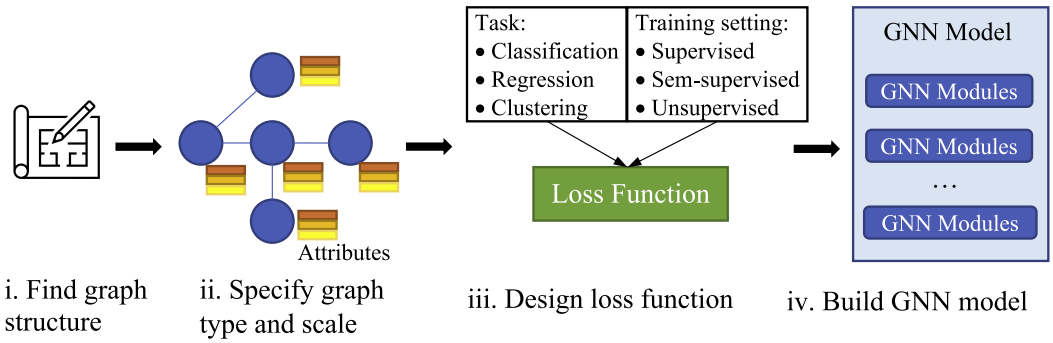


Fig. 2. Pipelines for Designing GNN Model (adapted from Zhou et al. [8] under CC BY 4.0).

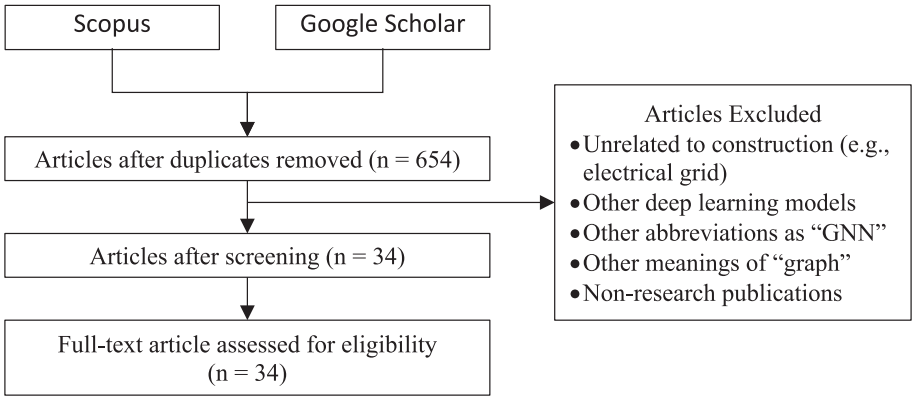


Fig. 3. PRISMA Flow.

2) Articles with insufficient information on the GNN model or its implementation.
3) Non-research publications, such as opinion pieces, editorials, or reviews without original research.

In summary, the rigorous search and screening process, along with the application of inclusion and exclusion criteria, ensured the selection of 34 relevant and high-quality articles for this systematic review. By synthesizing and analyzing the information derived from these articles, this review offers valuable insights into the current applications of GNNs in the construction industry and provides a solid foundation for future research in this area.

4. Findings of the study

4.1. Wave of research on GNNs

Fig. 4 illustrates the distribution of retrieved articles. The adoption of GNNs in the field of construction emerged in 2018 and is still in its infancy, but it was proliferating in 2021. Journal articles accounted for 61% (16 out of 26), while conference papers and preprints contributed 31% (8 out of 26) and 8% (2 out of 26), respectively. The top five sources are *Automation in Construction, ACM Transactions on Graphics, Proceedings of the 38th International Symposium on Automation and Robotics in Construction (ISARC), Proceedings of the 2021 European Conference on Computing in Construction, and arXiv*. It is important to note that the retrieval of articles for this review was conducted up to May 2022, so the number of publications for 2022 may not be complete, and the trend could potentially continue to grow.

Table 1 presents the distribution of the 34 articles we analyzed, each corresponding to one of four stages in the construction lifecycle. The design stage accounts for the majority of the research, with 12 publications, and is closely followed by the operation stage, with 11 publications. This distribution suggests a significant interest in the application of GNNs during the design and operation stages of

Table 1
Distribution of Articles over Construction Lifecycle.

Stage	No. of Articles	Reference
Planning	6	[16,40–44]
Design	12	[12,45–55]
Construction	5	[13,56–59]
Operation	11	[1,18,31,60–67]

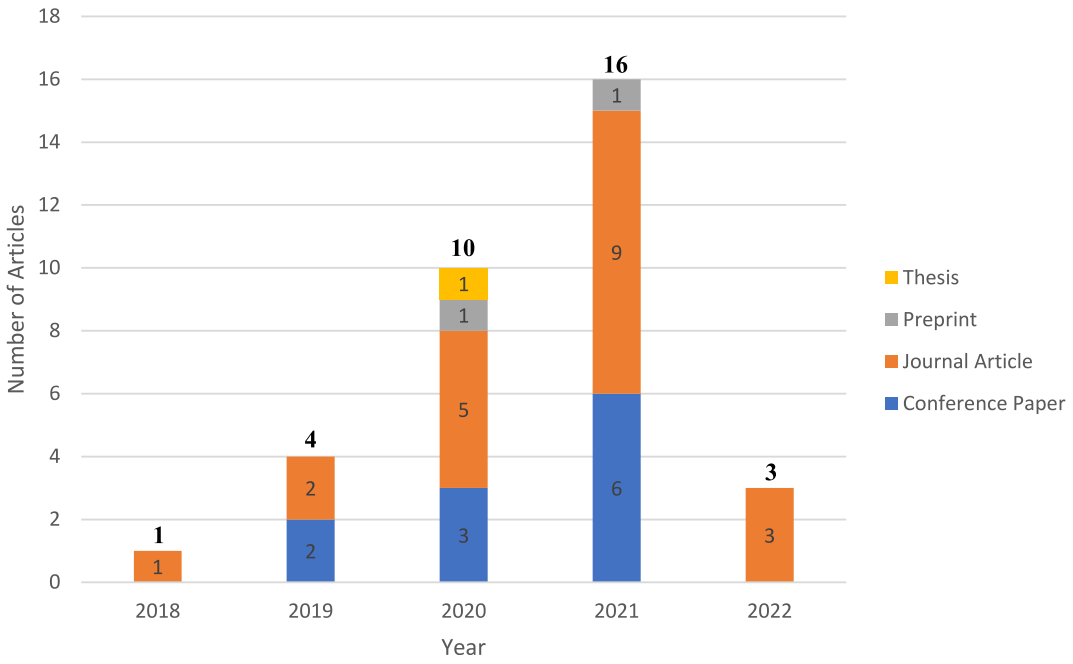


Fig. 4. Distribution of Articles over Years.

construction. In the design stage, GNNs are primarily used for tasks such as floor plan design and evaluation. During the operation stage, GNNs are commonly employed for defect detection or building component identification.

In contrast, fewer articles concentrate on the planning and construction stages, suggesting these areas as potential avenues for future research. Within the planning stage, the primary application of GNNs is for the analysis of building group patterns. During the construction stage, GNNs are primarily applied to assist in creating as-built BIM models. The insights gained from this distribution of research can help guide future studies to explore underrepresented applications of GNNs in the construction lifecycle.

4.2. Developing graphs in construction

Construction is a complex field, generating a wide variety of data types. For instance, the geometric and semantic information of a building can be stored in a building information modeling (BIM) model, while the geometric information of an existing structure can be captured as point cloud data. To apply GNNs in construction, it is essential to first transform the various types of data into graph structures. This process involves mapping the data into nodes and edges, which can then be used as inputs for GNNs. The following sections will provide an overview of the approaches used in existing literature for developing graphs from seven types of data, including industry foundation classes (IFC), point cloud, BIM models, triangle meshes, Delaunay triangulation, vectorized shapes and bubble diagrams. The framework for developing graphs from these data and their applications in construction is illustrated in Fig. 5.

4.3. IFC to graph

The IFC is a widely adopted, standardized digital schema for describing architectural, engineering, and construction information. It utilizes Express-G graphical notation [68] to describe BIM objects and

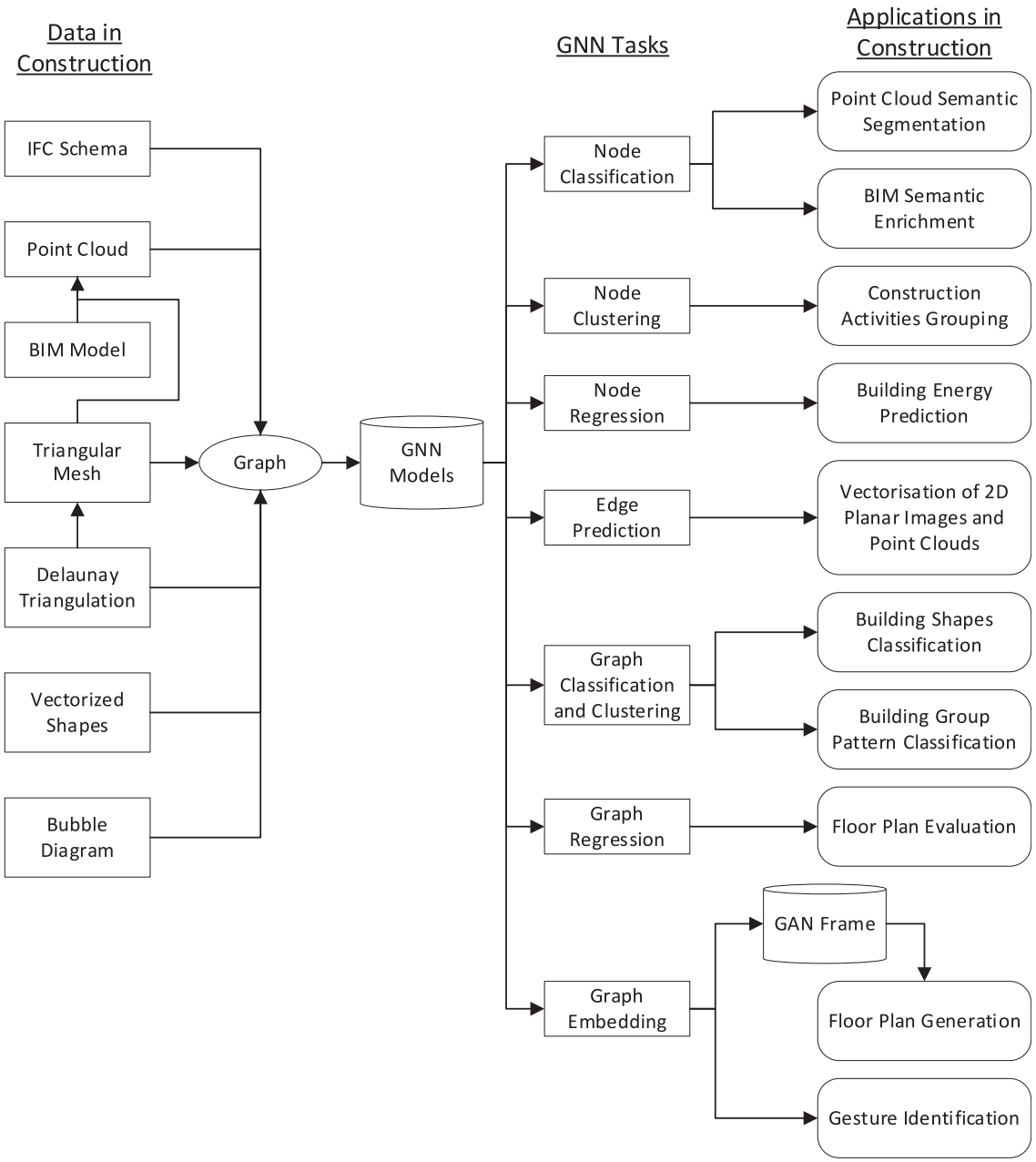


Fig. 5. Framework for Developing Graphs and GNN Applications in Construction.

their relationships. The IFC schema comprises IFC entities arranged in a hierarchical structure, with each entity having predefined attributes specified by the IFC specification. This hierarchical structure makes it possible to convert IFC models into graph structures, which can be used to apply GNNs in construction. The review of existing literature shows three main approaches to developing graphs from the IFC schema, which will be discussed in the following sections.

4.3.1. Approach #1: IFC hierarchy to graph

Tauscher et al. [69] proposed an approach to develop directed heterogeneous graphs from the IFC schema. In their approach, two types of IFC entities are regarded as nodes, i.e., IFC objects inherit from IfcObjectDefinition, and objectified relations derived from IfcRelationship including IfcRelAssigns, IfcRelDecomposes, IfcRelAssociates, IfcRelDefines and IfcRelConnects. Edges are directed and are constructed from two sources: i) the unidirectional relation derived from IFC objects' attributes linking one object to another, and ii) the bidirectional relation between two IFC objects linked by objectified relations.

An example illustrates this approach in Fig. 6. IfcWindow, IfcMaterialConstituentSet and IfcMaterialConstituent are IFC objects that inherit from IfcObjectDefinition, and therefore are regarded as nodes. Because IfcMaterialConstituent is an attribute of IfcMaterialConstituentSet, a unidirectional edge is added between them. Since IfcRelAssociatesMaterial is a type of objectified relation, it is also regarded as a node whose edges are bidirectional.

