# BIM-based construction quality assessment using Graph Neural Networks

**Navid Kayhani[1,2], Brenda McCabe[1], and Bharath Sankaran[2]**

[1]Department of Civil and Mineral Engineering, University of Toronto, Canada
[2]Naska.AI (formerly Scaled Robotics), Spain

navid.kayhani@mail.utoronto.ca, brenda.mccabe@utoronto.ca, bharath@naska.ai

**Abstract -**

**Automated construction quality control and as-built verification often involve comparing 3D point clouds captured on-site with as-designed Building Information Models (ad-BIM) at the individual element level. However, signal noise and occlusions, common in data captured from cluttered job sites, can negatively affect the performance of these methods that overlook the semantic relationships between elements. In this paper, we introduce a novel approach to automated quality control that enhances element-wise quality assessments by exploiting semantics in BIM. The proposed method represents ad-BIM as a graph by encoding elements' topological and spatial relationships. Exploiting this representation, we propose a Graph Neural Networks (GNNs)-based algorithm to infer element-wise built quality status. Our method significantly outperforms classical methods and allows for inference on partially observed or unobserved elements.**

**Keywords -**

**GNN; Quality Control; BIM; Semantic Enrichment; Point Cloud; Machine Learning.**

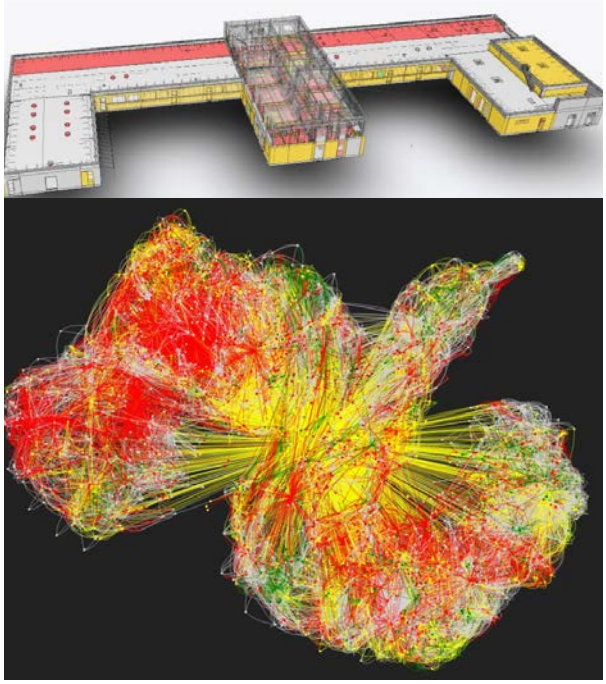

Figure 1. Graph representation of ad-BIM. The top shows the ad-BIM for an institutional building, with element-wise construction quality status labels. The bottom displays the same ad-BIM represented as a graph to leverage semantic relationships between elements to enhance as-built quality assessments.

## 1 Introduction

With the growing adoption of BIM within the construction industry, quality control processes are being automated to optimize cost and improve efficiency [1]. This is accomplished by comparing reality capture data (laser scanning, photogrammetry) to the as-designed BIM (ad-BIM) to detect errors in construction, thereby preventing costly downstream effects that subsequently affect the project cost and schedule [2].

Current approaches to automated quality control report element-wise quality where the as-built status of the building elements are evaluated in isolation [1, 3, 4] without considering the surrounding context. Irrespective of the reality capture system used, the data captured by these devices are plagued by noise and occlusions, which lead to inaccurate assessments. On cluttered construction sites, the reality capture data are affected by occlusions, sensor noise, weather conditions, and material properties, thereby resulting in the partial observability of the built structure (Figure 2). To cope with these sources of noise, it is essential to consider the semantic (e.g., spatial, topological, and temporal) relationships between building elements to enhance the quality control assessments.

In this paper, we present BIM-GNN, a novel method for detecting the built quality of building elements on a construction site using Graph Neural Networks (GNNs), incorporating semantic information extracted from ad-BIM. Our contributions are twofold. Firstly, we propose a method for constructing a graph representation of buildings from their ad-BIM (Figure 1), which is amenable to graph-based inference. In this representation, the nodes

