2025 European Conference on Computing in Construction

CIB W78 Conference on IT in Construction

Porto, Portugal
July 14-17, 2025

GRAPH REPRESENTATION LEARNING: EMBEDDING MULTIMODALITY BIM MODELS INTO GRAPHS

Zijian Wang[1*], and Huaquan Ying[2*]
[1]TUM Georg Nemetschek Institute, Technical University of Munich, Munich, Germany
[2]Faculty of Civil and Environment Engineering, Technion Israel Institute of Technology, Israel
* These authors contributed equally to this work.

Abstract

Existing AI research in BIM often overlooks the multimodal nature of BIM data. This study proposes a graph representation learning approach to embed multimodal design data into high-dimensional vectors for supporting learning. Specifically, large text embedding models are utilized to encode object attributes, while geometries are embedded by a point cloud-based approach. The individual embeddings are concatenated and processed by GraphSAGE to leverage graph topology. An object classification experiment on a self-constructed graph dataset achieved 97.7% accuracy and an F1 score of 0.97 using attributes and graph topology. Future work will refine geometry embeddings and explore broader BIM applications.

Introduction

Machine learning has been extensively applied in the BIM domain. Most existing studies adopt an approach where researchers first select a specific machine learning algorithm and then process a subset of building information into the required data format to fit the algorithm. For instance, in the task of BIM object classification for semantic enrichment, Koo et al. (2019) employed Support Vector Machines (SVM) algorithm. Then they constructed a tabular dataset by extracting BIM object attributes such as width, height, and area, which was fed into the SVM model for training and classification.

Similarly, instead of using attribute data, Collins et al. (2021) processed object geometries by converting them into points to form mesh graphs, enabling the application of 3D deep learning techniques. Another study involved capturing virtual images of BIM objects using simulated cameras and utilizing them as input for a vision-based deep learning model, MVCNN, to classify object types and sub-types (Koo et al., 2021). This method achieved high accuracy and F1-scores of 0.98, as the rendered object images with clear backgrounds devoid of noise (Koo et al., 2021).

Beyond geometries and attributes, researchers have also explored the use of object relationships in classification tasks. Wang et al. (2022) investigated the effectiveness of leveraging topology for room type classification. They compiled BIM design models into graphs, where nodes represented rooms and edges captured various connection types. An improved graph neural network (GNN) trained on this graph dataset achieved an accuracy of approximately 80%, by only utilizing topology (Z. Wang et al., 2022).

Despite the success of these studies, they primarily focus on a single data modality of BIM models. However, BIM inherently consists of heterogeneous data, encompassing at least 1) 3D geometry that defines object shapes and locations, 2) object properties that provide descriptive attributes, and 3) object relationships that capture contextual semantics (Sacks et al., 2018). Some recent studies have developed AI systems that consume multiple types of BIM data, such as semantic properties and geometry (Utkucu et al., 2024). However, these systems are often designed for specific tasks and are not end-to-end learnable. They often employ ensemble learning mechanisms to integrate multiple subsystems, each specialized in processing a specific type of BIM data. As a result, this significantly restricts their generalization and scalability.

Integrating these multimodal data sources is essential for a more comprehensive representation of BIM models. Additionally, existing AI applications in BIM lack representation techniques that can embed multimodal building information effectively in a machine-learning way to facilitate further learning and diverse BIM applications.

However, successful advancements in various computer science domains highlight the crucial role of representation learning. It transforms raw symbolic data into structured numerical formats that capture underlying relationships and semantic meanings (Bengio et al., 2013). In other words, it converts data from a human-understandable format into a learning-suitable representation that is suitable for machines to learn. The effectiveness of representation learning can pave the way for downstream statistic pattern discovery, and prediction tasks.