4.3.2. Approach #2: IFC hierarchy to graph (IfcRel as edges)

Similar to the first approach, Ismail et al. [70] proposed an approach to convert the IFC schema into directed heterogeneous graphs. In their approach, nodes are IFC objects that inherit from IfcObjectDefinition. Edges are defined in three ways: (1) inheritance relations of IFC objects; (2) the relation defined in object attributes; (3) the relation extracted from IfcRelationship. These edge relations are also used as edge labels. All edges are unidirectional. This approach has been integrated – with other IFC tools – into a cloud-based BIM server called IFCWebServer [71]. Fig. 7 shows an example of this approach. Compared to Approach #1, Approach #2 takes IfcRelationship as edges, attaches edge labels and only has unidirectional edges.

4.3.3. Approach #3: IFC plus 3D geometry to graph

The above two IFC-to-graph approaches explicitly develop graphs from the hierarchical structure of the IFC schema. However, semantic information about BIM objects and 3D spatial relationships among BIM objects is not considered.

The third IFC-to-graph approach proposed by Khalili and Chua [72] takes advantage of both the hierarchical structure of the IFC schema and the geometric and semantic information of BIM objects. Fig. 8 illustrates an example of this approach. Nodes represent 3D BIM objects such as building elements, spaces, and zones; edges represent the topological and spatial relationships between related 3D objects. The IfcRelation

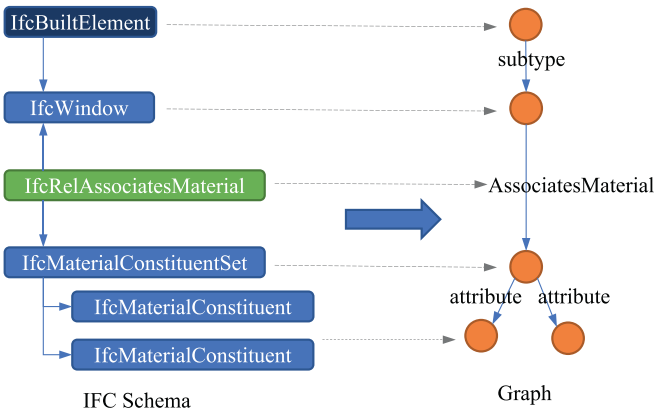


Fig. 7. An Example of the Process of Constructing Graphs from IFC Schema (Approach #2).

entity in the IFC schema determines the topological relationships. In addition, spatial relationships between BIM objects, including containment, connectivity, separation, and intersection, can be identified by analyzing the Representation and ObjectPlacement attributes of the IFC objects inherited from IfcProduct. This approach also enriches the semantic information of the graph with information extracted from the BIM model. Nevertheless, the analysis of the 3D representation of BIM objects needs conversions between SweptSolid and B-rep, which increases the complexity of this approach.

The three IFC-to-graph approaches have been used to generate graph representations of the IFC schema for analysis with graph theory. Furthermore, graph queries can be applied to the IFC-generated graphs to analyze BIM information, such as retrieving material information and finding the shortest route, as argued by Ismail et al. [70]. However, no studies in the retrieved literature have applied GNNs to these IFC-to-graph approaches.

4.4. Point cloud to graph

Point cloud data can present the geometry of objects as a set of points with attributes such as coordinates (XYZ), color (RGB), and reflection intensity [63]. Point cloud data can be generated from real-world objects with scanning devices (e.g., LiDAR or TLS) [62,63] or synthesized from digital objects (e.g., BIM models or CAD drawings) [58,59].

To construct a graph from a set of points, a common approach is to connect the points using edges. This process is demonstrated in Fig. 9, where the nearest neighbor search (NNS) method is employed. Two different NNS algorithms, the k-nearest neighbor (k−NN) algorithm [47,73] and the fixed-radius near neighbor (Fr-NN) algorithm [63], were used in the retrieved literature. Both of these algorithms develop a graph starting from a central point. The k-NN algorithm connects the k closest points to the central point, while the Fr-NN algorithm connects

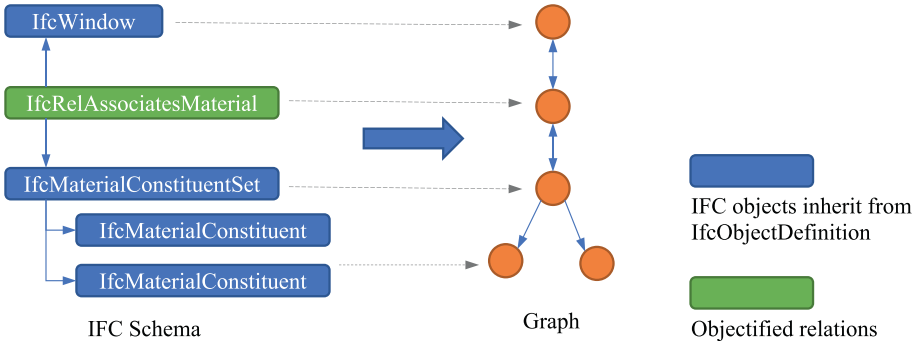


Fig. 6. An Example of the Process of Developing Graphs from IFC Schema (Approach #1).

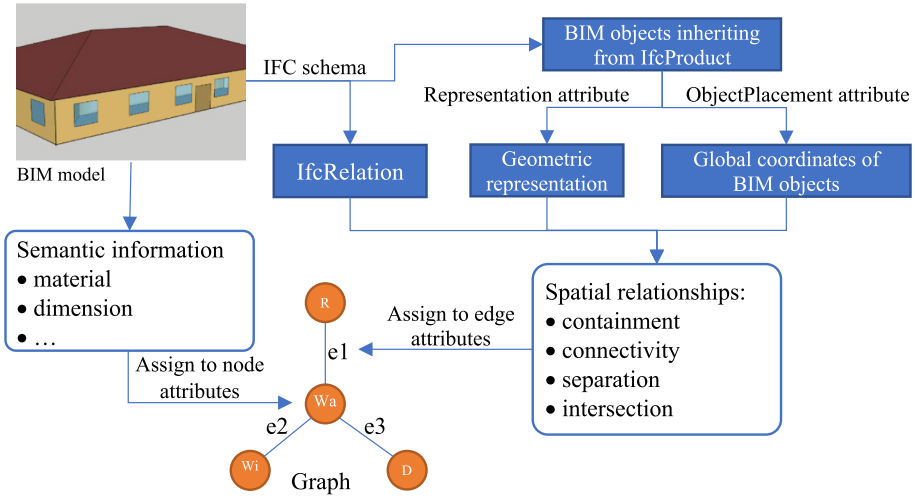


Fig. 8. An Example of the Process of Constructing Graphs from IFC Schema (Approach #3).

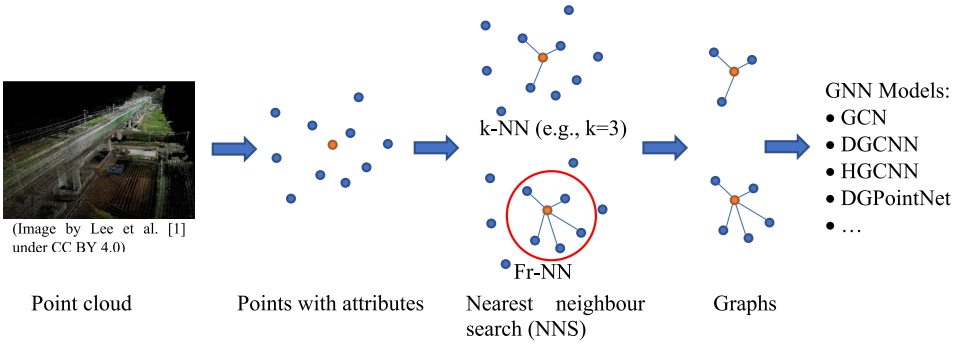


Fig. 9. An Example of the Process of Constructing Graphs from Point Clouds.

all points within a given radius. The value of k in the k-NN algorithm or the radius in the Fr-NN algorithm is a hyperparameter, which determines the constructed graph's size. Choosing a small value of k can amplify the effect of noise in the point cloud, while a larger value of k can lead to high computational costs [17]. There is no fixed method for determining the optimal value of k, but the Elbow method is a popular approach [74]. This method involves calculating the model's performance for different values of k and selecting the value that results in the best performance. If the process is illustrated in a line diagram, the line will resemble an arm, and the "elbow" point corresponds to the optimal value of k. Additionally, domain knowledge can also be used to determine suitable values of k; for example, in the ModelNet40 dataset, a computer vision benchmark dataset containing CAD models from 40 object categories, Wang et al. [73] found 20 to be the optimal value of k.

4.5. BIM model to point cloud to graphs

While GNNs have shown promise in analyzing point cloud data, obtaining high-quality data for training GNN models can be challenging. In practice, limitations in scanning equipment and space constraints can make it difficult to collect enough data [75]. Additionally, manually labeling each point in the point cloud can be a tedious and time-consuming task. To overcome these challenges, Ma et al. [59] proposed a workflow that uses commercial software to transform BIM models into synthetic point clouds. The process is illustrated in Fig. 10. In this workflow, the point cloud is generated from BIM objects, and data labeling is performed at the object level instead of the point level. The point cloud is labeled according to the semantic information stored in the original BIM model, which reduces the workload of capturing and labeling data in the real world. Finally, the synthetic data is mixed with real-scanned data. The authors tested the semantic segmentation performance of the mixed data with different mixing ratios, and the results

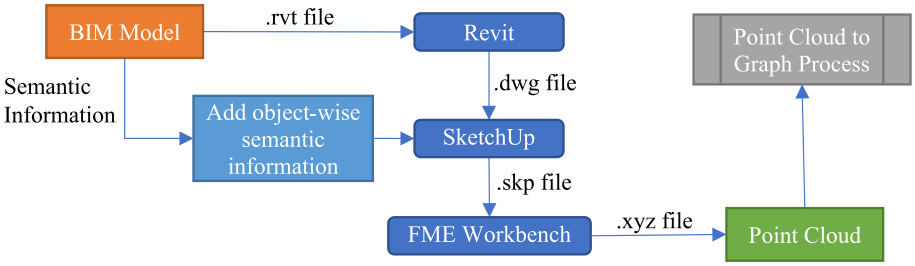


Fig. 10. An Example of the Process of Constructing Point Cloud from BIM Models.

showed that the mixed data could be used as a supplement when real data is insufficient, although its accuracy is lower than that of real data.

4.6. Triangle meshes to graphs

Triangular meshes are a popular form of representation for 3D objects [76], as they use a set of triangles to approximate the surface of an object. Each triangle in a triangular mesh has three vertices and three edges, and each pair of adjacent triangles shares two vertices and one edge. Triangular meshes can be exported from BIM models to obj files using the Revit API [58].

Two methods have been proposed to convert triangular meshes into graphs, as shown in Fig. 11. The first approach, mesh-to-graph, directly uses the vertices of the mesh as graph nodes and the edges of the triangles as graph edges [47]. The second strategy, termed mesh-to-point-to-graph, entails the random sampling of points within each triangle of the mesh to generate a point cloud. [47,58]. Once the point cloud is obtained, NNS methods can be applied to generate graphs.

4.7. Delaunay triangulation (DT) to graphs

A Delaunay triangulation (DT) is a type of triangulation used in computational geometry. Given a set of points on a plane, it connects those points using triangles in such a way that the circumcircle of any triangle formed by three points in the set does not contain any other point from the set [77]. Based on the DT, the minimum spanning tree (MST), Gabriel graph and Voronoi diagram can be generated [78]. In the field of GIS, DT is commonly used to generate a set of triangles from a set of sample points for terrain modeling [40]. The MST can be generated from a DT using the Bowyer-Watson algorithm [79].

Fig. 12 illustrates the process of converting DT into a graph. As DT can be considered a special form of triangle mesh, the methods discussed in Section 4.4 can also be applied to it. Both DT and MST have a topological structure that is similar to that of a graph, with the vertices of DT and MST being the nodes of the graph and the edges of DT and MST being the edges of the graph [16,42], which is the mesh-to-graph approach. Another approach, the mesh-to-point-to-graph approach, which involves sampling points randomly within each triangle of the mesh to form a point cloud, is not reported in the literature for DT.

4.8. Vectorized shapes to graphs

Vectorized shapes, such as lines, polylines, polygons, and primitives, are widely used as geometric representations in CAD systems. In GIS, vectorized shapes are commonly used to represent the outlines of buildings in map space [41]. Unlike raster shapes, which have a grid-like structure and can be easily processed by CNNs, vectorized shapes have an irregular structure and are closer to graphs, making them not suitable as inputs for CNNs [40]. In order to use vectorized shapes as inputs for GNNs, a conversion to a graph structure is necessary.

A workflow to extract graphs from vectorized CAD primitives was proposed by Simonsen et al. [46] and is illustrated in Fig. 13. The vertices of polylines and polygons are represented as nodes in the graph, while the lines between these vertices are represented as edges. Curves are divided into several line segments at a given angle and then converted to nodes and edges in the same way as polylines and polygons. The attributes of a node include geometric relationships with neighboring nodes, such as node degrees, the angle between lines, and the length of lines. This workflow can explicitly capture the geometric information of floor plans in CAD.

However, it is worth noting that this method may lose some of the inherent semantic information in CAD primitives. Additionally, high computational costs may arise from the many line segmentations used for curves. To tackle this problem, the Douglas-Peucker algorithm [80] can be used to approximate a curve with fewer line segmentations. In the absence of vectorized floorplans, the Raster-to-Vector algorithm [81] can be used to transform rasterized floorplan images into vectorized floorplans.