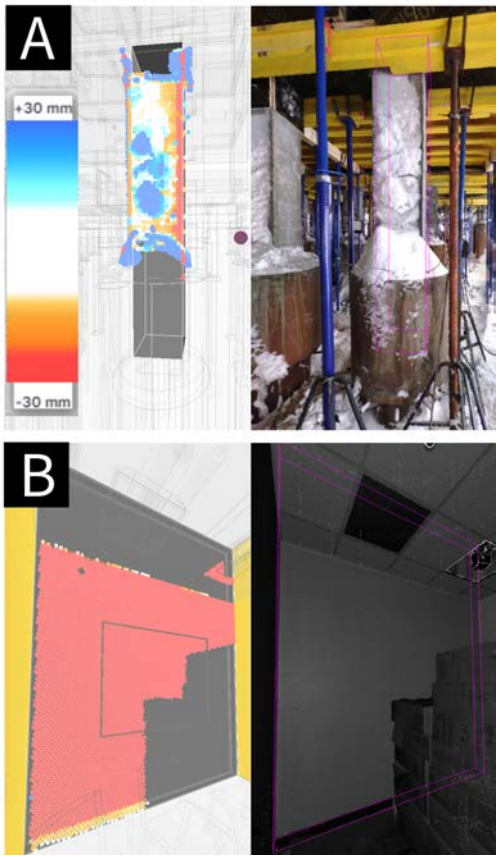


Figure 2. Noise and occlusion in on-site data. (A) Column covered by snow (noise) and occluded by ad-BIM elements (pedestal and steel beam). (B) Wall occluded by ad-BIM elements (ceiling) and clutter (boxes). Registered 3D point cloud data are shown as spheres indicating deviations as heatmaps.

represent BIM objects, and the edges represent their topological and spatial associations. Secondly, we propose a graph inference algorithm that utilizes the graph structure and the features computed on its nodes to classify the building elements using *Graph Attention Networks* [5] into one of four quality classes, namely, *verified* - element built within the desired tolerance, *deviated* - element built outside of desired tolerance, *missing* - element not built and *no data* - not enough information to make an assessment. Our experiments in Section 4 show that leveraging our graph representation allows our inference algorithm approach to significantly outperform existing element-based methods.

## 2 Background

In this section, we briefly review what most automated quality control techniques have in common and why we need to revisit the problem. Then, we provide a short overview of graph neural networks, graph representation of buildings, and the applications of GNNs in the architecture, engineering, and construction (AEC) domain.

### 2.1 Automated construction quality control

Automated quality control of construction projects is a well-studied area of research [6, 7]. Typically, solutions to this problem involve comparing the ad-BIM to 3D point clouds of the construction site captured with cameras [8] or laser scanners [7]. These comparisons can be done manually or using machine learning techniques. The machine learning-based solutions often treat the individual building elements as independent and identically distributed (iid) data which are fully observed. This is seldom the case, as the data are prone to noise and occlusions and the building elements are semantically linked. In this paper, we deal with the problem of partial observability by leveraging the graph structure and semantic relationships between elements of the ad-BIM to make robust quality control assessments of individual building elements.

### 2.2 Graphs and graph neural networks

As our ad-BIM is represented as a graph, we need learning algorithms that are capable of operating on non-euclidean permutation invariant data [9]. Most graph learning algorithms map the information the graph represents to a vector in a high-dimensional linear embedding space. These learned vectors can then be used for downstream learning tasks such as classification, regression, clustering, and generation. Graph Neural Networks (GNNs) typically use message-passing to update the representation of a node by processing its neighborhood's embedding from a previous layer to update the representation of the node in the current layer, as shown in Figure 3. The set of nodes that are directly connected to a given node via an edge is referred to as its neighborhood, which can include the node itself. In other words, GNNs with $l$ layers allow nodes to capture information within their $l$-hop neighborhood [10]. These networks take a graph as input and compute node embeddings through a series of non-linear transformations, allowing for prediction at the node, edge, or graph level. Many GNN architectures have been proposed that vary in their definitions of message, transformation, and aggregation operations. These architectures also differ in how they stack layers using different graph manipulation techniques to accomplish supervised or unsupervised tasks at the node or graph level.

### 2.3 Graph representation of BIM

Graph representation of buildings has been used for accessibility analysis [11], generative design [12], and large BIM file processing [13]. These representations capture

