
Two Fundamental Questions Concerning BIM Data Representation for Machine Learning

Zijian Wang, zijian.wang@campus.technion.ac.il

Faculty of Civil and Environmental Engineering, Technion Israel Institute of Technology, Israel

Huaquan Ying, enochying@campus.technion.ac.il

Faculty of Civil and Environmental Engineering, Technion Israel Institute of Technology, Israel

Rafael Sacks, cvsacks@technion.ac.il

Faculty of Civil and Environmental Engineering, Technion Israel Institute of Technology, Israel

Abstract

Successful applications of Artificial intelligence (AI) highlight the importance of representing domain-specific data in formats suitable for learning. Researchers in the BIM domain often adopt an inverse approach by selecting AI techniques first and then tailoring BIM information into specific formats, which results in incomplete information and limited application scenarios. We argue that BIM-specific data representations and learning techniques are crucial to leveraging the full richness and scope of the information in BIM models for AI applications. Consequently, this article poses two fundamental questions: 1) **What formats are most suitable for BIM data representation?** 2) **What are the corresponding learning techniques needed for BIM?** To begin the exploration of the first question, we propose a graph-based approach to represent design data for storage and computation. Through an object classification experiment, we demonstrate two AI algorithms achieve an accuracy of 95% and an F1 score of 0.95 after incorporating graph-related features.

Keywords: Artificial Intelligence, Building Information Modelling, graph representation, graph learning, machine learning.


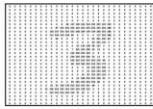








1 Introduction

The development of AI and its achievements have attracted attention all around the world. In the computer vision domain, digital images are usually stored in a three-dimensional matrix, providing a structured data format that can be efficiently manipulated by matrix operations. The initial Convolutional Neural Networks (CNNs), which aimed to simulate the function of neural cells of humans, were designed to process the representation of images in a matrix (Krizhevsky et al., 2012). In 2015, ResNet, a deep learning model, achieved a top-5 error rate of 3.57%, which is lower than the estimated human error rate of around 5% for the same task in the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) (He et al., 2016). For the first time, a deep learning algorithm surpassed human performance on the object classification task (He et al., 2015). This was a milestone in AI history, showing that accumulated large datasets with proper design algorithms can outperform humans in this specific capability.

Recently, large language models (LLMs), such as GPT and Gemini, have demonstrated their capability to handle varied and complex language tasks. This breakthrough in LLMs, exemplified by GPT-3.5, resulted from the continuous expansion of dataset sizes and improvements in training mechanisms (OpenAI, 2022). However, it is essential not to overlook the foundational works in the natural language processing (NLP) domain that paved the path for today's successes. An early fundamental work is Word2Vector, which represents words as dense, numerical vectors that capture semantic and syntactic relationships between words (Mikolov, Chen, et al., 2013). This addresses the challenge of generating embeddings, enabling computers to understand and

process language more effectively (Mikolov, Sutskever, et al., 2013). Additionally, another seminal work is the Transformer model, a novel deep learning architecture, which introduces the attention mechanism to allow parallel computation but capture long-range dependencies and context effectively (Vaswani et al., 2017). Transformers have profoundly advanced the field of NLP, leading to the development of state-of-the-art LLMs, which have also been applied in other domains, such as computer vision.

Reflecting on the development of AI in different domains (Figure 1), there is a route involving data representation from a machine-readable format to a learning-suitable one, such as processing languages to tokens, and audio to spectrograms. Each domain also develops specific learning algorithms that are designed for the data representation, such as CNNs for images and transformers for languages.

Domain	Machine-readable data	Learning-suitable representation*	Example techniques														
Computer Vision	 Image	 Matrix	Convolutional neural network (CNN)														
Natural Language Processing	 Text	<table border="1" data-bbox="833 808 1115 857"> <tr> <td>[CLS]</td> <td>This</td> <td>is</td> <td>an</td> <td>input</td> <td>text</td> <td>[SEP]</td> </tr> <tr> <td>101</td> <td>2023</td> <td>2000</td> <td>2019</td> <td>7953</td> <td>3793</td> <td>102</td> </tr> </table> Tokens	[CLS]	This	is	an	input	text	[SEP]	101	2023	2000	2019	7953	3793	102	Transformers
[CLS]	This	is	an	input	text	[SEP]											
101	2023	2000	2019	7953	3793	102											
Audio Processing	 Audio	 Spectrogram	Hidden Markov Models (HMMs)														
Networks	 Networks	 Graphs	Graph Neural Networks (GNNs)														
Building Information Modelling	 BIM models in files or database																

* The computer vision images are adopted from Boroojerdi and Rudolph (2022). We use the BERT tokenizer (Devlin et al. 2019) to generate the token example. The audio processing images are adopted from OpenSeq2Seq (2024).