4.9. Bubble diagrams to graphs

A bubble diagram in architecture is a kind of sketch that consists of bubbles connected by lines. Bubbles represent spaces, and lines represent the adjacency of these spaces. Bubble diagrams are often used at the preliminary stage of building design for space planning. They are, by nature, graphs: bubbles are the nodes, and lines are the edges, as represented in Fig. 14. To enhance the semantic information of a bubble diagram and use it as an input for GNNs, attributes can be added to the nodes (e.g., room types, perimeter, area, volume, etc.) and edges (e.g., doors, walls, stairs, openings, etc.). For existing buildings, bubble diagrams can be extracted from floor plans to analyze the spatial design of buildings [12,53]. As et al. [55] developed a Python script to automatically extract a bubble diagram and convert it into a graph format via Revit API.

5. Applications of GNN in construction

This section will examine the various ways GNNs are used in construction, as identified in the literature, providing an in-depth analysis of these topics and their potential benefits to the field.

5.1. Point cloud semantic segmentation

Advanced scanning devices, such as LiDAR and TLS, can provide reliable and accurate point cloud data even in an adverse environment [63]. After obtaining a point cloud, semantic segmentation is typically conducted to classify each point into predefined categories. However, the high number of points in a point cloud makes manual segmentation time-consuming and prone to errors.

Deep learning approaches have been proposed to automate point

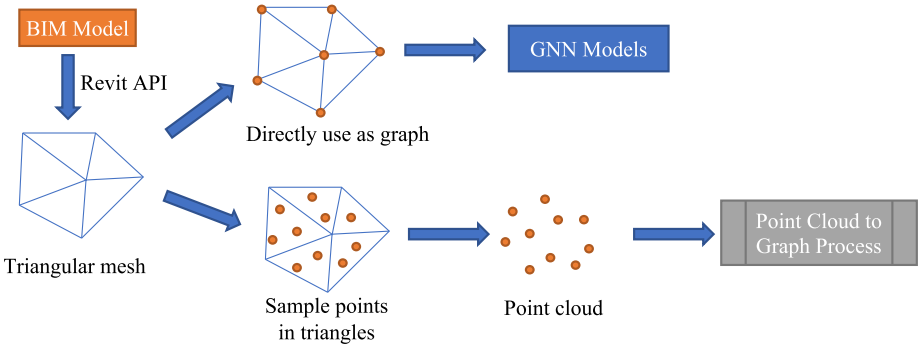


Fig. 11. An Example of the Process of Constructing Graph from Triangular Mesh.

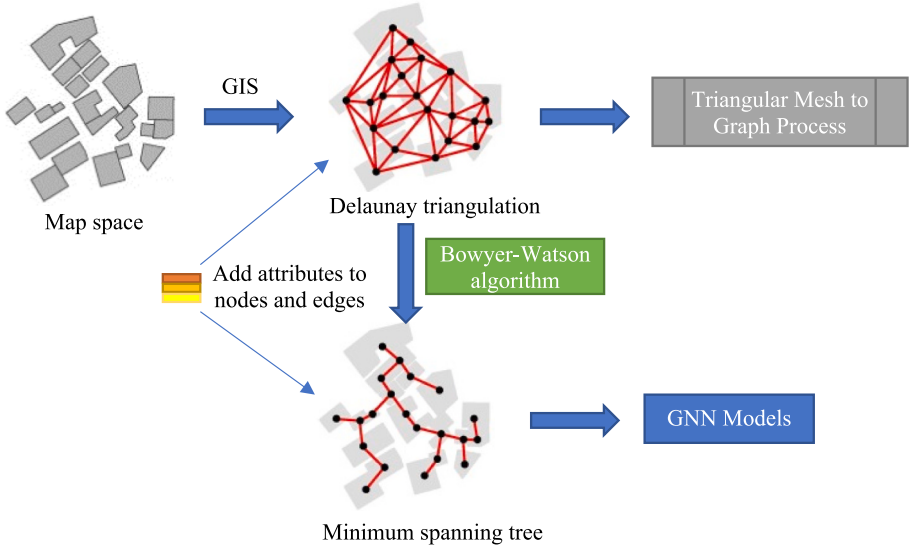


Fig. 12. An Example of the Process of Constructing Graph from Delaunay Triangulation (adapted from Yan et al. [40] under CC BY 4.0).

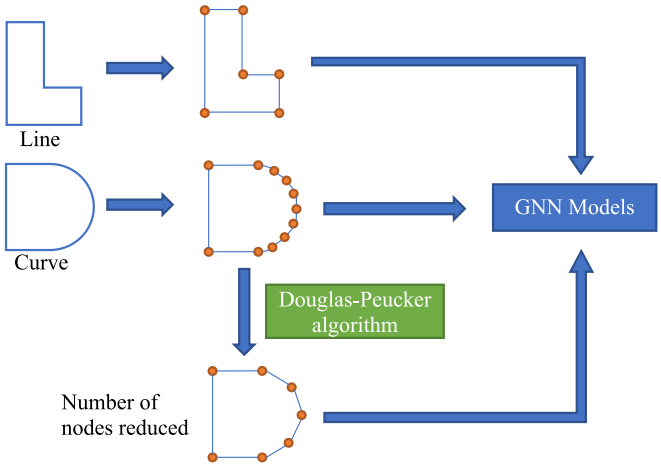


Fig. 13. Process of Constructing Graph from Vectorized Shapes.

cloud semantic segmentation. The first such approach is PointNet [82], which uses CNNs to extract both the features of each point and a global feature of the cloud. In PointNet, the feature of each point is first extracted by a multilayer perceptron (MLP). Then, a max-pooling layer is applied to extract the largest value across each channel to form the global feature of the point cloud. Finally, the point feature and the global feature are concatenated and processed by two MLPs to complete semantic segmentation. Although PointNet has achieved promising results as a pioneer in semantic segmentation, it only extracts the feature of each point and the global feature, but not local features. PointNet++ [83] addresses this issue by adding sampling layers and grouping layers to capture local features. The sampling layer uses the farthest point sampling method to select points from a point cloud, and the radius-based ball query is adopted in the grouping layer to construct local point sets around each sampled point. PointNet is then applied to the local point sets to extract the local features. Other popular algorithms for point cloud semantic segmentation include PointCNN [84] and PCNN [85].

To extract relationships between points, a GNN-based model, the Graph-based Dynamic Graph Convolutional Network (DGCNN) [73], has been proposed. DGCNN constructs a local neighborhood graph using the k-NN algorithm. The edge features are learned from the features of linked nodes, representing the relationship between a point and its neighbors. The convolution operator EdgeConv iteratively performs convolution on edges to learn local features. The constructed k-NN graphs are dynamic, which are recomputed after each EdgeConv layer in the updated feature space. The global feature is constructed by concatenating local features. Before the final output layer, the local features are concatenated again with the global feature to enhance the representation. This approach effectively captures the relationships between points in the point cloud, which is critical in the point cloud semantic segmentation task.

DGCNN has been found to achieve superior performance in point cloud semantic segmentation compared to PointNet and PointNet++, as evidenced by its higher mean intersection over union (mIoU) values on the ShapeNet part dataset (86.1%) in comparison to PointNet (83.7%) and PointNet++ (85.1%) [73]. Subsequently, various versions of DGCNN, such as HGCNN [1] and DGPointNet [62], have been proposed.

In the field of construction, GNN-enabled models have been used to perform point cloud semantic segmentation for surface inspection and structure component classification. One such example is the work of Bahreini and Hammad [64], who used a DGCNN model to identify

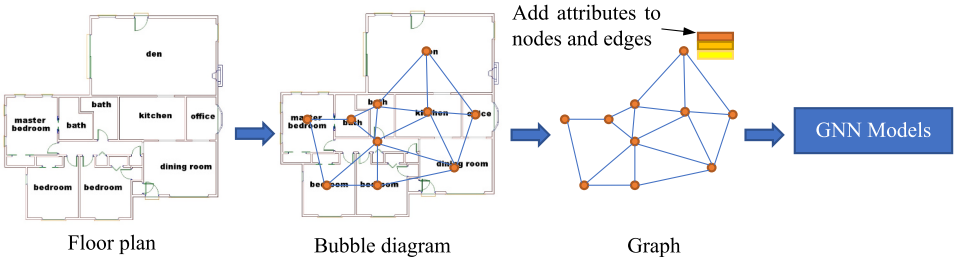


Fig. 14. An Example of the process of constructing graph from bubble diagram. "Floor plan"by *It's Great To Be Home* is licensed under CC BY-NC 2.0.

cracks and spalling on concrete surfaces from scanned 3D point cloud data of bridges. The point cloud data were manually labeled with three types of labels: crack, spalling, and non-defect. Each point was represented as a 7-dimensional vector, including its XYZ coordinates, RGB channels, and a normalized Y coordinate. The model achieved a recall of 55.20% for cracks and 89.77% for spalls. However, the supervised training in this approach requires manual labeling of thousands of points, which can be time-consuming and prone to error. To address this issue, Feng et al. [63] proposed a semi-supervised approach using a GCN-based model to detect pavement cracks from point cloud data. To improve performance, feature engineering was applied to distinguish the features of different categories. They used space mapping to amplify the difference in reflection intensity between cracks and non-crack areas and also designed four local features to enhance the representation of points. As a result, the recall of the GCN-based model was 10.8% higher than PointNet and 6.9% higher than DGCNN. Additionally, Kim and Kim [18] compared the performance of PointNet, PointCNN, and DGCNN in classifying bridge components from point clouds and found that DGCNN had the highest overall accuracy (OA) and mIoU.

In some studies, the original DGCNN model has been modified to improve its performance for classifying buildings. For instance, Lee et al. [1] proposed a DGCNN-based model called Hierarchical DGCNN (HGCNN), which shortens the architecture of DGCNN and increases the k-value of the k-NN graph. This model achieved equivalent levels of overall accuracy (OA) and IoU as PointNet and DGCNN while providing improved performance for taller components such as electric poles. Another study by Yajima et al. [62] introduced the DGPointNet model for classifying damaged building components in disaster-affected environments. This model concatenates local and global features extracted from both PointNet and DGCNN. The results of this study showed that DGPointNet outperforms PointNet and PointNet++ in terms of mIoU and OA while maintaining near real-time computation time.

The above studies demonstrate the effectiveness of GNN-based models for the semantic segmentation of point clouds in construction. However, it's worth noting that the accuracy of these models in classifying different building components or components in different locations can be inconsistent. For example, the accuracy of classifying abutments and piers using DGCNN is relatively lower than that of other bridge components, as reported in [18]. Similarly, DGCNN's cube-shaped blocks in semantic segmentation may lead to points of walls at corners being misclassified as columns, as reported in [57]. These issues should be considered when using GNN-based models for the semantic segmentation of point clouds in construction. One potential solution to this problem is to incorporate rule-based algorithms as shape or spatial constraints to correct misclassified points, as this can improve the model's accuracy.

5.2. BIM semantic enrichment for object classification

BIM semantic enrichment is a process that infers new semantic information about BIM objects or relationships from a BIM model [86]. This process can address the loss of semantic information that can occur during data exchange between different BIM tools. Additionally, it can be useful in the context of scan-to-BIM, where it can be used to assign correct attributes to as-built BIM models. One specific task that is often performed as part of BIM semantic enrichment is the classification of BIM objects [87].

Traditional approaches to BIM object classification are often rule-based. For example, Sacks et al. [88] developed a BIM semantic enrichment tool called SeeBIM that utilizes inference rule sets to match the geometric features of a BIM object and the topological relationships between a pair of objects. These inference rule sets are composed of unique rule strings, which can describe the unique relationships of a pair of objects and allow them to be identified and distinguished from other objects. The rule strings are extracted from the knowledge of domain experts and translated into computer code. Using this approach, SeeBIM achieved 100% accuracy for a bridge model. However, creating the inference rule sets can be challenging, as they must be carefully compiled to avoid infinite loops or rule conflicts. Another example is the method proposed by Ma et al. [89], which calculates the similarity of geometric and topological features of a BIM object with rules of a predefined knowledge base and outputs possibilities for each classification. The maximum possible classification can be regarded as the result. This method also achieved 100% accuracy for a bridge model in a case study.

While rule-based approaches to BIM object classification have demonstrated accurate results, they can be challenging to implement and maintain. Defining the rules requires domain expertise, and coding them into computer code can be complex. Furthermore, rule-based approaches tend to have a very narrow scope of application [12], as the rules must be specifically compiled for each scenario and lack generalizability. To address these issues, some researchers have proposed GNN-enabled approaches to BIM object classification that do not have the same limitations.

In an effort to address the limitations of traditional rule-based approaches, Collins et al. [47] proposed a GCN-enabled approach to classify 3D objects into corresponding IFC entities based on their geometric features. In their study, the 3D objects were represented as triangular meshes, and two graph construction approaches were used: "triangle meshes to graphs" and "Delaunay triangulation to graphs". They compared the performance of these two approaches and found that the mesh-to-point approach had higher overall accuracy and better generalization capability because randomness is introduced when sampling points. Additionally, they found that when used at a higher level of development (LOD), the mesh-to-point-to-graph approach had better performance on complex surfaces because its node density is higher than the mesh-to-graph approach.