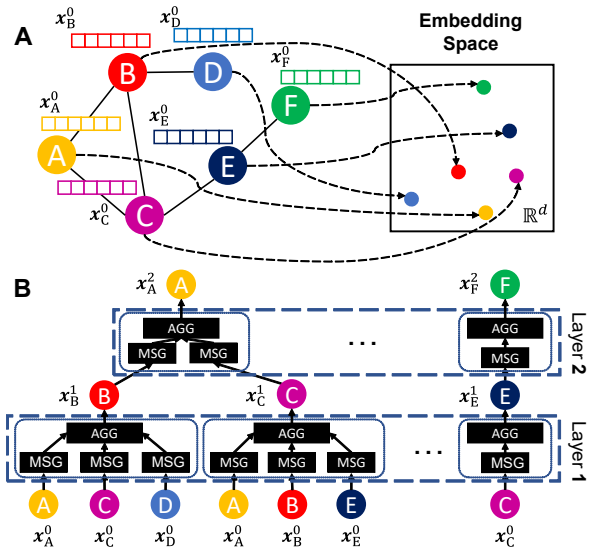


Figure 3. Graph neural networks (GNNs). (A) GNNs map information in the graph domain to a $d$-dimensional vector representation. (B) A 2-layer GNN example with a message-passing mechanism, where $\mathbf{x}_N^l$ indicates representation of node $N$ at layer $l$. Each node has its own computational graph, through which its representation is updated by transformation and aggregation of its neighborhood's representation. Here we only show nodes A and F. The transformed representation of the neighbor, self, and the connecting edge in layer $l-1$ constitute a message (MSG) in layer $l$. Every node aggregates (AGG) the messages it receives, using a permutation-invariant aggregation function (e.g., sum or average), and updates its representation.

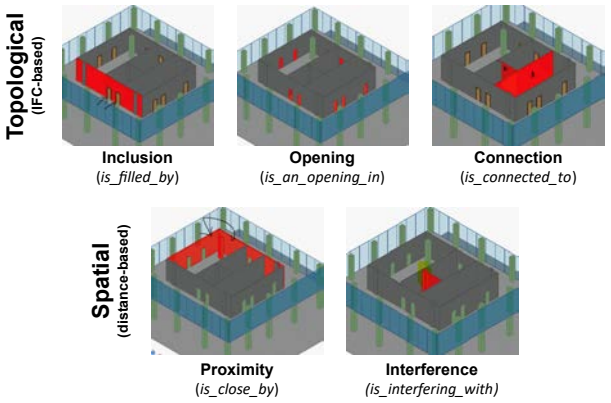


Figure 4. Topological and spatial relationships extracted from ad-BIM IFC.

geometries, semantics, and some spatial and topological relationships. They can be used to filter out irrelevant information, group related data, and identify key components. For instance, graphs were used to represent the relationships between IFC instances to enable topological querying, with semantic information being incorporated as the node and edge weights [14]. However, using these representations for learning-based quality control inference models has not been previously explored.

### 2.4 GNN in AEC domain

GNNs have seen limited use in the AEC domain but are gaining attention due to their potential applications [15]. For example, GNNs were used in conjunction with spatial vector data to classify patterns among groups of buildings for urban planning [16]. In architectural design, GNNs were used for the automated generation of floor plans that follow specific space planning rules [17]. In construction, these techniques were utilized to identify the most time-efficient construction sequence and to improve scheduling productivity and accuracy [18]. Finally, the automatic classification of room types [19] and the automation of the classification of BIM objects into different Industry Foundation Classes (IFC) [20] categories are two of the few examples of the use of BIM and GNNs.

## 3 Methods

In this section, we explain how we convert the ad-BIM IFC into a graph structure that can be used for automated machine learning-based quality control assessments. In Section 3.2, we describe the GNN model used for semantic-aware quality status classification of building elements.

### 3.1 Converting BIM to a graph

In this work, we consider the elements in ad-BIM as nodes and their associations with other elements as the edges of the graph. Two nodes are connected through an edge if they are topologically or spatially related. Incorporating temporal relationships is not considered in this paper. The topological and spatial relationships between BIM objects are extracted from ad-BIM in IFC format. The topological relationships are directly extractable from the hierarchical inheritance associations of objects in the IFC schema. Spatial relationships are determined by quantifying the distance between the various BIM objects' axis-aligned bounding boxes (AABB). We identified three types of topological relationships (Figure 4): (1) inclusion; (2) opening; (3) connection; and two types of spatial relationships: (1) proximity; (2) interference.

An inclusion relationship refers to the relationship between a container element (e.g., wall) and a filler element (e.g., door). For example, a wall instance of the *IfcWall*

class (e.g., *IfcWallStandard*) would be connected to an *IfcOpeningElement* through an *IfcRelVoidsElement*. A door in the same wall would be an instance of the *IfcDoor* class and would be connected to the same *IfcOpeningElement* through an *IfcRelFillsElement*. An opening relationship describes the association between a void and its container, which may or may not be filled by a filler element. A connection relationship is defined using *IfcRelConnectsElements*, which describes the elements' connectivity with a connection geometry (such as a point, curve, or surface). For example, walls A and B in Figure 4 would be connected through a surface and not through any other element.