For example, in the domain of Natural Language Processing (NLP), representation learning has been instrumental in developing models that understand and generate human language. A pioneering example is Word2Vec, which embeds text into dense high-dimensional vectors that capture semantic and syntactic relationships between words (Mikolov et al., 2013). This transformation from human-readable words to learning-suitable dense vectors has significantly contributed to the success of subsequent algorithms, allowing them to identify statistical patterns and improve performance in tasks such as translation and sentiment analysis.

Similarly, in computer vision, representation learning focuses on extracting meaningful information from images, facilitating tasks such as object recognition and image classification. A notable example is the encoding process in Convolutional Neural Networks (CNNs), where the initial layers transform raw pixel values into deep and compact feature representations (Krizhevsky et al., 2017; LeCun et al., 1998). The feature maps produced by the early CNN layers serve as powerful representations of the input image, enabling the model to effectively understand and recognize visual content across various applications.

In summary, AI studies in the BIM domain have predominantly followed an inverse research approach, where mature AI techniques are selected first, and building information is subsequently sampled and formatted to fit the algorithm's requirements. This approach often overlooks the inherent complexity, richness, and multimodality of building information. As a result, fundamental research questions related to applying AI to BIM are unaddressed: from identifying a suitable data format to store and integrate heterogeneous design data, developing representation learning techniques, and designing learning algorithms that can leverage these embeddings for domain-specific applications (Z. Wang et al., 2024).

In this study, we identify a critical research gap: **The lack of exploration into representation learning on BIM, which aims to process diverse BIM data into a high-dimensional vector space to support learning.**

To address this gap, we explore using graphs to integrate and embed building information. This study seeks to answer the following research question: **Can graph representation learning effectively embed complex and multimodal building information into high-dimensional vector spaces?**

To achieve that, we propose a multi-step method that embeds individual data modalities separately and subsequently integrates the embedding vectors. Additionally, a pilot experiment on object classification is conducted to evaluate feasibility. The experiment results and subsequent discussions are presented in the following sections.

Graph representation learning of BIM models

To enable AI in Building Information Modeling (BIM), it is essential to embed heterogeneous BIM data effectively. Given the inherent graph structure of BIM data, a graph-based embedding approach is proposed. The workflow of this approach is illustrated in Figure 1 and consists of three key steps:

(1) Graph representation of BIM data, where a simple graph data structure is designed to restructure BIM data into a uniform graph format suitable for AI applications;
(2) BIM object embedding, which embeds each single modality and then concatenates multi-modal vectors together;
(3) BIM object embedding updating, which updates the BIM object embeddings achieved in step (2) by considering graph topology using a graph learning-based approach.

As an initial exploration of effective graph representation learning for BIM, this study focuses on embedding BIM objects specifically within the architectural domain.

BIM graph representation

BIM provides an essential approach to generating and managing digital representations of the physical and functional characteristics of buildings and other physical assets. A real-world BIM model typically contains extensive and diverse data, often structured in an object-oriented manner based on well-defined data schemas. Due to the wide range of object types, properties, and relationships, BIM data schemas tend to be highly complex to ensure comprehensive coverage.

As the most widely used open and standardized BIM data schema, Industry Foundation Classes (IFC) provides a typical example. The latest official version (IFC 4.3 ADD2) contains 876 entities and 2,513 properties and continues to evolve. BIM object instances and properties following the IFC schema are often represented hierarchically and in a nested structure. Although these schemas provide an effective framework for data storage and retrieval, their complexity poses significant challenges for AI applications, particularly at the BIM model level.

In this study, we propose a simple, flexible, and, most importantly, AI-suitable data structure for BIM based on graph theory. Although prior efforts, such as IfcOWL (Pauwels & Terkaj 2016), have introduced graph-like representations using semantic web technologies, they largely preserve the structure of IFC schemas while merely altering the representation format, as their goal is to represent and integrate data, rather than supporting learning tasks. As a result, these approaches remain complex and are not well-suited for AI applications.

Our proposed graph structure is currently focused on the architectural domain and is defined as follows.