Another GNN-enabled approach to BIM object classification was proposed by Wang et al. [12], where they used GraphSAGE to classify rooms into corresponding room types in IFC entities. They constructed graphs from bubble diagrams. The overall accuracy on a test dataset of residential apartments was 72.87%. However, the accuracy varies widely for different room types: it was above 80% for living rooms, bedrooms, kitchens, and dining rooms, while it was below 50% for toilets, balconies, and laundry rooms. This result may be due to the fact that GraphSAGE does not aggregate edge features during propagation. Therefore, the model only takes into account the attribute of rooms and ignores the connection types between them when classifying room types. To address this issue, Wang et al. [12] modified GraphSAGE into SAGE-E, which concatenates both node features and edge features during message passing. The results showed that SAGE-E with four layers had the best accuracy. In the study, the performances of GraphSAGE, GCNs, scalable inception graph neural networks, multiplayer perceptron and decision tree were also tested on the same dataset, and the accuracy results were 70.84%, 49.40%, 60.24%, 63.46%, and 65.38% respectively. Interestingly, although GCNs are a type of GNNs, they had the worst performance. This result could be attributed to the aggregation function of GCNs, which uses a symmetric-normalized approach, which can cause a loss of topological features of the graph [90]. GCNs are usually used for graphs where node features are more important than their graph topology. Therefore, the information on room connections was lost with GCNs, resulting in poor accuracy.

Similar to BIM semantic segmentation, Simonsen et al. [46] applied GNN models for classifying CAD primitives into door and non-door categories. They constructed the graph using an approach introduced in Section 4.8 and compared the performance of six GNN models: GATs, GraphSAGE, attention-based graph neural network (AGNN), topology adaptive graph convolutional network (TAGCN), and graph isomorphism networks (GINs). Among these, GATs achieved the highest accuracy of 97.3%, GINs had the lowest accuracy of 87.6%, and the accuracy of the other models was around 95%.

5.3. Vector shape classification in GIS

As previously outlined in Section 4.6, vector shapes are a frequently encountered element in GIS systems and can be represented as graphs. By analyzing the characteristics of nodes and edges, GNN models can learn feature representations of the graph in the feature space. These representations can then be used to train a classifier that can categorize the graph into a specific class. This type of classification, known as graph-level classification, can be applied to various types of vector shapes, including building shapes and group patterns.

5.3.1. Classification of building shapes

In GIS, building shapes are often represented as 2D vector shapes on a map. Yan et al. [40] proposed a model, known as the graph convolutional autoencoder (GCAE), which uses GCNs and an encoder-decoder framework to cluster building shapes through unsupervised learning. Similarly, Liu et al. [41] proposed a GCN-based model, which employs a convolution operator called TriangleConv, to classify building shapes through supervised learning. Both models take advantage of both local and regional features of nodes to aid in learning the feature representation. The primary difference between the two models is their convolutional operators. The GCAE model updates the node representation through convolution within its k-ordering neighboring nodes, while the TriangleConv only implements one-dimensional convolution on node features and does not consider the topological structure during convolution. According to benchmarking on the same dataset, GCAE model has a better accuracy than TriangleConv.

5.3.2. Classification of building group patterns

The classification of building group patterns plays a vital role in identifying landscape configuration, analyzing the social functions of buildings, and cartographic generalization. As previously discussed, graphs for building groups can be constructed by using DT or MST where buildings are represented as nodes and the lines connecting them as edges [16,42]. Yan et al. [16] proposed a graph convolutional neural network to classify building groups on a GIS map into regular and irregular patterns. They constructed graphs using both DT and MST and found that the use of DT or MST did not significantly impact the accuracy of the model. However, the model's accuracy was found to be sensitive to the degree of the polynomial kernel. The optimal degree for DT graphs was 3 and for MST graphs, it was 4. The model achieved an accuracy of 98.02% on the testing dataset. Similarly, Zhao et al. [42] proposed a GCN model to classify building groups into six patterns further. They constructed graphs using the Constrained Delaunay Triangulation (CDT) method. The building area, shape, and orientation were used as node features. Skip connections were added in the model architecture to pass representations from shallow layers to deep layers, resulting in an accuracy of classification on the test dataset between 90% to 92%.

5.4. Vectorisation of 2D planar images and point clouds

GNNs can also be used for vectorization of 2D planar images and point clouds. In the case of 2D planar images, Zhang et al. [44] proposed a model called Conv-MPN, which uses a different method of convolutional message passing and 3-dimensional node features to vectorize building roof edges from satellite GRB images. In point cloud vectorization, Chuang and Sung [58] proposed a framework that combines the 3DSSD model and GRAN model. The 3DSSD model is used to extract vertices of objects from a point cloud generated by a BIM model. Graphs are then constructed from these vertices, where the vertices are nodes, and edges are determined by referring to the BIM model. Finally, the GRAN model is trained on the constructed graphs. The trained model can then predict the edges between nodes on a new point cloud, thus vectorizing it.

5.5. Floor plan auto-generation and auto-evaluation

Generating floor plans using data-driven approaches is a popular topic in the application of deep learning in architecture. As previously discussed, floorplans can be converted into graph structures, so with the advancement of GNNs, two GNN-based approaches have been proposed in the literature for the task of generating floor plans.

Hu et al. [50] proposed a framework called Graph2Plan that generates floor plans from bubble diagrams and boundary layouts as inputs. The framework employs a GNN module and a CNN module to respectively embed the bubble diagram and boundary layout into feature spaces. The concatenated features from the GNN and CNN modules are used to generate the bounding box of each room, and the refined boxes are generated as output through a series of neural networks. In contrast, Nauata et al. [51] proposed a model called House-GAN, based on generative adversarial networks (GANs), for floor plan generation. House-GAN takes bubble diagrams as input and produces a 3-dimensional room segmentation mask tensor for floor plan generation. The model uses Conv-MPN as its backbone for convolutional message passing due to its satisfactory performance in high-order reasoning. It concatenates the sum-pooled features of both connected rooms and unconnected rooms.

In addition to the differences in network architectures mentioned above, there are two other important distinctions between Graph2Plan and House-GAN. First, Graph2Plan relies on searching for a similar layout from an existing database based on the customer's preference and boundary layout, while House-GAN generates the spatial location of rooms from random initial noise. Second, Graph2Plan takes a boundary layout as one of its inputs, whereas the dimension of the floor boundary in House-GAN is determined by the size of the room segmentation mask tensor.

Expanding floor plans from 2D to 3D, the volumetric design represents the spatial arrangement of spaces across floor levels within a building. Chang et al. [45] proposed an integrated framework of GNNs and GANs called Building-GAN to generate a volumetric design. Building-GAN takes a program graph (i.e., a bubble diagram) and a voxel graph (i.e., a space partition) as inputs to generate a volumetric design. In the generator of GAN, two GNN models are used to embed the program graph and the voxel graph separately. Then, a pointer network updates the node embedding of the voxel graph based on the embedded program graph and voxel graph from the previous layer. As a result, the output is a masked voxel graph that can be used for floor plan generation.

In addition to the generative design of floor plans, GNNs also have the potential for design evaluation. As detailed by et al. [53], a GCN-based module named Neural FPs was utilized. Originally designed for predicting the functions of chemical compounds, this module was adapted to assess the target functional scores of a residential building. This model performs graph-level regression, identifying high-scored patterns (subgraphs) of room settings and combinations. A novel feature of this model is that it predicts subjective scores (i.e., liveability and sleepability) as well, indicating that implicit evaluation criteria can be learned by GNNs too.

5.6. Other applications

Accurate prediction of building energy consumption is a challenging but important task, as it can inform decisions about energy efficiency and sustainability. Hu et al. [31] developed a novel approach to addressing this problem by representing the relationships between building shadows as a directed dynamic graph. In this graph, buildings are represented as nodes and the shadows cast from one building to another are treated as edges. Using the ST-GCN model, they were able to make predictions about building energy consumption based on this graph representation.

Effective grouping of construction activities is a crucial aspect of

project management in the construction industry. Hong et al. [13] addressed this issue by representing construction schedules as graphs, with nodes representing individual construction activities and edges indicating logical relationships between activities. They applied the GCN model to this representation through semi-supervised learning, enabling the model to classify different construction activities.

To improve safety on construction sites, Tang and Golparvar-Fard [56] created a spatio-temporal graph representing the interactions between construction activities, worker body key points, and construction tools. They then used a proposed spatio-temporal GNN model on this graph to identify workers' gestures, which were subsequently utilized for safety evaluation.

6. Limitations, and solved and unsolved challenges of applying GNNs to construction

6.1. Graph data preparation and pre-processing in construction

One of the main challenges in graph development is that the process is often manual, involving tasks such as converting BIM models to point clouds, triangular meshes to point clouds, floor plans to bubble diagrams, and geometric primitives to graphs. These tasks are time-consuming and prone to errors, which can negatively impact the quality of the dataset being prepared. Additionally, this manual process can become a bottleneck when creating large-scale datasets or when working with real-time, dynamic data. This challenge is twofold.

First, the development of graph data is more complex than traditional data such as texts, images, and numbers. The topological structure, i.e., the adjacency matrix, must be defined first, and then node and edge attributes must be assigned to the graph. However, in many cases, topological structures and attributes are stored separately, leading to a risk of losing this information during the conversion process. For example, when converting a BIM model to a point cloud, the geometry of BIM objects originally stored in a Revit file is transformed through software, but the semantic information can be lost in the process and must be manually added to the point cloud [59].

Another challenge in developing graph data is that raw data may be stored in different formats. For example, floor plans may be stored in the form of CAD/BIM models, raster images, or even hand drawings. Due to the diversity of data formats, automatic conversion methods cannot be unified or standardized, and the types of information that can be extracted from different formats also vary. Even when using the same data format, different application domains may require extracting different information, which further complicates the implementation of automatic conversion. For example, a BIM model may need triangular meshes for BIM semantic enrichment [47], whereas floor plan evaluation requires a bubble diagram [53]. This diversity of formats and requirements creates more complexities for the conversion process and more limitations for the efficient use of data.

The process of developing datasets and pre-processing data for training and testing GNN models is complex and time-consuming. Additionally, the size of self-built datasets is often limited, and their quality cannot be guaranteed, which can lead to underfitting or low generalization capability of GNN models. In addition, it becomes difficult to benchmark different models without standard datasets. The availability of open construction-related datasets is essential for adopting machine learning in the construction field. Out of the existing 26 studies, only 7 studies used open databases or datasets, while the other studies had to create their own datasets due to the lack of available open datasets. Currently, most of the open datasets focus mainly on floor plans, such as RPLAN [91], the Repository of Unique Buildings (RUB) [46], and LIFULL HOME'S Dataset [92]. This limits the possibilities for researchers, engineers, and practitioners to use larger, more diverse, and general data.

With this in mind, it is important to explore more forms of graph data in the context of construction. Graph data can generally be categorized into two scenarios: structural scenarios, in which the relational structure is explicit, and non-structural scenarios, in which the relational structure is implicit or absent [8]. For structural scenarios, graph data can be extracted from existing tools, drawings, and other construction objects. In contrast, for non-structural scenarios, images and texts are the common sources of graph data. All graph data identified in this study belong to structural scenarios. Future research should focus on investigating graph data in non-structural scenarios in order to better understand the potential and challenges of graph data in construction.

In order to alleviate the costly manual annotation process, unsupervised learning might be a solution. Particularly, researchers in the AEC field could potentially adapt unsupervised learning techniques for point cloud data from the computer vision domain to address this challenge [93]. Furthermore, it is important to note that data collected from construction sites are often corrupted by noise. The retrieved literature seldom employs data cleaning techniques, or only utilizes basic methods. Future research may consider adopting advanced data cleaning techniques, especially for denoising point cloud data [94,95].

6.2. Graph data processing and Mining in Construction

Many of the GNNs used in the construction field, such as GCNs, GATs, and GraphSAGE, were originally developed in the field of computer science and have been benchmarked using giant graphs such as citation network datasets Citeseer, Cora and Pubmed, the knowledge graph dataset NELL, and the protein-protein interaction (PPI) dataset [24,25,33]. These datasets usually consist of a small number of graphs, but each graph is very large. For example, Cora only contains a single graph but has 2708 nodes and 5429 edges. The optimal number of layers for popular GNNs on these datasets is two or three [24,25,33]. However, it's important to note that the size and nature of construction-related graphs are quite different from those of these benchmark datasets, and the optimal architecture for GNNs used in construction might be different.

In the field of construction, the characteristics of datasets are quite different from those used in computer science. Except for point cloud data, there are more graphs in a dataset, but the size of each individual graph is much smaller. For example, the dataset of Building-GAN [45] contains 120,000 graphs to represent buildings, and each graph only contains a few nodes to represent spaces. The RoomGraph dataset [12] has 244 graphs to represent floor plans, each with 9 to 10 nodes to represent rooms in a floor plan. The size and scale of construction-related datasets pose a unique set of challenges for GNNs, which have mostly been tested on larger, more complex graphs. Therefore, it's important for future research to focus on the adaptation and optimization of GNN architectures for small-scale graphs in the context of construction.

When comparing the graph data derived from computer science and construction, two main distinctions can be observed: i) except for some applications such as point cloud conversions, the size of graphs in the construction field is generally much smaller than the graphs used in computer science benchmarks; ii) the labeled data in the construction field is often scarce.

The first distinction may cause the over-smoothing issue, which is a common problem in GNNs where the model becomes too smooth and loses the ability to capture the fine-grained structure of the data. The second distinction leads to the issues surrounding GNN pre-training, which is when pre-training GNN models on large-scale datasets and fine-tuning them on task-specific, smaller datasets can improve the performance, but it is hard to achieve with scarce labeled data from the construction field.