Our proposed method for identifying spatial relationships between BIM objects calculates a distance matrix based on the minimum distance between the objects' AABB. The distance between two AABBs can be calculated using the coordinates of their bottom-left and top-right corners in 3D space. We then parametrize the proximity and interference relationships between objects. In the equation below, $d$ is the distance between the two BIM objects, $d_{cmin,cmax}$ are the proximity thresholds and $d_{int}$ is the interference threshold. Finally, semantics such as element's type and floor are extracted from the ad-BIM to build the ad-BIM graph $G = (V, E)$, where $V$ and $E$ refer to the set of its nodes and edges, respectively. Nodes in $V$ contain features extracted from ad-BIM, graph structure, and on-site data, while edges in $E$ have no attributes.

$$\text{spatial} = \begin{cases} d_{cmin} \leq d \leq d_{cmax} & \text{proximity} \\ d \leq \min\left(d_{int}, d_{cmin}\right) & \text{interference} \end{cases} \tag{1}$$

### 3.2 BIM-GNN Classifier model

#### 3.2.1 Architecture and training settings

Graph Attention Network (GAT) [5] is a GNN architecture that is built by stacking graph attention layers (GAT convolution). GAT convolutions use attention mechanisms to implicitly specify weights to different nodes in a neighborhood, indicating their importance. They may consist of multiple "heads", each with its own set of parameters. These heads attend to different aspects of the graph, allowing the model to learn multiple representations of the graph simultaneously. Mathematically, the attention coefficient between node $i$ and node $j$ is computed as:

$$\alpha_{i,j} = \frac{exp(LeakyReLU(a^T[\mathbf{W}h_i \parallel \mathbf{W}h_j])}{\sum_{k \in \mathcal{N}_i} exp(LeakyReLU(a^T[\mathbf{W}h_i \parallel \mathbf{W}h_k])} \tag{2}$$

where $\alpha_{i,j}$ is the attention coefficient between node $i$ and $j$, $a$ is a learnable parameter vector, $h_i$ and $h_j$ are the node representations for nodes $i$ and $j$, $\parallel$ is the concatenation operator, $\mathcal{N}_i$ is the set of all the neighbors of node $i$, and $W$ is a learnable weight matrix.

Finally, the new hidden representation of node $i$ ($h'_i$) in a GAT convolution with $K$ heads is calculated as:

$$h'_i = \parallel_{k=1}^{K} \sigma\left(\sum_{j \in \mathcal{N}_i} \alpha_{ij}^k \mathbf{W}^k h_j\right) \tag{3}$$

As depicted in Figure 5, our model has a sequential architecture and consists of two multi-headed GAT convolutions and a fully connected layer with ReLU activations. We apply dropout before each layer and to the normalized attention coefficients in GAT convolutions. The output is a four-dimensional array that is normalized using a log-softmax function. The training is performed following a transductive learning approach. In this approach, the model is trained on training labels while accessing the entire graph structure and node features. We used the negative log-likelihood as the loss, Adam optimizer with a learning rate of $5e{-}3$, and weight decay of $5e{-}4$. The model was trained for 10K epochs with early stopping.

#### 3.2.2 Baseline model

In this work, we use an ensembled baseline model ($ML{-}Vanilla$) to predict element-wise quality status. The baseline model is trained on features learned on point cloud data [21] and engineered features that relate to scan-vs-BIM coverage as discussed in Section 4.1.3. This baseline model $ML{-}Vanilla$ was trained on a large number of individual element data from multiple different construction projects yet without considering relationships between elements.

## 4 Experiments & Discussion

In our experiments, we are interested in answering the following questions:(1) How robust is our method on a dataset with a mixture of graphs with both highly imbalanced and balanced labels? (2) What are the impacts of the size of the training, validation, and test sets on the model performance? (3) How does the BIM-GNN model perform compared to the $ML{-}Vanilla$ model, which does not consider the element relationships and graph structure? (4) How do different feature types contribute to the model's expressiveness? (5) How can BIM-GNN help label unobserved or partially observed elements?