Each node represents a physical building element (e.g., *wall*, *slab*, *column*, and *roof*) or a key spatial element (e.g., *site*, *building*, and *story*). Node attributes consist of two components. The first component encompasses all semantic properties, which are simply organized as a dictionary of key-value pairs (e.g., <ClassName: value>, <PropertyName: value>) and/or sub-dictionaries. Unlike existing data schemas, our approach does not require a unified terminology for properties. Thanks to advanced text embedding techniques (see the **BIM graph node embedding** section), properties with different names but similar meanings can be effectively processed without strict standardization. This significantly simplifies the graph structure.

The second component refers to geometry data, which are uniformly represented as a triangle mesh, regardless of whether the geometry is a surface or a solid. This further reduces structural complexity.

With respect to the edge, we currently only consider spatial and topological relationships among architectural BIM objects. More specifically, four types of relationships are included: *belonging* (defines spatial relationships between physical and spatial elements), *host* (defines the topological relationship between openings and their host elements), *touch* (identifies when two physical elements are in direct geometric contact), and *intersect* (represents instances where two physical elements geometrically intersect). In short, we consider the building topology and organize them into graphs as edges.

To facilitate the construction of instance graphs, we have developed a dedicated IFC-to-BIMGraph converter, which transforms IFC-based BIM models into our proposed BIM graph structure.

BIM graph node embedding

To enable AI applications, each graph node—specifically its attributes—must be transformed from raw representations into high-dimensional vector embeddings. As discussed earlier, node attributes comprise two independent data modalities: semantic text and geometry, which require separate processing.

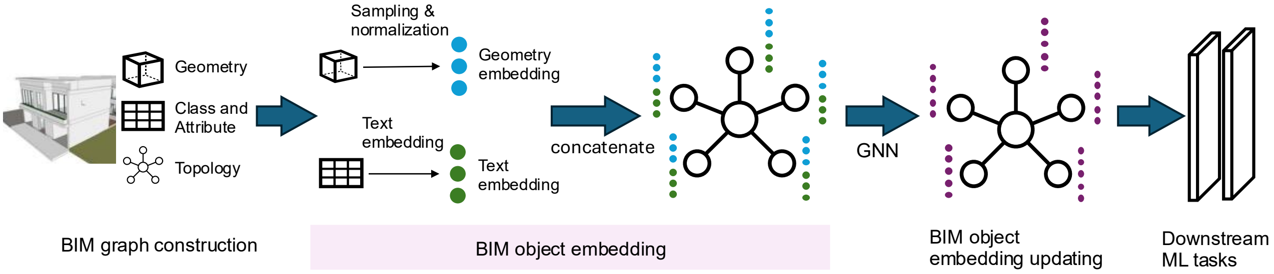

Figure 1: Graph representation learning (GRL) of BIM models

Semantic text embedding

We leverage text embedding techniques to convert semantic attributes into numerical vectors. In NLP, text embedding is a fundamental step for transforming raw text into vector representations, facilitating meaningful AI model training and processing (Patil et al., 2023). Various text embedding techniques exist and with the rise of large language models (LLMs), deep transformer-based embedding models trained on vast corpora have become powerful tools for generating context-aware, dynamic embeddings for varying-length text, including full sentences and paragraphs. These embeddings enable applications such as semantic search, classification, clustering, and recommendation systems. Ying & Sacks (2024) have utilized LLM embedding models to facilitate natural language-based querying of building design requirements.

For our study, we employ a compact version of OpenAI's embedding models—*text-embedding-3-small* (OpenAI, 2025)—as node attributes are generally short text strings. Additionally, we reduce the output vector length from the default 1,536 to 1,000, starting with a shorter version to test its effectiveness.