Thus, these distinctions highlight the unique challenges and opportunities for GNNs in the construction field, the need for adapting GNNs to the characteristics of construction-related data and developing methods for making the best use of the limited labeled data.

6.2.1. Over-smoothing on small graphs in construction

It has been proven that over-smoothing is a common problem for GNNs, as GNNs tend to smooth out the differences between the nodes of a graph due to the message passing mechanism [96,97]. This challenge is more significant in small graphs than in large graphs, as when a GNN model goes deeper, it is likely to sample all nodes of a small graph, making different nodes tend to have similar embeddings at the end [98]. As a result, downstream tasks, such as node classification, are unable to correctly distinguish different nodes from their similar embeddings. Moreover, when using a single-layer GNN, the information from more than two hops cannot be passed to the target node, which can hinder the efficiency of the model. Thus, over-smoothing is a crucial issue that must be addressed when GNNs are applied to small graphs in the construction field. To tackle this problem, techniques such as using deeper GNNs with skip connections, recurrent neural networks, or other architectures that can help preserve the fine-grained structure of the data may be used.

Several approaches have been proposed to address the issue of over-smoothing in graph neural networks (GNNs). One such approach is proposed by Chen et al. [99], in which two metrics, Mean Average Distance (MAD) and MADGap, are used to measure the similarity between embeddings of individual nodes and groups of nodes, respectively. The authors then incorporate the MADGap metric into the loss function to regularize the model and avoid over-smoothing, a technique they refer to as MADReg. Additionally, they propose an algorithm called AdaEdge, which aims to increase the information-to-noise ratio during training by selectively adding and deleting edges between nodes in the graph. Another approach is proposed by Zhou et al. [100], which introduces a technique called Differentiable Group Normalization (DGN) to tackle over-smoothing. DGN is applied between GNN layers and uses a trainable cluster assignment matrix to assign nodes with similar embeddings to the same group. This technique allows for normalization to be applied group-wise, thereby promoting high similarity within groups and large distances between groups. Chen et al. [101] also proposed a metric called the Neighborhood Discrepancy Rate (NDR), which measures the cosine distance between a node and a virtual node aggregated from the node's neighbors. Based on this metric, an approach called Adaptive Discrepancy Retaining (ADR) is designed to ensure that non-similarity in the shallow layers of the GNN is retained as the information flows through deeper layers during message passing.

All of these approaches have shown to be effective in preventing over-smoothing, and can be considered for use in future research, particularly when working with small graphs, such as residential floor plans.

6.2.2. GNN pre-training for graph data in construction

Preparing graph data in the construction industry is a costly endeavor, as obtaining labeled graph data can be scarce. For instance, labeling point cloud data or entities from scan-to-BIM models is a challenging task. However, GNNs require large amounts of data to function effectively. A solution to this issue is to use unlabelled data to pre-train the model as initialization and then use a smaller amount of task-specific labeled data to fine-tune the model for downstream tasks. This approach, known as pre-training, has proven successful in natural language processing and computer vision fields, as demonstrated by models such as GPT-3 and ResNet-50.

Some researchers have proposed training schemes for pre-training on graph data. One such approach is suggested by Hu et al. [102], which proposed that pre-training on graph data requires the pre-trained GNN model to be expressive at both the node and graph levels in order to ensure that semantic information is captured at both levels. They proposed two self-supervised techniques for node-level pre-training: context prediction and attribute masking. Context prediction trains the model to map nodes surrounded by similar structural contexts to similar embeddings in the feature space, while attribute masking trains the model to predict masked node attributes. In addition, the graph is then trained with graph-level supervised learning to ensure the model is robust at the graph level for downstream tasks. Another approach is proposed by Qiu et al. [103], who investigate the pre-training of the structural representation of graph data without attributes. They generate augmented subgraphs from the original graph using random walk sampling and use contrastive self-supervised learning to train GNN encoders to distinguish the similarity between the original graph and the augmented subgraphs, as well as the similarity between the original graph and the pseudo subgraphs generated from a noise distribution. Inspired by the GPT models in NLP, Hu et al. [104] propose a generative pre-training approach. In this approach, a node and its edges are randomly masked, and the GNN model is trained to generate the real nodes' attributes and edges in order to capture both semantic and structure representations of the graph. This generative process is conducted in an autoregressive manner. Xia et al. [105] review the pre-training approaches for graph data. Future research on GNNs in construction can refer to this review to select a suitable pre-training approach.

Existing GNN models may not be fully suitable for the direct use of graphical data in the construction field due to the specificity of the data. Therefore, it may be necessary to modify the original model to achieve the desired results. An example is a study conducted by Wang et al. [12], where they modified the GraphSAGE model to aggregate both node and edge features. This modification is necessary for their study because edges representing room connections are important for analyzing the spatial relationships among rooms. The modified model can better handle the unique characteristics of graph data in the construction field.

6.3. Dynamic and heterogeneous graph data in construction

A construction project is an inherently dynamic process, from planning to operation. The complexity of this process can be seen in two distinct dimensions. Firstly, the temporal dynamics of construction are evident in the ever-changing conditions under which construction occurs. For example, weather patterns shift throughout the day and over the course of a year, impacting work schedules and the feasibility of certain tasks. In the operation phase, occupant behavior fluctuates, influencing the effective utilization and maintenance of the built environment. These temporal changes translate into dynamic node and edge features in the graph data.

In the second dimension, construction is spatially dynamic, continuously altering the physical and relational structure of the project. When translating this spatial dynamism into graph data, the number of nodes and their interconnections, represented as edges, can change significantly. For instance, as the construction project progresses, new building elements are added, and existing ones are modified or removed, leading to an increase, alteration, or decrease in the number of nodes in the graph representation. The relationship between these elements also evolves due to factors such as the sequence of construction tasks, the rearrangement of equipment, and the spatial reorganization of the site. These changes result in the addition, removal, or modification of edges in the graph.

Adding another layer of complexity, construction projects involve a wide variety of elements - materials, components, equipment, and personnel, each with unique attributes and relationships. The materials range from concrete and steel to glass and wood, each with different physical properties and usage scenarios. The equipment used can vary from heavy machinery like cranes and excavators to hand tools and safety gear. The personnel involved in a project can span from architects and engineers to construction workers and site managers, each playing different roles and having different interactions with the project. This results in graph data that is not only dynamic but also highly heterogeneous, with multiple node and edge types.

However, the majority of existing GNN models have been developed with a focus on static and homogeneous graphs [23]. These models assume uniform node and edge types, a static graph structure, and constant node and edge features. This premise is largely inconsistent with

the dynamic and heterogeneous nature of graph data in construction. Thus, applying these traditional GNNs to construction data can lead to significant model bias and inaccurate predictions. Moreover, the continuous evolution of graphs in construction due to temporal and spatial dynamics poses significant challenges to the learning stability and computational efficiency of GNNs. The frequent changes in node features, edge features, and graph structure require GNNs to continuously adapt and learn, which can be computationally expensive and time-consuming.

Given the challenges of dynamic and heterogeneous graph data in construction, it can be beneficial to draw parallels with developments in other fields, such as traffic analysis, social network analysis and recommendation systems. These application fields exhibit similar characteristics to graphs in construction, including dynamics and heterogeneity, and have made substantial progress in applying GNNs to manage these complexities.

For managing the dynamics of graph data, several methodologies have been developed. A prominent approach is the Spatial-temporal Graph Neural Networks (STGNNs), a type of graph neural network specifically designed for dynamic graphs [9]. The architecture of STGNNs is based on the combination of GNNs and RNNs. The GNNs are responsible for extracting and processing features from the graph structure, capturing the spatial relationships between nodes. On the other hand, RNNs, such as Long Short-Term Memory (LSTM) [106] or Gated Recurrent Units (GRU) [107], are used to model the temporal dynamics of the graph, learning from the changes in node and edge features over time. Several types of STGNNs have been developed and applied to social network analysis [106], traffic prediction [108], skeleton-based action recognition [109].

In terms of managing the heterogeneity of graph data, meta-path-based methods have been proposed. Meta-path-based methods identify different types of connections (meta-paths) between nodes to capture the heterogeneous relationships in the graph [110]. For each type of meta-paths, a specific set of parameters of GNNs is used to learn specific patterns of different relationships. Several studies have attempted to convert GNN models to the those compatible with heterogeneous graphs [110–112].

Therefore, while the challenges of dynamic and heterogeneous graph data in construction are significant, they are not insurmountable. By borrowing and adapting advanced GNN methods from other fields, it is possible to develop new methodologies that can effectively handle the dynamic and heterogeneous nature of construction graph data. Such cross-disciplinary exchange can catalyze innovation and drive the advancement of GNN applications in the construction industry."

7. Conclusions and future directions

7.1. Conclusions

There has been a growing interest in the use of GNNs in the construction industry to address challenges that cannot be effectively managed by traditional deep learning models. This review provides an overview of the various and novel applications of GNNs in construction. Although research on GNNs in construction is still in its early stage, the number of studies on GNNs in construction is rapidly increasing, indicating a growing interest in the field, and providing new opportunities for future research.

This paper makes several contributions to the field of GNNs in construction. It is one of the first scholarly attempts to synthesize past research on GNNs in the construction domain and raise awareness in the field about the potential of GNNs. The study identified the various approaches used to construct graphs from objects in construction, providing a foundation for applying GNNs in construction. The current applications of GNNs in construction were also discussed, revealing that current applications mainly focus on point cloud semantic segmentation, object classification, and generative design of floor plans. These findings demonstrate that GNNs can provide promising solutions for handling irregular data in construction. Additionally, the paper offers insight into the challenges and opportunities related to the use of GNNs in the construction industry, providing a valuable point of reference for practitioners and future researchers.

From a broader perspective, this study is original in its approach, as it brings the concept of GNNs from computer science and provides discussions that facilitate the customization of GNNs for construction applications. The study relied on knowledge and theories from computer science, modifying them to suit the unique context of the construction industry. This contributed to "the transferability and applicability" of existing theories from other disciplines to the specific construction setting, thus providing evidence of originality [129]. Furthermore, the study offers an original perspective on GNNs challenges, applications, and possible integration scenarios through the use of "contingency approaches." Contingency approaches provide an original understanding of a topic by highlighting "what processes and practices apply in which contexts, what relationships hold or do not hold in which contexts, and where do methods work and do not work or how do they vary in different contexts" [130].

Despite the contributions of this paper, it is important to acknowledge its limitations. The suggestions for the use of GNNs beyond what has been tried remain conceptual in nature, and future studies should test their validity with empirical data from real-life projects. Additionally, given the novelty of the field, the size of the available literature is limited for a comprehensive assessment of the field, and many studies are still in the process of publication, and some might be confined to industry-based projects, where the details may not always become available in the public domain. This review is, therefore, limited to the major scholarly works in the initial stages of this new field in the construction domain. Future studies are needed to assess the maturity of the domain at later stages.

7.2. Future directions

Moving forward, it is necessary to consider several promising pathways for the application of GNNs in construction. As the technology continues to grow and mature, the upcoming research directions will not only further our understanding of the potential of GNNs but also open doors to new opportunities.

7.2.1. Knowledge graph reasoning in construction

Information exchange in the construction industry heavily relies on documents, memos, and emails, which can lead to inefficiencies and errors in a rapidly changing environment [113]. To address this challenge, some scholars propose transforming the form of information exchange from traditional documents to knowledge graphs [114]. A knowledge graph is a structured representation of human knowledge, composed of two core components: ontology and reasoner. According to [115], a knowledge graph "acquires and integrates information into an ontology and applies a reasoner to derive new knowledge representation learning."

In the field of construction, a variety of ontologies have been established for different scenarios, such as concrete bridge rehabilitation [116], road asset management [117], and construction defects [118]. However, most reasoners on these ontologies are rule-based and require manual intervention [117], which means that the process of deriving new knowledge from the ontology is not fully automated, limiting the scalability and efficiency of knowledge graph-based information exchange in the construction industry.

From the perspective of graphs, a knowledge graph is a directed and heterogeneous graph consisting of entities (nodes) and relations (edges) [8]. The graph structure of a knowledge graph makes it possible to adopt the GNN model in the reasoner of knowledge graphs. Apart from the construction industry, several GNN-based approaches have been used for knowledge graph embedding, relation extraction and graph

completion [119,120]. These studies confirm the feasibility of exploring the adoption of GNN-based in construction-related ontologies. For example, in the construction industry, structured information can be extracted from the BIM model, and unstructured information can be extracted using natural language processing (NLP) from project documents. The extracted information is then applied to the predefined ontology, forming a graph. On the constructed knowledge graph, GNN models can classify duplicated entities and predict missing entities or relations [120]. The adoption of GNNs can help to improve the efficiency and accuracy of information exchange in the construction industry by automating the process of deriving new knowledge from the ontology.

7.2.2. Generative design

Generative design (GD) is a design process that employs computer techniques to generate a variety of designs based on predefined constraints and objectives, as established by designers at the outset of the process [121]. A typical GD system comprises two main components: a generator and an evaluator. The generator takes design constraints as inputs and generates alternative designs, while the evaluator assesses all alternative designs against design objectives and selects the most suitable designs for the designer to make a final decision. GD has been applied in the construction industry in various fields, such as architecture, structural engineering, and urban design [122]. However, current GD approaches in construction are still in their infancy and are heavily reliant on algorithms, which are not capable of generating the complex geometry of structures and buildings [123]. To develop the generative algorithm, design parameters must be extracted manually, and the relationships among parameters must be determined manually [124]. Additionally, a certain level of programming expertise is required to encode design objectives and constraints into a computer-readable format [122]. To address these challenges, scholars are advocating for the exploration of data-driven GD approaches, particularly deep learning approaches [123], which offer a promising avenue for future research in the field.