### 4.1 Dataset

#### 4.1.1 Description

The dataset used in this work contains ad-BIMs and scans captured from three institutional building projects in Europe, including two scans from a university building (UB1 and UB2), one scan from a hospital (LH), and one scan from a special school (SS) project. All scans were

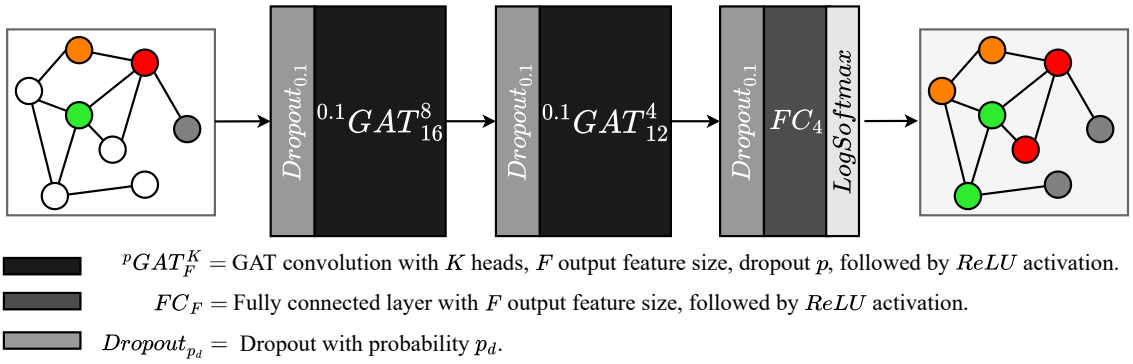


Figure 5. BIM-GNN Classifier architecture.

processed and registered with their ad-BIMs, such that only a subset of the elements of the ad-BIM are associated with the 3D point cloud. Then engineered and learned features for each element are computed solely based on the processed (partial) per-element scan of the on-site observations. An automatic ML-based process assigns each element a label based on the extracted features. A post-processing review is conducted to ensure the reliability of the assigned labels. We call a processed scan an analysis.

#### 4.1.2 Statistics

As summarized in Table 1, each analysis was converted to a graph with an average node count of 3103 and an average degree of 50. The edge counts varied between tens to hundreds of thousands. However, the relationships extracted from the ad-BIM turned out to be dominated by Spatial relationships across all analyses.

Figure 6 depicts the distribution of ground truth quality status labels for nodes in each graph. Imbalanced label distribution across all scans was observed where *No Data* accounts for a large proportion of the elements mainly due to data incompleteness in real-world projects.

#### 4.1.3 Features

Nodes in the created ad-BIM graphs contain three features: BIM-based, graph-based, and scan-based. BIM-based features include semantics such as element type and floor. Graph-based features capture information about a node's local neighborhood and include node degree, eigenvector centrality, and clustering coefficient. The node degree is the number of edges incident to that node. Eigenvector centrality measures the importance of a node in a graph. The clustering coefficient measures how the node's neighbors are connected to one another. Scan-based features include BaselineML and PointNet features. BaselineML is a set of hand-engineered features that quantify the degree to which the ad-BIM elements match the scans and include features such as scanned fraction, relative alignment error, etc. PointNet features are extracted from the hidden representation of the last two layers of a pre-trained Pointnet [21] model.

Table 1. Dataset summary

| Name | Nodes | Edges | Spatial Edges | Avg. Deg. |
|---|---|---|---|---|
| SS | 6264 | 242k | 99.7% | 77 |
| LH | 919 | 21k | 100% | 46 |
| UB1 | 3256 | 55k | 100% | 26 |
| UB2 | 2014 | 25k | 100% | 34 |

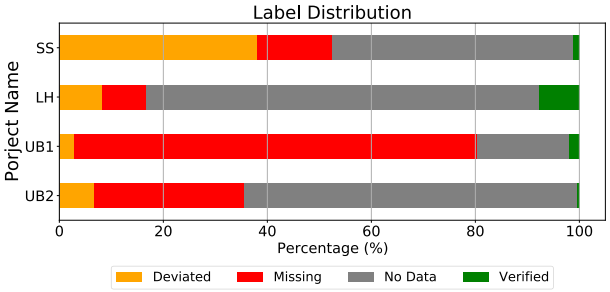


Figure 6. Distribution of the ground truth labels in the dataset.

### 4.2 BIM-GNN Classifier performance

In our experiments, we used nine randomized training/validation/test splits. We started our evaluations with a split of 60% training, 20% validation, and 20% test, considering all features defined in Section 4.1.3. We denote this combination of features and data split as the *original* dataset. We chose the F1-Score with weighted average as the metric for evaluation since it is suitable for evaluating both balanced and imbalanced datasets. Table 2 summarizes the performance of our model on the test sets

Table 2. Model performance

| Name | F1-Score (weighted) *ML−Vanilla* | F1-Score (weighted) **Ours** |
|---|---|---|
| SS | 61.53 ± 1.20% | **70.51** ±5.05% |
| LH | 62.17 ± 2.06% | **74.20** ±2.38% |
| UB1 | 75.41 ± 1.09% | **79.99** ±1.52% |
| UB2 | 22.26 ± 1.79% | **88.44** ±1.59% |
| Total | 55.34 ± 20.26% | **77.99** ±7.37% |

across 9 runs based on the metric above. The results suggest that our model outperforms the baseline significantly. The average F1-Score of *ML−Vanilla* is 55.34% with a standard deviation of 20.26%, while the average F1-Score of our model is 77.99% with a standard deviation of 7.37%. The average improvement is 22% with a lower variance. The improvement is more significant when the dataset is imbalanced. For example, the average F1-Score of *ML−Vanilla* on the UB2 graph is 22.26%, while the average F1-Score of our model is 88.44%.