Geometry embedding

For geometric attributes, we propose a point cloud-based embedding method. The process consists of three key steps:

1) Point Sampling: We sample 1,024 points from the triangle mesh using the farthest point sampling (FPS) technique (Qi et al., 2017), ensuring a uniform distribution that preserves the geometric structure of the original shape.
2) Normalization: The 3D coordinates of sampled points are centralized and normalized to a unit sphere, which aids AI training while maintaining shape integrity.
3) Feature Preservation: To retain location and scale information, we concatenate the center point (3D coordinates) and scale factor (1D) with the normalized point cloud. After flattening all sampled points, the resulting geometry embedding vector has a length of 3,076.

Once both semantic and geometry embeddings are obtained, they are concatenated into a 4,076-dimensional vector representing the BIM object node. This unified representation enables efficient AI-driven processing of BIM data.

BIM graph node embedding updating

The BIM object embeddings obtained so far represent only individual embeddings, derived from each object's semantic properties and geometric features. However, these embeddings do not yet incorporate graph topology, which defines relationships between BIM objects through edges. The local neighborhood structure of each BIM object can provide valuable contextual information to enrich its embedding

To integrate this relational information, we apply the GraphSAGE model (Hamilton et al., 2017), a general and inductive GNN framework. GraphSAGE enhances node embeddings by aggregating features from neighboring nodes, allowing for a more contextualized representation of each BIM object.

In our context, we apply this algorithm for the task of BIM object classification. We consider a three-level neighborhood aggregation strategy, where the mean aggregator is used to compute the embedding of a node as the mean of its own features and those of its neighbors across multiple levels.

Experiment

An initial experiment was conducted to evaluate the feasibility of the proposed graph representation learning approach for BIM models, focusing on node classification. It is a fundamental task in BIM, as object classes provide essential semantic information that enhances interoperability and supports various applications (Belsky et al., 2016). Moreover, node classification is a starting point for GRL due to its simplicity and interpretability. It offers a clear evaluation of how well the proposed embeddings capture the underlying relationships and characteristics of BIM objects.

Dataset construction

A graph dataset was constructed using five BIM models created with different software and covering various building scenarios, as summarized in Table 1.

Table 1: Statistic of the constructed dataset

No	Scenario	Software	#Node	#Edge
1	School	Revit	576	762
2	Residential	ArchiCAD	15,676	119,144
3	House	ArchiCAD	360	2,357
4	Hospital	Revit	3,256	9,948
5	Conference center	Revit	1,215	5,170

The dataset consists of 15 IFC object classes, 21,083 nodes, and 137,381 edges. All models were stored in the Industry Foundation Classes (IFC) format to facilitate interoperability. A graph compiler, developed using Python and IfcOpenShell (2024), was employed to extract building information from IFC files and organize it into graph structures. An example graph and its corresponding model are shown in Figure 2.

This compiler systematically extracts building objects categorized under the *IfcObject* class, along with their associated attributes and relationships.

In terms of attributes, all available properties associated with each building object were extracted, covering material properties, dimensions, and functional definitions. Additionally, user-defined property sets were extracted to capture custom data. The extracted attributes were stored in CSV format for subsequent processing.

For relationships, the dataset includes relationships extracted from the IFC schema and enriched relationships by geometric computations. The *IfcRelContainedInSpatialStructure* relationship, which defines the spatial hierarchy among sites, buildings, storeys, and their contained objects, was extracted and regarded as *belonging* edges in the graph. Hosting relationships were identified using *IfcOpeningElement*, capturing dependencies where objects are hosted by others. To further enrich the graph structure, spatial relationships such as *touch* and *intersect* were introduced through geometric computations.