As a promising deep learning approach, GNNs may be a solution for the current limitations in generative design in construction. GNNs are able to learn not only explicit design rules but also implicit design rules from existing designs, allowing for the generation of more complex alternative designs [53]. Furthermore, GNNs are an end-to-end learning model, meaning that raw data can be directly fed into the model without requiring manual feature extraction, reducing the requirement for programming skills. GANs have been integrated with GNNs to generate floor plans in the existing studies [45,51]. In these studies, the generator and discriminator are trained to learn the design rules and evaluation rules. Compared to the classic GD, a fully trained GNN-based GD can directly generate suitable designs without the need for iteration or evaluation, resulting in a reduction in calculation time. As previously discussed, the potential of GNNs in generative design has not been fully explored, and the current adoption of GNNs in GD is mainly limited to the generation of floor plans [45,51,53]. Further adoption of GNN-based GD in other areas, such as structure design, plumbing plans, HVAC ducting, and urban design, has yet to be explored.

7.2.3. Integration with BIM and the internet of things

Integrating machine learning with BIM and the Internet of Things (IoT) has emerged as a promising area of research, offering significant opportunities to enhance various aspects of the AEC domain, including design, construction, and facility management [113,125,126]. Specifically, MLPs and CNNs have demonstrated successful applications in this context [127,128].

However, current GNN applications in BIM are predominantly centered on the identification and categorization of BIM entities. GNNs possess the capacity to scrutinize intricate relationships within BIM models, including spatial, functional, and structural dependencies, by representing them as graphs. This analysis can aid in detecting potential design clashes, facilitating automated code compliance checks, and augmenting the overall design quality. Additionally, GNNs can help streamline construction processes by optimizing resource allocation, scheduling, and coordination based on BIM graph representations. GNNs also have the potential to bolster facility management tasks by extracting valuable insights from BIM models and evaluating the repercussions of design alterations on energy consumption, maintenance needs and space utilization. By incorporating GNNs with BIM, AEC professionals can attain enhanced collaboration, heightened efficiency, and better-informed decision-making throughout a project's lifecycle.

The integration of GNNs and BIM can be significantly enriched by the incorporation of IoT. As the construction industry increasingly adopts IoT technologies, a vast amount of real-time data is being generated from various sources, including embedded sensors in construction machinery, equipment, and building components [38]. This high-dimensional, heterogeneous, and dynamic data is a valuable resource for training GNNs in construction. When converting the IoT data as graphs, GNNs can capture the interdependencies between different IoT devices and their temporal data, providing a more comprehensive understanding of the construction site or built environment. This real-time data and analysis can be used to update the BIM models, making them "live" representations of the actual structures.

The integration of GNNs, BIM, and IoT could enable a variety of applications, such as real-time monitoring and analysis of construction processes, predictive maintenance of building systems, and more efficient facility management. For example, the combination of IoT sensor data, BIM, and GNNs could be used to predict and diagnose equipment failures, identify patterns that lead to downtime, and offer preventative solutions.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No data was used for the research described in the article.

Appendix A. Search String on Scopus

TITLE-ABS-KEY (graph) AND ALL ("deep learning" OR "machine learning" OR "neural network" OR "neural network" OR "artificial intelligence" OR gnn OR "graph neural network") AND SRCTITLE ("Automation in Construction" OR "Journal of Construction Engineering and Management" OR "Journal of Computing in Civil Engineering" OR "Construction Management and Economics" OR "Engineering, Constrt" OR "Construction Innovation" OR "Canadian Journal of Civil Engineering" OR "International Journal of Construction Education and Research" OR "Building Research and Information" OR "Architectural Engineering and Design Management" OR "Building and Environment" OR "International Journal of Construction Management" OR "Australasian Journal of Construction Economics and Building" OR "Applied Energy" OR "Computer-Aided Civil and Infrastructure Engineering" OR "Applied Thermal Engineering" OR "Energy" OR "Energy and Buildings" OR "Journal of Energy Engineering" OR "Sustainable Energy Technologies and Assessment" OR "Energy Conversion and Management" OR "Building and Environment" OR "Construction and Building Materials" OR "Computers and Structures" OR "Archives of Civil and Mechanical Engineering" OR "Materials and Structures" OR "Smart Materials and Structures" OR "Journal of Building Performance Simulation" OR "Journal of Building Engineering" OR "Journal of Civil Engineering and Management" OR "Building Simulation" OR "buildings" OR "Architectural Engineering and Design Management" OR "European Journal of Environmental and Civil Engineering" OR "Journal of Civil Engineering

Education" OR "Proceedings of the Institution of Civil Engineers" OR "International Journal of Civil Engineering").

References

[1] J.S. Lee, J. Park, Y.-M. Ryu, Semantic segmentation of bridge components based on hierarchical point cloud model, Automation in Construction 130 (2021), 103847, https://doi.org/10.1016/j.autcon.2021.103847.

[2] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, Nature 521 (7553) (2015) 436–444, https://doi.org/10.1038/nature14539.

[3] Y. Xu, Y. Zhou, P. Sekula, L. Ding, Machine learning in construction: from shallow to deep learning, Developments in the Built Environment 6 (2021), 100045, https://doi.org/10.1016/j.dibe.2021.100045.

[4] T.D. Akinosho, L.O. Oyedele, M. Bilal, A.O. Ajayi, M.D. Delgado, O.O. Akinade, A. A. Ahmed, Deep learning in the construction industry: a review of present status and future innovations, Journal of Building Engineering 32 (2020), 101827, https://doi.org/10.1016/j.jobe.2020.101827.

[5] F. Elghaish, S. Talebi, E. Abdellatef, S.T. Matarneh, M.R. Hosseini, S. Wu, M. Mayouf, A. Hajirasouli, T.-Q. Nguyen, Developing a new deep learning CNN model to detect and classify highway cracks, Journal of Engineering, Design and Technology 20 (4) (2022) 993–1014, https://doi.org/10.1108/JEDT-04-2021-0192.

[6] C. Wu, X. Wang, P. Wu, J. Wang, R. Jiang, M. Chen, M. Swapan, Hybrid deep learning model for automating constraint modelling in advanced working packaging, Automation in Construction 127 (2021), 103733, https://doi.org/10.1016/j.autcon.2021.103733.

[7] C. Fan, F. Xiao, Y. Zhao, A short-term building cooling load prediction method using deep learning algorithms, Applied Energy 195 (2017) 222–233, https://doi.org/10.1016/j.apenergy.2017.03.064.

[8] J. Zhou, G. Cui, S. Hu, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, M. Sun, Graph neural networks: a review of methods and applications, AI Open 1 (2020) 57–81, https://doi.org/10.1016/j.aiopen.2021.01.001.

[9] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, P.S. Yu, A comprehensive survey on graph neural networks, IEEE Transactions on Neural Networks and Learning Systems 32 (1) (2021) 4–24, https://doi.org/10.1109/TNNLS.2020.2978386.

[10] W. Jiang, J. Luo, Graph neural network for traffic forecasting: a survey, Expert Systems with Applications 207 (2022), 117921, https://doi.org/10.1016/j.eswa.2022.117921.

[11] C. Wu, X. Li, R. Jiang, Y. Guo, J. Wang, Z. Yang, Graph-based deep learning model for knowledge base completion in constraint management of construction projects, Computer-Aided Civil and Infrastructure Engineering 38 (6) (2023) 702–719, https://doi.org/10.1111/mice.12904.

[12] Z. Wang, R. Sacks, T. Yeung, Exploring graph neural networks for semantic enrichment: room type classification, Automation in Construction 134 (2022), 104039, https://doi.org/10.1016/j.autcon.2021.104039.

[13] Y. Hong, V. Hovhannisyan, H. Xie, I. Brilakis, Determining construction method patterns to automate and optimise scheduling – a graph-based approach, in: 2021 European Conference on Computing in Construction, 2021, pp. 59–66, https://doi.org/10.35490/ec3.2021.165.

[14] M. Zitnik, M. Agrawal, J. Leskovec, Modeling polypharmacy side effects with graph convolutional networks, Bioinformatics 34 (13) (2018) i457–i466, https://doi.org/10.1093/bioinformatics/bty294.

[15] J. Gilmer, S.S. Schoenholz, P.F. Riley, O. Vinyals, G.E. Dahl, Neural message passing for quantum chemistry, in: Proceedings of the 34th International Conference on Machine Learning, 2017, pp. 1263–1272, https://doi.org/10.5555/3305381.3305512, accessed January 16 2023.

[16] X. Yan, T. Ai, M. Yang, H. Yin, A graph convolutional neural network for classification of building patterns using spatial vector data, ISPRS Journal of Photogrammetry and Remote Sensing 150 (2019) 259–273, https://doi.org/10.1016/j.isprsjprs.2019.02.010.

[17] H.A. Abu Alfeilat, A.B.A. Hassanat, O. Lasassmeh, A.S. Tarawneh, M.B. Alhasanat, H.S. Eyal Salman, V.B.S. Prasath, Effects of distance measure choice on k-nearest neighbor classifier performance: a review, Big Data 7 (4) (2019) 221–248, https://doi.org/10.1089/big.2018.0175.

[18] H. Kim, C. Kim, Deep-learning-based classification of point clouds for bridge inspection, Remote Sensing 12 (22) (2020) 3757, https://doi.org/10.3390/rs12223757.

[19] Y. Jia, J. Wang, M.R. Hosseini, W. Shou, Graph neural networks in building life cycle: a review, in: 2022 European Conference on Computing in Construction, 2022, pp. 1–8, https://doi.org/10.35490/EC3.2022.164.

[20] M.R. Hosseini, I. Martek, E.K. Zavadskas, A.A. Aibinu, M. Arashpour, N. Chileshe, Critical evaluation of off-site construction research: a scientometric analysis, Automation in Construction 87 (2018) 235–247, https://doi.org/10.1016/j.autcon.2017.12.002.

[21] S. Zhang, H. Tong, J. Xu, R. Maciejewski, Graph convolutional networks: a comprehensive review, Computational Social Networks 6 (1) (2019) 11, https://doi.org/10.1186/s40649-019-0069-y.

[22] M.M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, P. Vandergheynst, Geometric deep learning: going beyond euclidean data, IEEE Signal Processing Magazine 34 (4) (2017) 18–42, https://doi.org/10.1109/MSP.2017.2693418.

[23] Z. Zhang, P. Cui, W. Zhu, Deep learning on graphs: a survey, IEEE Transactions on Knowledge and Data Engineering 34 (1) (2022) 249–270, https://doi.org/10.1109/TKDE.2020.2981333.

[24] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Lio, Y. Bengio, Graph attention networks, in: 6th International Conference on Learning Representations (ICLR 2018), 2018, pp. 1–12, https://doi.org/10.48550/arXiv.1710.10903.

[25] T.N. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks, in: 5th International Conference on Learning Representations (ICLR 2017), 2017, pp. 1–14, https://doi.org/10.48550/arXiv.1609.02907.

[26] J.H. Giraldo, S. Javed, N. Werghi, T. Bouwmans, Graph CNN for moving object detection in complex environments from unseen videos, in: 2021 IEEE/CVF International Conference on Computer Vision Workshops (ICCVW), 2021, pp. 225–233, https://doi.org/10.1109/ICCVW54120.2021.00030.

[27] J. Xiong, Z. Xiong, K. Chen, H. Jiang, M. Zheng, Graph neural networks for automated de novo drug design, Drug Discovery Today 26 (6) (2021) 1382–1393, https://doi.org/10.1016/j.drudis.2021.02.011.

[28] O. Wieder, S. Kohlbacher, M. Kuenemann, A. Garon, P. Ducrot, T. Seidel, T. Langer, A compact review of molecular property prediction with graph neural networks, Drug Discovery Today: Technologies 37 (2020) 1–12, https://doi.org/10.1016/j.ddtec.2020.11.009.

[29] W. Liao, B. Bak-Jensen, J.R. Pillai, Y. Wang, Y. Wang, A review of graph neural networks and their applications in power systems, Journal of Modern Power Systems and Clean Energy 10 (2) (2022) 345–360, https://doi.org/10.35833/MPCE.2021.000058.

[30] M. Gori, G. Monfardini, F. Scarselli, A new model for learning in graph domains, in: 2005 IEEE International Joint Conference on Neural Networks, 2005, pp. 729–734, https://doi.org/10.1109/IJCNN.2005.1555942.

[31] Y. Hu, X. Cheng, S. Wang, J. Chen, T. Zhao, E. Dai, Times series forecasting for urban building energy consumption based on graph convolutional network, Applied Energy 307 (2022), 118231, https://doi.org/10.1016/j.apenergy.2021.118231.

[32] F. Scarselli, M. Gori, A.C. Tsoi, M. Hagenbuchner, G. Monfardini, The graph neural network model, IEEE Transactions on Neural Networks 20 (1) (2009) 61–80, https://doi.org/10.1109/TNN.2008.2005605.

[33] W.L. Hamilton, Z. Ying, J. Leskovec, Inductive representation learning on large graphs, in: Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS 2017), 2017, pp. 1025–1035, https://doi.org/10.5555/3294771.3294869, accessed March 15 2022.