Figure 1. Data representation in various domains

All these exciting achievements inspire BIM researchers. However, we observe that most, if not all, BIM machine learning studies adopt an inverse approach by first selecting successful algorithms and then processing information into a specific format. For example, one might first select a decision tree algorithm and then parse object attributes into a suitable table. The main problem of this approach is that the diversity of building information is ignored, meaning that the processed information often fails to represent the richness and full meaning of design data. Additionally, these applications often result in a situation of specific purposes and narrow scopes, as we assume that techniques for processing images or point cloud data are the desired ones instead of considering developing techniques that can fit the specific characteristics of BIM.

BIM data are complex. First, even a simple BIM model contains varied data types, such as objects, relationships, and geometries. There is a need to integrate diverse modalities. Additionally, BIM data are often stored in software databases with limited access or in open schema files suitable for collaboration. Moreover, in the construction industry, the culture tends to protect work rather than share it with the community, resulting in limited open resources for BIM models. Luckily, there are increasingly more open datasets and design models available from recent studies. In short, the diversity of design information, the data accessibility issues, and the limited availability of data pose challenges in representing building information and leveraging learning.

Therefore, we argue that new data representation and new learning techniques are essential to leverage the full richness and scope of the information in BIM models for machine learning applications. This raises two fundamental questions:

1) What formats are most suitable for BIM data representation?

2) What are the corresponding learning techniques needed for BIM?

We raise these two questions, although we cannot yet provide comprehensive answers. Instead, in this paper we explore a potential approach that adopts graphs to represent and leverage multi-modal building information for learning, which begins to address the first question.

2 Background

2.1 Machine learning in BIM

Machine learning techniques have been widely applied to the BIM domain. However, most existing studies adopt an inverse approach. Researchers first identify a specific machine learning algorithm that performs well, and then process part of the building information into the required data format to launch the algorithm. For example, in the semantic enrichment domain for BIM object classification, researchers initially used traditional machine learning algorithms, such as Support Vector Machines (SVM) (Koo & Shin, 2018). To generate the required tabular data for SVM, researchers first extracted attributes from IFC objects, such as area and volume, as features for each object to form a row, and then manually labeled the class of each piece of data as the ground truth.

Soon, more complex and advanced machine learning and deep learning techniques were explored. To launch the vision-based MVCNN algorithm, researchers set up virtual cameras to take images of a BIM object from 12 different cardinal directions (Koo, Jung, & Yu, 2021). Similarly, the performance of point-based deep learning models captured the attention of researchers. Subsequently, the geometry of BIM objects was parsed as point clouds, which were fed into PointNet for classification (Koo, Jung, Yu, et al., 2021). Alternatively, the geometry was processed as graphs to serve GNNs (Collins et al., 2021). Another example validating the inverse approach is the testing of GNNs on BIM graphs, where researchers even manually labeled object relationships in a matrix and converted it to graphs for testing the emerging deep learning techniques (Wang et al., 2022).

The inverse approach results in a relatively narrow application scope. More importantly, processing design data into a specific format preserves part of the information while abandoning other semantics. Tables can store attributes but struggle to represent relationships and geometries. Images retain vision features, but other details are hard to preserve. Graphs can maintain relationships and object attributes, but geometry and vision features are difficult to incorporate. Some studies explore leveraging different types of building information to improve classification performance using ensemble learning techniques (Utkucu et al., 2024). However, all of these efforts still follow the inverse approach by focusing on algorithms that handle specific types of information, such as geometry and vision, and then combining them. This method still prioritizes the selection of algorithms over the initial proper representation of design data. A comprehensive representation of the full richness and scope of information contained in BIM models for machine learning applications is still lacking.

2.2 Graph representation

As a BIM project progresses through its lifecycle, it generates various types of data. For many years, the linked building data (LBD) community has researched application of semantic web techniques to integrate and query building information (Pauwels, 2021). Building information is represented in a graph format by following designed ontologies, and query techniques, such as SPARQL, are used to retrieve information from these graphs to support applications (Pauwels et al., 2022). The LBD community has also developed ontologies for the built environment, such as the Building Topology Ontology (BOT) for describing building topology (Rasmussen et al., 2020). To create instance graphs, they develop converters to compile model information stored in IFC files into knowledge graphs, such as IFCToLBD (Bonduel et al., 2018). However, the converter

compiles part of the information for specific applications, ignoring some building data, such as geometry, which constrains the scope of applications (Bonduel et al., 2018).

Another emerging research domain in BIM uses graphs to represent and link multidisciplinary design models to support intelligent applications (Sacks et al., 2022; Törmä, 2013). The idea is to leverage the advantages of explicit edges in the graph format – each design model is compiled as a subgraph, and subgraphs are linked by across-domain relationships to form the metadata. Then, applications are applied to the linked meta-graphs, such as consistency maintenance, which propagates changes from one discipline graph to another, and enhancing interoperability by representing diverse design data in a uniform graph format (Wang et al., 2023a, 2023b). The initial CBIM prototype illustrated the feasibility of adopting a graph-based approach to support multidisciplinary intelligent applications (Wang et al., 2023b). However, this emerging domain requires extensive research on developing automatic methods to instantiate across-domain relationships to link multidisciplinary subgraphs. The potential solution is to investigate the graph representation of design information and the development of effective algorithms for the BIM graph linking scenarios.