For geometric data, the geometry of every solid object was extracted and saved as separate OBJ files. However, abstract objects such as *IfcBuilding, IfcSite,* and *IfcStorey* do not inherently contain geometric information. To ensure compatibility with the geometry embedding method, synthetic geometries were generated for these classes. Specifically, a bounding box was created for *IfcBuilding* to encompass all contained objects, a horizontal plane was generated at the corresponding altitude for *IfcStorey,* and a plane was generated at the site's altitude for *IfcSite.*

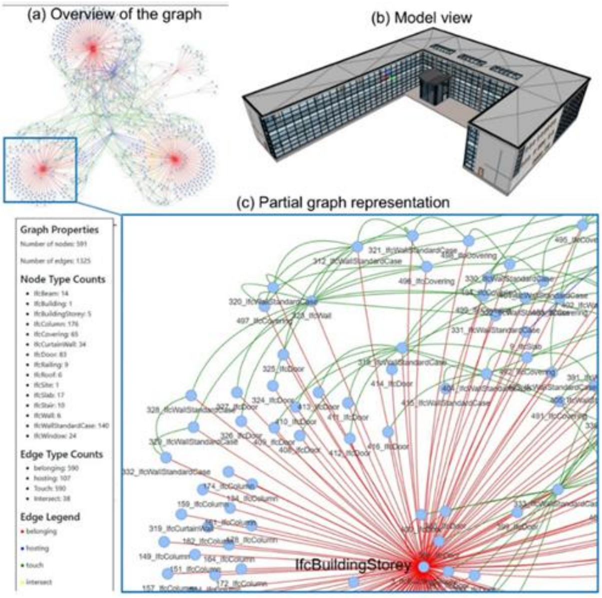

Figure 2: Examples of examples and corresponding graph representation

To ensure label accuracy, a manual verification process was conducted by visually inspecting the original geometries. It was observed that object types were frequently misclassified in the original models. For example, walls were sometimes modeled as columns, and slabs when positioning them horizontally. In the No.2 residential building, 566 objects, including walls, columns, covers, and windows, were relabeled to enhance data accuracy. This manual refinement improves the reliability of the dataset for subsequent learning tasks.

Experiment plan

The experiment is organized into two groups to evaluate the contributions of embeddings and graph topology in the proposed GRL approach, as shown in Table 2.

The first group (No.1 to No.3) focuses on assessing the effectiveness of different types of node embeddings while utilizing all available edge connections. Specifically, Experiment No.1 combines attribute-based embeddings and geometry-based embeddings by concatenating the two feature types directly. Experiment No.2 relies solely on geometric embeddings to analyze the impact of shape-related features on representation learning, while No.3 evaluates the effectiveness of text-based embeddings alone.

The second group (No.4 to No.7) investigates the influence of graph topology by varying the types of edge connections while excluding node embeddings. In this configuration, only the presence or absence of connections between nodes is considered, without leveraging edge features such as edge classes. Experiment No.4 evaluates a fully connected graph incorporating all edge types, whereas Experiments No.5 to No.7 selectively adopt subsets of edge connections to analyze their contributions to performance.

Training and evaluation

During implementation, we utilize the Deep Graph Library (DGL) (M. Wang et al., 2019). We employ a three-layer GraphSAGE architecture (Hamilton et al., 2017) with a hidden size of 64, which enables the model to capture both local and higher-order neighborhood information effectively. Dropout regularization with a rate of 0.2 is applied to mitigate overfitting.

To enhance computational efficiency and scalability, mini-batch training is adopted, allowing the model to process subsets of nodes at each step rather than the entire graph at once. During the training, each graph is randomly split into training, validation, and test sets with a ratio of 60%, 20%, and 20%, respectively.

For evaluation, accuracy and F1-score are selected as the primary performance metrics. Accuracy, a widely used metric in classification tasks, measures the proportion of correctly classified instances relative to the total number of instances F1-score; on the other hand, it provides a balanced evaluation by considering both precision and recall, making it particularly useful in cases of imbalanced class distributions.

Table 2: Experiment results of graph representation learning on BIM

No	Node Embeddings	Edge connections*	Accuracy	F1 score
1	Attribute, geometry	All (belonging, host, touch, intersect)	86.6%	0.83
2	Geometry	All	58.8%	0.57
3	Attribute	All	97.7%	0.97
4	-	All	57.9%	0.53
5	-	Belonging	47.6%	0.33
6	-	Belonging, host	53.1%	0.42
7	-	Belonging, touch, intersect	57.7%	0.52

* Types of edges are used to construct graphs.