[34] L. Wu, Y. Chen, K. Shen, X. Guo, H. Gao, S. Li, J. Pei, B. Long, Graph neural networks for natural language processing: a survey, foundations and trends®, Machine Learning 16 (2) (2023) 119–328, https://doi.org/10.1561/2200000096.

[35] D. Moher, A. Liberati, J. Tetzlaff, D.G. Altman, Preferred reporting items for systematic reviews and meta-analyses: the PRISMA statement, BMJ 339 (2009), b2535, https://doi.org/10.1136/bmj.b2535.

[36] M. Oraee, M.R. Hosseini, E. Papadonikolaki, R. Palliyaguru, M. Arashpour, Collaboration in BIM-based construction networks: a bibliometric-qualitative literature review, International Journal of Project Management 35 (7) (2017) 1288–1301, https://doi.org/10.1016/j.ijproman.2017.07.001.

[37] M.R. Hosseini, N. Chileshe, J. Zuo, B. Baroudi, Adopting global virtual engineering teams in AEC projects, Construction Innovation 15 (2) (2015) 151–179, https://doi.org/10.1108/CI-12-2013-0058.

[38] A. Ghosh, D.J. Edwards, M.R. Hosseini, Patterns and trends in internet of things (IoT) research: future applications in the construction industry, Engineering Construction and Architectural Management 28 (2) (2021) 457–481, https://doi.org/10.1108/ECAM-04-2020-0271.

[39] S. Hopewell, S. McDonald, M.J. Clarke, M. Egger, Grey literature in meta-analyses of randomized trials of health care interventions, Cochrane Database of Systematic Reviews 2 (2007), https://doi.org/10.1002/14651858.MR000010.pub3. MR000010.

[40] X. Yan, T. Ai, M. Yang, X. Tong, Graph convolutional autoencoder model for the shape coding and cognition of buildings in maps, International Journal of Geographical Information Science 35 (3) (2021) 490–512, https://doi.org/10.1080/13658816.2020.1768260.

[41] C. Liu, Y. Hu, Z. Li, J. Xu, Z. Han, J. Guo, TriangleConv: a deep point convolutional network for recognizing building shapes in map space, ISPRS International Journal of Geo-Information 10 (10) (2021) 687, https://doi.org/10.3390/ijgi10100687.

[42] R. Zhao, T. Ai, W. Yu, Y. He, Y. Shen, Recognition of building group patterns using graph convolutional network, Cartography and Geographic Information Science 47 (5) (2020) 400–417, https://doi.org/10.1080/15230406.2020.1757512.

[43] F. Collins, Encoding of Geometric Shapes from Building Information Modeling (BIM) using Graph Neural Networks, Technische Universität München, 2020. https://mediatum.ub.tum.de/1577170 (accessed January 18 2022).

[44] F. Zhang, N. Nauata, Y. Furukawa, CONV-MPN: convolutional message passing neural network for structured outdoor architecture reconstruction, in: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020, pp. 2795–2804, https://doi.org/10.1109/CVPR42600.2020.00287.

[45] K.-H. Chang, C.-Y. Cheng, J. Luo, S. Murata, M. Nourbakhsh, Y. Tsuji, Building-GAN: Graph-Conditioned Architectural Volumetric Design Generation, arXiv preprint, arXiv:2104.13316, 2021, https://doi.org/10.48550/arXiv.2104.13316.

[46] C.P. Simonsen, F.M. Thiesson, M.P. Philipsen, T.B. Moeslund, Generalizing floor plans using graph neural networks, in: 2021 IEEE International Conference on Image Processing (ICIP), 2021, pp. 654–658, https://doi.org/10.1109/ICIP42928.2021.9506514.

[47] F.C. Collins, A. Braun, M. Ringsquandl, D.M. Hall, A. Borrmann, Assessing IFC classes with means of geometric deep learning on different graph encodings, in: 2021 European Conference on Computing in Construction, 2021, pp. 332–341, https://doi.org/10.35490/ec3.2021.168.

[48] Z. Wang, T. Yeung, R. Sacks, Z. Su, Room type classification for semantic enrichment of building information modeling using graph neural networks, in: Proceedings of the 38th International Conference of CIB W78, 2021, pp. 773–781. https://itc.scix.net/paper/w78-2021-paper-077. accessed April 15 2022.
[49] M. Del Campo, A. Carlson, S. Manninger, 3D graph convolutional neural networks in architecture design, in: Proceedings of the 40th Annual Conference of the Association of Computer Aided Design in Architecture (ACADIA), 2020, pp. 688–696. http://papers.cumincad.org/cgi-bin/works/paper/acadia20_688 (accessed January 18 2022).
[50] R. Hu, Z. Huang, Y. Tang, O.V. Kaick, H. Zhang, H. Huang, Graph2Plan: learning floorplan generation from layout graphs, ACM Transactions on Graphics 39 (4) (2020) 118, https://doi.org/10.1145/3386569.3392391.
[51] N. Nauata, K.-H. Chang, C.-Y. Cheng, G. Mori, Y. Furukawa, House-GAN: Relational Generative Adversarial Networks for Graph-Constrained House Layout Generation, arXiv preprint, arXiv:2003.06988, 2020, https://doi.org/10.48550/arXiv.2003.06988.
[52] H. Zhang, Text-to-form, in: Proceedings of the 40th Annual Conference of the Association for Computer Aided Design in Architecture (ACADIA), 2020, pp. 238–247. http://papers.cumincad.org/cgi-bin/works/Show?acadia20_238 (accessed January 18 2022).
[53] I. As, S. Pal, P. Basu, Composing frankensteins: Data-driven design assemblies through graph-based deep neural networks, in: In: 107th ACSA Annual Meeting Proceedings, Black Box, 2019, pp. 433–438, https://doi.org/10.35483/ACSA.AM.107.90.
[54] K. Wang, Y.-A. Lin, B. Weissmann, M. Savva, A.X. Chang, D. Ritchie, PlanIT: planning and instantiating indoor scenes with relation graph and spatial prior networks, ACM Transactions on Graphics 38 (4) (2019) 132, https://doi.org/10.1145/3306346.3322941.
[55] I. As, S. Pal, P. Basu, Artificial intelligence in architecture: generating conceptual design via deep learning, International Journal of Architectural Computing 16 (4) (2018) 306–327, https://doi.org/10.1177/1478077118800982.
[56] S. Tang, M. Golparvar-Fard, Machine learning-based risk analysis for construction worker safety from ubiquitous site photos and videos, Journal of Computing in Civil Engineering 35 (6) (2021) 04021020, https://doi.org/10.1061/(ASCE)CP.1943-5487.0000979.
[57] H. Kim, C. Kim, 3D as-built modeling from incomplete point clouds using connectivity relations, Automation in Construction 130 (2021), 103855, https://doi.org/10.1016/j.autcon.2021.103855.
[58] T.-Y. Chuang, C.-C. Sung, Learning-guided point cloud vectorization for building component modeling, Automation in Construction 132 (2021), 103978, https://doi.org/10.1016/j.autcon.2021.103978.
[59] J.W. Ma, T. Czerniawski, F. Leite, Semantic segmentation of point clouds of building interiors with deep learning: augmenting training datasets with synthetic BIM-based point clouds, Automation in Construction 113 (2020), 103144, https://doi.org/10.1016/j.autcon.2020.103144.
[60] Z. Chen, Q. Deng, H. Ren, Z. Zhao, T. Peng, C. Yang, W. Gui, A new energy consumption prediction method for chillers based on GraphSAGE by combining empirical knowledge and operating data, Applied Energy 310 (2022), 118410, https://doi.org/10.1016/j.apenergy.2021.118410.
[61] W. Duan, Y. Wang, J. Li, Y. Zheng, C. Ning, P. Duan, Real-time surveillance-video-based personalized thermal comfort recognition, Energy and Buildings 244 (2021), 110989, https://doi.org/10.1016/j.enbuild.2021.110989.
[62] Y. Yajima, S. Kim, J. Chen, Y.K. Cho, Fast online incremental segmentation of 3D point clouds from disaster sites, in: Proceedings of the 38th International Symposium on Automation and Robotics in Construction (ISARC 2021), 2021, pp. 341–348, https://doi.org/10.22260/ISARC2021/0048.
[63] H. Feng, W. Li, Z. Luo, Y. Chen, S.N. Fatholahi, M. Cheng, C. Wang, J.M. Junior, J. Li, GCN-based pavement crack detection using mobile LiDAR point clouds, IEEE Transactions on Intelligent Transportation Systems 23 (8) (2022) 11052–11061, https://doi.org/10.1109/TITS.2021.3099023.
[64] F. Bahreini, A. Hammad, Point cloud semantic segmentation of concrete surface defects using dynamic graph CNN, in: 38th International Symposium on Automation and Robotics in Construction (ISARC), 2021, pp. 379–386, https://doi.org/10.22260/ISARC2021/0053.
[65] L. Zhou, S.-X. Chen, Y.-Q. Ni, A.W.-H. Choy, EMI-GCN: a hybrid model for real-time monitoring of multiple bolt looseness using electromechanical impedance and graph convolutional networks, Smart Materials and Structures 30 (3) (2021), 035032, https://doi.org/10.1088/1361-665x/abe292.
[66] W. Tan, N. Qin, L. Ma, Y. Li, J. Du, G. Cai, K. Yang, J. Li, Toronto-3D: a large-scale mobile LiDAR dataset for semantic segmentation of urban roadways, in: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 2020, pp. 797–806, https://doi.org/10.1109/CVPRW50498.2020.00109.
[67] R. Pierdicca, M. Paolanti, F. Matrone, M. Martini, C. Morbidoni, E.S. Malinverni, E. Frontoni, A.M. Lingua, Point cloud semantic segmentation using a deep learning framework for cultural heritage, Remote Sensing 12 (6) (2020) 1005, https://doi.org/10.3390/rs12061005.
[68] International Organization for Standardization [ISO], ISO 10303-11:2004 Industrial Automation Systems and Integration — Product Data Representation and Exchange — part 11: Description Methods: The Express Language Reference Manual, 2004, https://au.i2.saiglobal.com/management/display/anchor/420168 (accessed May 26 2022).
[69] E. Tauscher, H.-J. Bargstädt, K. Smarsly, Generic BIM queries based on the IFC object model using graph theory, in: Proceedings of the 16th International Conference on Computing in Civil and Building Engineering, 2016, pp. 905–912. http://www.see.eng.osaka-u.ac.jp/seeit/icccbe2016/pages/conference5.html. accessed May 31 2023.
[70] A. Ismail, A. Nahar, R. Scherer, Application of graph databases and graph theory concepts for advanced analysing of BIM models based on IFC standard, in: Proceedings of 24th International Workshop on Intelligent Computing in Engineering (EG-ICE 2017), 2017, pp. 1–12. https://ifcwebserver.org/Learn.html. accessed 2 July 2022.
[71] A. Ismail, B. Strug, G. Ślusarczyk, Building knowledge extraction from BIM/IFC data for analysis in graph databases, in: 17th International Conference on Artificial Intelligence and Soft Computing (ICAISC 2018), 2018, pp. 652–664, https://doi.org/10.1007/978-3-319-91262-2_57.
[72] A. Khalili, D.K.H. Chua, IFC-based graph data model for topological queries on building elements, Journal of Computing in Civil Engineering 29 (3) (2015) 04014046, https://doi.org/10.1061/(ASCE)CP.1943-5487.0000331.
[73] Y. Wang, Y. Sun, Z. Liu, S.E. Sarma, M.M. Bronstein, J.M. Solomon, Dynamic graph CNN for learning on point clouds, ACM Transactions on Graphics 38 (5) (2019) 146, https://doi.org/10.1145/3326362.
[74] C. Yuan, H. Yang, Research on k-value selection method of k-means clustering algorithm, J 2 (2) (2019) 226–235, https://doi.org/10.3390/j2020016.
[75] F. Pomerleau, F. Colas, R. Siegwart, A review of point cloud registration algorithms for mobile robotics, foundations and trends®, Robotics 4 (1) (2015) 1–104, https://doi.org/10.1561/2300000035.
[76] W. Sun, C. Bradley, Y.F. Zhang, H.T. Loh, Cloud data modelling employing a unified, non-redundant triangular mesh, Computer-Aided Design 33 (2) (2001) 183–193, https://doi.org/10.1016/S0010-4485(00)00088-9.
[77] D.T. Lee, B.J. Schachter, Two algorithms for constructing a delaunay triangulation, International Journal of Computer and Information Sciences 9 (3) (1980) 219–242, https://doi.org/10.1007/BF00977785.
[78] M. Tüceryan, T. Chorzempa, Relative sensitivity of a family of closest-point graphs in computer vision applications, Pattern Recognition 24 (5) (1991) 361–373, https://doi.org/10.1016/0031-3203(91)90050-F.
[79] S. Rebay, Efficient unstructured mesh generation by means of delaunay triangulation and bowyer-Watson algorithm, Journal of Computational Physics 106 (1) (1993) 125–138, https://doi.org/10.1006/jcph.1993.1097.
[80] A. Saalfeld, Topologically consistent line simplification with the Douglas-Peucker algorithm, Cartography and Geographic Information Science 26 (1) (1999) 7–18, https://doi.org/10.1559/152304099782424901.
[81] C. Liu, J. Wu, P. Kohli, Y. Furukawa, Raster-to-vector: revisiting floorplan transformation, in: 2017 IEEE International Conference on Computer Vision (ICCV), 2017, pp. 2214–2222, https://doi.org/10.1109/ICCV.2017.241.
[82] C.R. Qi, H. Su, K. Mo, L.J. Guibas, Pointnet: deep learning on point sets for 3D classification and segmentation, in: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 77–85, https://doi.org/10.1109/CVPR.2017.16.
[83] C. Qi, L. Yi, H. Su, L.J. Guibas, PointNet++: Deep hierarchical feature learning on point sets in a metric space, in: Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS 2017), 2017, pp. 5105–5114, https://doi.org/10.48550/arXiv.1706.02413.
[84] Y. Li, R. Bu, M. Sun, W. Wu, X. Di, B. Chen, PointCNN: Convolution on X-Transformed Points, arXiv preprint, arXiv:1801.07791, 2018, https://doi.org/10.48550/arXiv.1801.07791.
[85] M. Atzmon, H. Maron, Y. Lipman, Point Convolutional Neural Networks by Extension Operators, arXiv preprint, arXiv:1803.10091, 2018, https://doi.org/10.48550/arXiv.1803.10091.
[86] M. Belsky, R. Sacks, I. Brilakis, Semantic enrichment for building information modeling, Computer-Aided Civil and Infrastructure Engineering 31 (4) (2016) 261–274, https://doi.org/10.1111/mice.12128.
[87] T. Bloch, R. Sacks, Clustering information types for semantic enrichment of building information models to support automated code compliance checking, Journal of Computing in Civil Engineering 34 (6) (2020) 04020040, https://doi.org/10.1061/(ASCE)CP.1943-5487.0000922.
[88] R. Sacks, L. Ma, R. Yosef, A. Borrmann, S. Daum, U. Kattel, Semantic enrichment for building information modeling: procedure for compiling inference rules and operators for complex geometry, Journal of Computing in Civil Engineering 31 (6) (2017) 04017062, https://doi.org/10.1061/(ASCE)CP.1943-5487.0000705.
[89] L. Ma, R. Sacks, U. Kattel, T. Bloch, 3D object classification using geometric features and pairwise relationships, Computer-Aided Civil and Infrastructure Engineering 33 (2) (2018) 152–164, https://doi.org/10.1111/mice.12336.
[90] W.L. Hamilton, Graph representation learning, synthesis lectures on artificial intelligence and machine, Learning 14 (3) (2020) 54. https://www.cs.mcgill.ca/~wlh/grl_book/ (accessed March 15 2022).
[91] W. Wu, X.-M. Fu, R. Tang, Y. Wang, Y.-H. Qi, L. Liu, Data-driven interior plan generation for residential buildings, ACM Transactions on Graphics 38 (6) (2019) 234, https://doi.org/10.1145/3355089.3356556.
[92] National Institute of Informatics, LIFULL HOME'S Dataset. https://www.nii.ac.jp/dsc/idr/en/lifull/, 2021 accessed June 2 2022.
[93] H. Chen, S. Luo, X. Gao, W. Hu, Unsupervised learning of geometric sampling invariant representations for 3D point clouds, in: 2021 IEEE/CVF International Conference on Computer Vision Workshops (ICCVW), 2021, pp. 893–903, https://doi.org/10.1109/ICCVW54120.2021.00105.
[94] C. Dinesh, G. Cheung, I.V. Bajić, Point cloud denoising via feature graph Laplacian regularization, IEEE Transactions on Image Processing 29 (2020) 4143–4158, https://doi.org/10.1109/TIP.2020.2969052.
[95] J. Zeng, G. Cheung, M. Ng, J. Pang, C. Yang, 3D point cloud denoising using graph Laplacian regularization of a low dimensional manifold model, IEEE Transactions