Figure 9 depicts a low-dimensional representation of node logits pre- and post-training using t-distributed stochastic neighbor embedding (t-sne). T-sne maps high-dimensional data to a lower-dimensional space while preserving its structure. Figure 9 reveals separate clusters for each label. While our model can differentiate between *missing*, *no data*, and the other two classes, it struggles to differentiate between *deviated* and *verified* in certain cases. Further investigation is needed as mistakenly detecting *deviated* as *verified* may lead to unnoticed quality issues, although this is partly due to labeling subjectivity.

### 4.3 Influence of training data percentage

The *original* dataset allows the model to access 60% of the labels in the training set directly and 20% in validation indirectly, meaning if we have access to 80% of the labels, node features, and ad-BIM, we could outperform the baseline. However, to address the practicality of such a high percentage of available labels, we decreased the percentage of available labels (training 3: validation 1) and repeated the experiments. Results in Figure 7 show that even with 10% of the node labels (7% training), our proposed method achieved considerably higher F1-Score than the classical approach.

### 4.4 Ablation study

An ablation study is performed to investigate the importance of different features. We retrained our model by removing one feature or feature set at a time from the *original* dataset, namely, PointNet (*no_pnet*), type (*no_type*), and BaselineML (*no_bsln*) features, plus graph-based (*no_graph*) and scan-based (*no_bsln_pnet*)

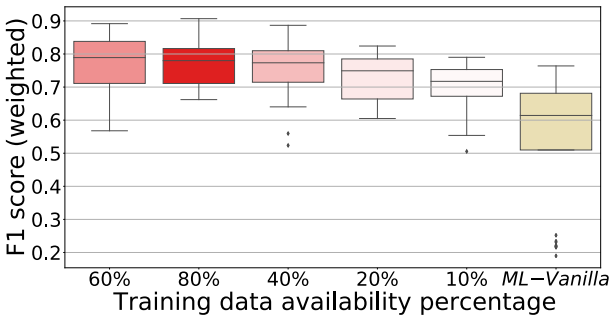


Figure 7. Impact of labeled data availability percentage during training on F1-Score and comparison with *ML−Vanilla*.

feature sets (Table 3). The results, shown in Figure 8, suggest that the PointNet features (learned scan-based features) do not positively contribute to the model's expressiveness, while hand-engineered BaselineML features seem to be more essential. Removing the type and the graph-based features can worsen the model's performance while removing the scan-based features has the most negative impact. The results indicate that all features, except for PointNet, contribute to the superior performance of our model. Nonetheless, we retained the PointNet features to ensure smoother performance, as a significant increase in F1-Score variance across various analyses is observed when excluded (*no_pnet*).

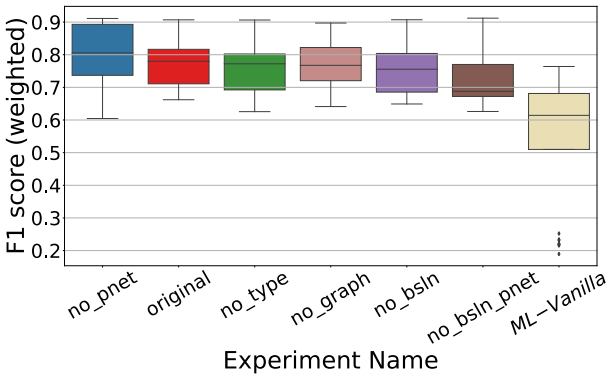


Figure 8. Impact of feature ablation on F1-score and comparison with *ML−Vanilla*.