Results

The experimental results are presented in Table 2. The first group of experiments (No.1 to No.3) explores the impact of different embeddings. The results demonstrate that attribute-based embeddings (No.3) yield the highest accuracy of 97.7% and an F1-score of 0.97, indicating that using large text embedding models is effective for processing object attributes as embeddings.

In contrast, the concatenation of attribute and geometry embeddings (No.1) achieves an accuracy of 86.6% and an F1-score of 0.83, which is 11% lower than using attribute embeddings alone. Additionally, Experiment No.2 with only geometry embeddings resulted in significantly lower accuracy (58.8%) and F1-score (0.57). The results from No.1 and No.2 illustrate that the geometry embeddings do not contribute to performance or even decrease accuracy.

The second group (No.4 to No.7) investigates the influence of different edge connections without node embeddings. Experiment No.4, which considers all edge types, achieves an accuracy of 57.9% and an F1-score of 0.53, showing that relational information alone, even without node attributes, can still capture meaningful semantics. This demonstrates the potential of graphs to encode spatial and contextual information inherent in BIM data.

Overall, the results suggest that attribute embeddings provide the most significant contribution to classification accuracy, while the geometric embeddings could decrease the performance. It means that the geometry embedding technique is not effective and needs to be further improved. On the other hand, graph topology alone, without node embeddings, offers a strong baseline, making graph representation a viable approach for BIM analysis tasks where other modality data may be incomplete or unavailable.

Compare with other machine learning algorithms

The comparison results between the GRL approach and selected machine learning algorithms are presented in Table 3. All comparisons are conducted under the same dataset division and the same embeddings. The results demonstrate that the accuracy and F1-scores of GraphSAGE are closely aligned with those of traditional machine learning models, such as decision trees and random forests. Notably, the training time of GraphSAGE is comparable to or even shorter than some traditional models.

The findings also validate the effectiveness of using large text embedding models to embed BIM text-based attributes, as different algorithms consistently achieve higher accuracy and F1 scores when using geometry embeddings. The relatively lower performance of geometry-based embeddings indicates that further improvements are needed.

The classification performance of GRL is close but not beyond other machine learning methods. This may be caused by the simplicity of the current experimental task, which includes a limited dataset and only 15 object classes. We believe that the potential of graph representation learning is not yet fully realized, especially after considering topology and relationships.

Table 3: Comparison with other machine learning methods

Embedding	Model	Accuracy	F1 score	Training time (s)
Attribute, geometry	Decision tree	97.3%	0.97	29
	GraphSAGE	86.6%	0.83	38
	KNN	83.9%	0.82	1
	Random forest	**98.7%**	0.99	37
	SVM	95.9%	0.96	211
Geometry	Decision tree	**66.5%**	0.65	38
	GraphSAGE	58.8%	0.57	32
	KNN	35.5%	0.29	1
	Random forest	62.0%	0.50	48
	SVM	53.8%	0.43	375
Attribute	Decision tree	97.1%	0.97	3
	GraphSAGE	97.7%	0.97	32
	KNN	98.4%	0.98	1
	Random forest	**98.4%**	0.98	8
	SVM	98.0%	0.98	4

Conclusions

Strength and weakness

In this study, we propose a novel graph-based method for embedding multi-modal BIM data into a high-dimensional vector space that can enable various machine learning algorithms tasks. Our approach introduces a property graph-based data structure to reorganize and store IFC-based BIM data in a relatively straightforward manner, which facilitates the implementation of machine learning algorithms.

Specifically, we define physical building objects and key spatial structure elements—including *site*, *building*, and *story*—as nodes in the property graph, with relevant attributes attached. These attributes are further categorized into two types: semantic properties and geometric data. Additionally, we define four topological relationships—*belonging*, *host*, *touch*, and *intersect*—as graph edges to capture spatial and structural connections between objects.