on Image Processing 29 (2020) 3474–3489, https://doi.org/10.1109/TIP.2019.2961429.
[96] C. Cai, Y. Wang, A Note on Over-Smoothing for Graph Neural Networks, arXiv preprint, arXiv:2006.13318, 2020, https://doi.org/10.48550/arXiv.2006.13318.
[97] J.H. Giraldo, F.D. Malliaros, T. Bouwmans, Understanding the Relationship between Over-Smoothing and Over-Squashing in Graph Neural Networks, arXiv preprint, arXiv:2212.02374, 2022, https://doi.org/10.48550/arXiv.2212.02374.
[98] P. Elinas, E.V. Bonilla, Addressing Over-Smoothing in Graph Neural Networks via Deep Supervision, arXiv preprint, arXiv:2202.12508, 2022, https://doi.org/10.48550/arXiv.2202.12508.
[99] D. Chen, Y. Lin, W. Li, P. Li, J. Zhou, X. Sun, Measuring and relieving the over-smoothing problem for graph neural networks from the topological view, in: 34th AAAI Conference on Artificial Intelligence (AAAI-20), 2020, pp. 3438–3445, https://doi.org/10.1609/aaai.v34i04.5747.
[100] K. Zhou, X. Huang, Y. Li, D. Zha, R. Chen, X. Hu, Towards deeper graph neural networks with differentiable group normalization, in: Proceedings of the 34th International Conference on Neural Information Processing Systems (NeurIPS 2020), 2020, pp. 4917–4928, https://doi.org/10.5555/3495724.3496137, accessed May 25 2022.
[101] Y. Chen, Y. Bian, X. Xiao, Y. Rong, T. Xu, J. Huang, On Self-Distilling Graph Neural Network, arXiv preprint, arXiv:2011.02255, 2020, https://doi.org/10.48550/arXiv.2011.02255.
[102] W. Hu, B. Liu, J. Gomes, M. Zitnik, P. Liang, V. Pande, J. Leskovec, Strategies for Pre-Training Graph Neural Networks, arXiv preprint, arXiv:1312.6203, 2019, https://doi.org/10.48550/arXiv.1312.6203.
[103] J. Qiu, Q. Chen, Y. Dong, J. Zhang, H. Yang, M. Ding, K. Wang, J. Tang, GCC: Graph contrastive coding for graph neural network pre-training, in: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2020, pp. 1150–1160, https://doi.org/10.1145/3394486.3403168.
[104] Z. Hu, Y. Dong, K. Wang, K.-W. Chang, Y. Sun, GPT-GNN: generative pre-training of graph neural networks, in: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2020, pp. 1857–1867, https://doi.org/10.1145/3394486.3403237.
[105] J. Xia, Y. Zhu, Y. Du, S.Z. Li, A Survey of Pretraining on Graphs: Taxonomy, Methods, and Applications, arXiv preprint, arXiv:2202.07893, 2022, https://doi.org/10.48550/arXiv.2202.07893.
[106] F. Manessi, A. Rozza, M. Manzo, Dynamic graph convolutional networks, Pattern Recognition 97 (2020), 107000, https://doi.org/10.1016/j.patcog.2019.107000.
[107] Y. Seo, M. Defferrard, P. Vandergheynst, X. Bresson, Structured Sequence Modeling with Graph Convolutional Recurrent Networks, arXiv preprint, arXiv:1612.07659, 2016, https://doi.org/10.48550/arXiv.1612.07659.
[108] J. Bai, J. Zhu, Y. Song, L. Zhao, Z. Hou, R. Du, H. Li, A3T-GCN: attention temporal graph convolutional network for traffic forecasting, ISPRS International Journal of Geo-Information 10 (7) (2021) 485, https://doi.org/10.3390/ijgi10070485.
[109] L. Shi, Y. Zhang, J. Cheng, H. Lu, Two-stream adaptive graph convolutional networks for skeleton-based action recognition, in: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 12018–12027, https://doi.org/10.1109/cvpr.2019.01230.
[110] X. Wang, H. Ji, C. Shi, B. Wang, P. Cui, P. Yu, Y. Ye, Heterogeneous Graph Attention Network, in: Proceedings of the World Wide Web Conference WWW 2019, 2019, Pp, 2022-2032, https://doi.org/10.1145/3308558.
[111] M. Schlichtkrull, T.N. Kipf, P. Bloem, R. van den Berg, I. Titov, M. Welling, Modeling relational data with graph convolutional networks, in: European Semantic Web Conference 2018, 2018, pp. 593–607, https://doi.org/10.1007/978-3-319-93417-4_38.
[112] D. Busbridge, D. Sherburn, P. Cavallo, N.Y. Hammerla, Relational Graph Attention Networks, arXiv preprint, arXiv:1904.05811, 2019, https://doi.org/10.48550/arXiv.1904.05811.
[113] M.R. Hosseini, J. Jupp, E. Papadonikolaki, T. Mumford, W. Joske, B. Nikmehr, Position paper: digital engineering and building information modelling in Australia, Smart and Sustainable Built Environment 10 (3) (2021) 331–344, https://doi.org/10.1108/sasbe-10-2020-0154.
[114] M.H. Rasmussen, M. Lefrançois, P. Pauwels, C.A. Hviid, J. Karlshøj, Managing interrelated project information in AEC knowledge graphs, Automation in Construction 108 (2019), 102956, https://doi.org/10.1016/j.autcon.2019.102956.
[115] L. Ehrlinger, W. Wöß, Towards a definition of knowledge graphs, in: Joint Proceedings of the Posters and Demos Track of 12th International Conference on Semantic Systems - SEMANTiCS2016 and 1st International Workshop on Semantic Change & Evolving Semantics (SuCCESS16), 2016. https://dblp.org/rec/conf/i-semantics/EhrlingerW16.html. accessed April 30 2023.
[116] C. Wu, P. Wu, J. Wang, R. Jiang, M. Chen, X. Wang, Ontological knowledge base for concrete bridge rehabilitation project management, Automation in Construction 121 (2021), 103428, https://doi.org/10.1016/j.autcon.2020.103428.
[117] X. Lei, P. Wu, J. Zhu, J. Wang, Ontology-based information integration: a state-of-the-art review in road asset management, Archives of Computational Methods in Engineering 29 (2021) 2601–2619, https://doi.org/10.1007/s11831-021-09668-6.
[118] D.-Y. Lee, H.-L. Chi, J. Wang, X. Wang, C.-S. Park, A linked data system framework for sharing construction defect information using ontologies and BIM environments, Automation in Construction 68 (2016) 102–113, https://doi.org/10.1016/j.autcon.2016.05.003.
[119] S. Ji, S. Pan, E. Cambria, P. Marttinen, P.S. Yu, A survey on knowledge graphs: representation, acquisition, and applications, IEEE Transact. Neural Networks Learn. Syst. 33 (2) (2021) 494–514, https://doi.org/10.1109/TNNLS.2021.3070843.
[120] C. Wu, X. Li, Y. Guo, J. Wang, Z. Ren, M. Wang, Z. Yang, Natural language processing for smart construction: current status and future directions, Automation in Construction 134 (2022), 104059, https://doi.org/10.1016/j.autcon.2021.104059.
[121] S. Khan, M.J. Awan, A generative design technique for exploring shape variations, Advanced Engineering Informatics 38 (2018) 712–724, https://doi.org/10.1016/j.aei.2018.10.005.
[122] W. Ma, X. Wang, J. Wang, X. Xiang, J. Sun, Generative design in building information modelling (BIM): approaches and requirements, Sensors 21 (16) (2021) 5439, https://doi.org/10.3390/s21165439.
[123] S. BuHamdan, A. Alwisy, A. Bouferguene, Generative systems in the architecture, engineering and construction industry: a systematic review and analysis, International Journal of Architectural Computing 19 (3) (2021) 226–249, https://doi.org/10.1177/1478077120934126.
[124] J. Zhang, N. Liu, S. Wang, Generative design and performance optimization of residential buildings based on parametric algorithm, Energy and Buildings 244 (2021), 111033, https://doi.org/10.1016/j.enbuild.2021.111033.
[125] A. Zabin, V.A. González, Y. Zou, R. Amor, Applications of machine learning to BIM: a systematic literature review, Advanced Engineering Informatics 51 (2022), 101474, https://doi.org/10.1016/j.aei.2021.101474.
[126] J.C.P. Cheng, W. Chen, K. Chen, Q. Wang, Data-driven predictive maintenance planning framework for MEP components based on BIM and IoT using machine learning algorithms, Automation in Construction 112 (2020), 103087, https://doi.org/10.1016/j.autcon.2020.103087.
[127] F. Zhang, A.P.C. Chan, A. Darko, Z. Chen, D. Li, Integrated applications of building information modeling and artificial intelligence techniques in the AEC/FM industry, Automation in Construction 139 (2022), 104289, https://doi.org/10.1016/j.autcon.2022.104289.
[128] A. Darko, A.P.C. Chan, M.A. Adabre, D.J. Edwards, M.R. Hosseini, E.E. Ameyaw, Artificial intelligence in the AEC industry: Scientometric analysis and visualization of research activities, Automation in Construction 112 (2020), 103081, https://doi.org/10.1016/j.autcon.2020.103081.
[129] N. Chileshe, PhD in construction management research: What is original contribution to knowledge? The case of TQM, in: Proceedings of the 21st Annual Association of Researchers in Construction Management (ARCOM) Conference, 2005, pp. 1267–1278, in: http://www.arcom.ac.uk/-docs/proceedings/ar2005-1267-1278_Chileshe.pdf. accessed October 20 2022.
[130] H. Boer, M. Holweg, M. Kilduff, M. Pagell, R. Schmenner, C. Voss, Making a meaningful contribution to theory, International Journal of Operations & Production Management 35 (9) (2015) 1231–1252, https://doi.org/10.1108/IJOPM-03-2015-0119.