More interestingly, comparing *ML−Vanilla* and *no_bsln_pnet* results suggests that our model can still outperform the baseline even without the scan-based features. This essentially means that given the graph structure and the labels for some elements (60% training), we can better predict the quality status of elements even if they are unobserved. This is a significant improvement on the baseline, which views the elements in isolation and solely relies

Table 3. Ablation study experiment settings

| | Scan-based | | BIM-based | | Graph-based |
|---|---|---|---|---|---|
| Name | BaselineML | PointNet | Type | Floor | Eigen, Deg., Cluster |
| *original* | ✓ | ✓ | ✓ | ✓ | ✓ |
| *no_pnet* | ✓ | | ✓ | ✓ | ✓ |
| *no_bsln* | | ✓ | ✓ | ✓ | ✓ |
| *no_type* | ✓ | ✓ | | ✓ | ✓ |
| *no_bsln_pnet* | | | ✓ | ✓ | ✓ |
| *no_graph* | ✓ | ✓ | ✓ | ✓ | |

on scan-based features. Our results suggest that incorporating semantics can improve element-wise quality status classification performance despite the partial observability of elements as an essential building block in construction quality assessment applications.

## 5 Conclusion and future work

In this work, we demonstrated the utility of exploiting the semantic (i.e., topological and spatial) relationships encoded within the ad-BIM graph to improve automated construction quality assessment despite partial observability of elements. To enhance our approach, we plan to explore additional graph learning algorithms and incorporate more relationship types to better capture the relationships between mechanical, electrical, and plumbing (MEP) elements. Additionally, we aim to add relationship types as edge features and further evaluate the generalizability of our method by applying it to more projects. We also hope to leverage the temporal relationships in 4D-BIM to predict element labels based on time- and sequence-dependent contexts.

## 6 Acknowledgements

This study was supported by Mitacs through the Mitacs Accelerate program and by the EU Horizon project, HumanTech, under GA 101058236. The first author is grateful to colleagues at Naska.AI (formerly Scaled Robotics SL) for their invaluable support. Special thanks go to Adarsh Jois and Alba Maria Herrera for their contributions and discussions. The first author thanks Sara Villuendas and CEO Stuart Maggs for their exceptional administrative support. We also thank the Natural Science and Engineering Research Council (NSERC) for their financial support through grant number RGPIN-2017-06792.

## References

[1] Burcu Akinci, Frank Boukamp, Chris Gordon, Daniel Huber, Catherine Lyons, and Kuhn Park. A formalism for utilization of sensor systems and integrated project models for active construction quality control. *Automation in construction*, 15(2):124–138, 2006. doi:10.1016/j.autcon.2005.01.008.

[2] James L Burati Jr, Jodi J Farrington, and William B Ledbetter. Causes of quality deviations in design and construction. *Journal of construction engineering and management*, 118(1):34–49, 1992. doi:10.1061/(ASCE)0733-9364(1992)118:1(34).

[3] LiJuan Chen and Hanbin Luo. A bim-based construction quality management model and its applications. *Automation in construction*, 46:64–73, 2014. doi:10.1016/j.autcon.2014.05.009.

[4] Stan Vincke and Maarten Vergauwen. Vison based metric for quality control by comparing built reality to bim. *Automation in Construction*, 144:104581, 2022. doi:10.1016/j.autcon.2022.104581.

[5] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph Attention Networks. *International Conference on Learning Representations*, 2018. URL `https://openreview.net/forum?id=rJXMpikCZ`.

[6] Thomas Czerniawski and Fernanda Leite. Automated digital modeling of existing buildings: A review of visual object recognition methods. *Automation in Construction*, 113:103131, 2020. doi:10.1016/j.autcon.2020.103131.

[7] Erzhuo Che, Jaehoon Jung, and Michael J Olsen. Object recognition, segmentation, and classification of mobile laser scanning point clouds: A state of the art review. *Sensors*, 19(4):810, 2019. doi:10.3390/s19040810.

[8] Navid Kayhani, Wenda Zhao, Brenda McCabe, and Angela P Schoellig. Tag-based visual-inertial localization of unmanned aerial vehicles in indoor construction environments using an on-manifold extended kalman filter. *Automation in Construction*, 135:104112, 2022. doi:10.1016/j.autcon.2021.104112.

[9] Michael M Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42, 2017. doi:10.1109/MSP.2017.2693418.

[10] William L Hamilton. Graph representation learning. *Synthesis Lectures on Artifical Intelligence and Machine Learning*, 14(3):1–159, 2020. doi:10.2200/S01045ED1V01Y202009AIM046.