We employ modality-suitable techniques to embed the nodes and edges of the BIM property graph. Specifically, we utilize large text embedding models for semantic properties. We developed a point cloud-based method to embed geometry data, which has the feature of reserving shapes, dimensions, and location information. The embeddings from attributes and geometry are concatenated directly. Furthermore, we apply graph learning algorithms to refine BIM object node embeddings by incorporating edge relationships.

Regarding the research question, we evaluated the proposed graph-based BIM embedding method through a BIM object classification task. The experimental results validated the feasibility of using graphs to leverage multi-modal BIM data (attributes, building topology) efficiently as embeddings for machine learning applications.

However, our findings also indicate areas for improvement. While semantic properties are well embedded, the geometry embedding technique requires further refinement. Additionally, the study has several limitations: (1) the evaluation dataset is relatively small, compiled from only five BIM models; and (2) the classification task is simplistic, involving only a few object types with distinct characteristics. These factors may limit the validity of our results in assessing the effectiveness of individual embedding components and the overall graph node representation. Moreover, the potential advantages of the graph-based approach may be underestimated due to the simplicity of the evaluation task, where graph topology could not be fully leveraged.

Future work

To further improve the proposed graph-based BIM embedding method, our future work will focus on the following four key aspects:

Improving geometry embedding techniques. The current method largely relies on raw geometry data as input features, which has proven ineffective for machine learning tasks where geometric information is critical. Future work will explore learning-based methods, such as PointNet (Qi et al., 2017) and MeshNet (Feng et al., 2019), to generate learning-based geometric embeddings.

Enhancing the fusion of semantic and geometric embeddings. Currently, semantic and geometric embeddings are combined through simple vector concatenation, assuming equal importance and perfect alignment across tasks—an assumption that may not always hold. To address this, we will investigate attention mechanisms (Vaswani et al., 2017) to enable dynamic and adaptive fusion of multimodal embeddings based on task-specific needs.

Exploring advanced graph representation learning methods. At present, we have only examined GraphSAGE, which focuses on graph topology while disregarding the semantic properties of edges. In addition, the algorithm works in a supervised learning mode, which makes the result embeddings often suitable to specific tasks designed for training and lacks generality. Future work will explore more advanced graph learning algorithms capable of incorporating edge attributes to generate edge-ware node embeddings in an unsupervised manner, thereby improving the quality and generality of the representation of BIM data.

Establishing large-scale and comprehensive datasets. This experiment only compiles 5 real-world models, where the size of the overall dataset is still comparatively small. We believe that constructing a large dataset can support various meaningful tasks within the BIM domain to enable thorough and rigorous evaluation.

Acknowledgments

The author Zijian Wang received post-doc funding from TUM Georg Nemetschek Institute at the Technical University of Munich. These authors contributed equally to this work.

References

Belsky, M., Sacks, R., & Brilakis, I. (2016). Semantic Enrichment for Building Information Modeling. *Computer-Aided Civil and Infrastructure Engineering*, *31*(4), 261–274. https://doi.org/10.1111/mice.12128

Bengio, Y., Courville, A., & Vincent, P. (2013). Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *35*(8), 1798–1828. https://doi.org/10.1109/TPAMI.2013.50

Collins, F. C., Braun, A., Ringsquandl, M., Hall, D. M., & Borrmann, A. (2021). Assessing IFC classes with means of geometric deep learning on different graph encodings. *Proceedings of the 2021 European Conference on Computing in Construction*, *2*, 332–341. https://doi.org/10.35490/ec3.2021.168

Feng, Y., Feng, Y., You, H., Zhao, X. & Gao, Y. (2019). Meshnet: Mesh neural network for 3d shape representation. *In Proceedings of the AAAI conference on artificial intelligence*, 33(1), pp. 8279-8286. https://doi.org/10.1609/aaai.v33i01.33018279