[11] Nimalaprakasan Skandhakumar, Farzad Salim, Jason Reid, Robin Drogemuller, and Ed Dawson. Graph theory based representation of build-

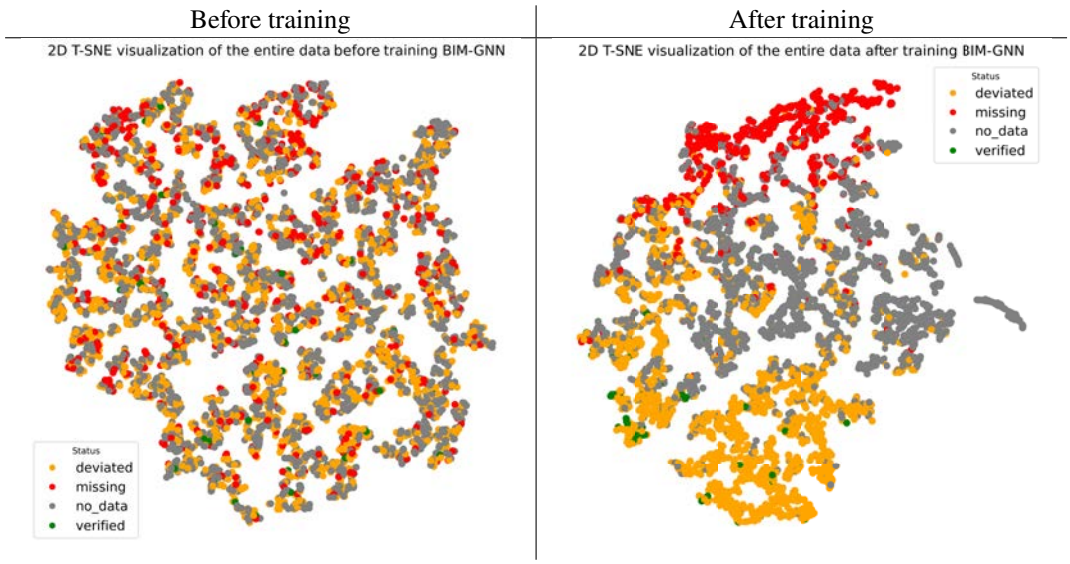


Figure 9. T-SNE embedding of the node-wise logits in SS graph.

ing information models for access control applications. *Automation in Construction*, 68:44–51, 2016. doi:10.1016/j.autcon.2016.04.001.

[12] Vincent JL Gan. Bim-based graph data model for automatic generative design of modular buildings. *Automation in Construction*, 134:104062, 2022. doi:10.1016/j.autcon.2021.104062.

[13] Xiaoping Zhou, Jichao Zhao, Jia Wang, Ming Guo, Jiayin Liu, and Honghong Shi. Towards product-level parallel computing of large-scale building information modeling data using graph theory. *Building and Environment*, 169:106558, 2020. doi:10.1016/j.buildenv.2019.106558.

[14] A Khalili and DK H Chua. Ifc-based graph data model for topological queries on building elements. *Journal of Computing in Civil Engineering*, 29 (3):04014046, 2015. doi:10.1061/(ASCE)CP.1943-5487.0000331.

[15] Yilong Jia, Jun Wang, M Reza Hosseini, and Wenchi Shou. Graph neural networks in building life cycle: a review. In *EC3 Conference 2022*, volume 3, pages 0–0. University of Turin, 2022. doi:10.35490/EC3.2022.164.

[16] Xiongfeng Yan, Tinghua Ai, Min Yang, and Hongmei Yin. A graph convolutional neural network for classification of building patterns using spatial vector data. *ISPRS Journal of Photogrammetry and Remote Sensing*, 150 (September 2018):259–273, 2019. ISSN 09242716. doi:10.1016/j.isprsjprs.2019.02.010.

[17] Ruizhen Hu, Zeyu Huang, Yuhan Tang, Oliver van Kaick, Hao Zhang, and Hui Huang. Graph2Plan: Learning Floorplan Generation from Layout Graphs. *ACM Transactions on Graphics*, 39(4):1–14, apr 2020. ISSN 0730-0301. doi:10.1145/3386569.3392391.

[18] Ying Hong, Haiyan Xie, Vahan Hovhannisyan, and Ioannis Brilakis. A graph-based approach for unpacking construction sequence analysis to evaluate schedules. *Advanced Engineering Informatics*, 52(May):101625, apr 2022. ISSN 14740346. doi:10.1016/j.aei.2022.101625.

[19] Zijian Wang, Rafael Sacks, and Timson Yeung. Exploring graph neural networks for semantic enrichment: Room type classification. *Automation in Construction*, 134(October 2021):104039, 2022. ISSN 09265805. doi:10.1016/j.autcon.2021.104039.

[20] Fiona C. Collins, Alexander Braun, Martin Ringsquandl, Daniel M. Hall, and André Borrmann. Assessing IFC classes with means of geometric deep learning on different graph encodings. In *Proceedings of the 2021 European Conference on Computing in Construction*, volume 2, pages 332–341, 2021. doi:10.35490/EC3.2021.168.

[21] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017. URL `http://arxiv.org/abs/1612.00593`.