Hamilton, W. L., Ying, R., & Leskovec, J. (2017). Inductive representation learning on large graphs. *Advances in Neural Information Processing Systems*, *2017-Decem*, 1025–1035. http://arxiv.org/abs/1706.02216

IfcOpenShell. (2024). *IfcOpenShell*. https://ifcopenshell.org/

Koo, B., Jung, R., Yu, Y., & Kim, I. (2021). A geometric deep learning approach for checking element-to-entity mappings in infrastructure building information models. *Journal of Computational Design and Engineering*, *8*(1), 239–250. https://doi.org/10.1093/jcde/qwaa075

Koo, B., La, S., Cho, N. W., & Yu, Y. (2019). Using support vector machines to classify building elements for checking the semantic integrity of building information models. *Automation in Construction*, *98*, 183–194. https://doi.org/10.1016/j.autcon.2018.11.015

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2017). ImageNet classification with deep convolutional neural networks. *Communications of the ACM*, *60*(6), 84–90. https://doi.org/10.1145/3065386

LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278–2323. https://doi.org/10.1109/5.726791

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. *1st International Conference on Learning Representations, ICLR 2013 - Workshop Track Proceedings*, 1–12. http://arxiv.org/abs/1301.3781

OpenAI (2025). Vector embeddings. https://platform.openai.com/docs/guides/embeddings (Last accessed Jan. 30th, 2025)

Pauwels, P. & Terkaj, W. (2016). EXPRESS to OWL for construction industry: Towards a recommendable and usable ifcOWL ontology. *Automation in construction*, 63, 100-133. https://doi.org/10.1016/j.autcon.2015.12.003

Patil, R., Boit, S., Gudivada, V. & Nandigam, J. (2023). A survey of text representation and embedding techniques in nlp. *IEEE Access*, 11, 36120-36146. https://doi.org/10.1109/ACCESS.2023.3266377

Qi, C.R., Su, H., Mo, K.& Guibas, L.J. (2017). Pointnet: Deep learning on point sets for 3d classification and segmentation. *Proceedings of the IEEE conference on computer vision and pattern recognition 2017,* pp. 652-https://doi.org/660. 10.1109/CVPR.2017.16

Sacks, R., Eastman, C., Lee, G., & Teicholz, P. (2018). BIM Handbook. In *John Wiley & Sons*. John Wiley & Sons, Inc. https://doi.org/10.1002/9781119287568

Utkucu, D., Ying, H., Wang, Z. & Sacks, R. (2024). Classification of architectural and MEP BIM objects for building performance evaluation. *Advanced Engineering Informatics*, 61, p.102503. https://doi.org/10.1016/j.aei.2024.102503

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A., Kaiser, Ł.& Polosukhin, I. (2017). Attention is All you Need. *Advances in Neural Information Processing Systems*. 30. https://arxiv.org/abs/1706.03762

Wang, M., Zheng, D., Ye, Z., Gan, Q., Li, M., Song, X., Zhou, J., Ma, C., Yu, L., Gai, Y., Xiao, T., He, T., Karypis, G., Li, J., & Zhang, Z. (2019). *Deep Graph Library: A Graph-Centric, Highly-Performant Package for Graph Neural Networks*. 1–18. http://arxiv.org/abs/1909.01315

Wang, Z., Sacks, R., & Yeung, T. (2022). Exploring graph neural networks for semantic enrichment: Room type classification. *Automation in Construction*, *134*(June 2021), 104039. https://doi.org/10.1016/j.autcon.2021.104039

Wang, Z., Ying, H., & Sacks, R. (2024). Two Fundamental Questions Concerning Representation for Machine Learning BIM Data. *The 41st International Conference of CIB W78*. http://itc.scix.net/paper/w78-2024-25

Ying, H. & Sacks, R. (2024). From Automatic to Autonomous: a Large Language Model-driven Approach for Generic Building Compliance Checking. *The 41st International Conference of CIB W78.* https://easychair.org/publications/preprint/k2PJ