3 Graph representation of BIM models for storage and learning

The aim of this paper is not to fully answer the two questions proposed. Instead, we explore the use of graphs to represent different types of design information serving machine learning, aiming to contribute some understanding of the first question about the data representation of BIM. We do not touch on the second question related to learning techniques.

We utilize two types of graphs: knowledge graphs for storing data and property graphs for computation, as shown in Figure 2. A BIM model has various types of information, including geometry, object attributes, object relationships, and so on. The knowledge graph is adopted to integrate and store these heterogeneous design data, where nodes stand for objects, and edges are object relationships. Additionally, some data that are not suitable to be represented as graphs, such as geometries, are stored in default geometry format files, and the file addresses are inserted in corresponding nodes as attribute values (Ouyang et al., 2023). Notably, the format and vocabulary in the knowledge graph should follow open schemas and ontologies to ensure interoperability. The main goals of the knowledge graph are to integrate various types of BIM data and store the as-is design information.

For computation, property graphs are used. The property graph can be generated from the knowledge graph by processing multiple types of design data as embeddings. For example, the object relationships are processed as features for edges and nodes, and the attributes are extracted as vectors for nodes. For data not in a graph format, other processing techniques are needed to leverage these data into property graphs. For instance, we compute 24 different geometry features and concatenate them into node attribute vectors as a way of embedding geometry into graphs. Property graphs are designed for computation without the restrictions of formats. Instead, the construction of property graphs could be different for specific scenarios.

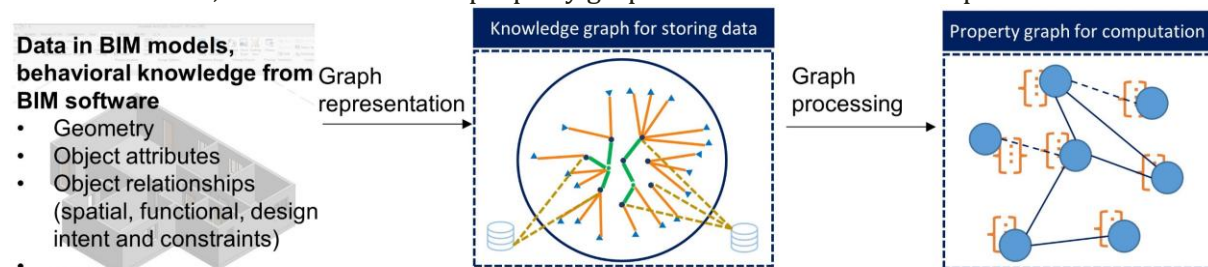


Figure 2. Using graphs to represent and store building information, and to leverage learning.

4 Case study

4.1 Apartment graph data construction

The architectural object classification task is selected as a case study. We collected 32 apartment unit designs, stored in IFC files. We constructed a parser using Python and

IfcOpenShell (2024) to retrieve the information contained in IFC files. The extracted raw information includes all building objects, object geometries if available, object attributes, and some relationships. The object geometries are processed and stored in OBJ file format for each object.

For relationships, we utilized *IfcRelAggregates* to retrieve the containing relationship among the project, site, building, and storey. We used *IfcRelContainedInSpatialStructure* to parse building elements contained in the corresponding storey. We adopted *IfcOpenElement* to retrieve the hosting objects (linked by *IfcRelVoidsElement*) and the hosted objects (linked by *IfcRelFillsElement*). Additionally, we computed the distance between two exact geometries and generated the contact relationship if the distance is zero. In short, there are three types of relationships adopted in the knowledge graphs: 1) containing, 2) hosting/hostedby, and 3) contact. The first two are retrieved from IFC files and the last one is generated by geometry computation, as some design software may not generate the contact semantics.

Table 1. Designed features.

No	Name	Category	No	Name	Category
1	Num of edges	Building relationship	17	Max y	Geometry
2	Num of edge types	Building relationship	18	Max z	Geometry
3	Betweenness centrality	Graph topology	19	Min x	Geometry
4	Closeness centrality	Graph topology	20	Min y	Geometry
5	Clustering coefficient	Graph topology	21	Min z	Geometry
6	Avg shortest path length	Graph topology	22	Num_face	Geometry
7	Subgraph centrality	Graph topology	23	Max_face	Geometry
8	Area	Geometry	24	Num_face_adjacency	Geometry
9	Volume	Geometry	25	Num_facet	Geometry
10	Bbx length	Geometry	26	Num_unique_edges	Geometry
11	Bbx width	Geometry	27	Len_unique_edges	Geometry
12	Bbx height	Geometry	28	Num_vertices	Geometry
13	Bbx_height/bbx_length	Geometry	29	Average_degree	Geometry
14	Bbx_height/bbx_width	Geometry	30	Euler characteristic	Geometry
15	Bbx_width/bbx_length	Geometry	31	Genus	Geometry
16	Max x	Geometry	32	Num of connected components	Geometry

All parsed and enriched building semantics are compiled as a knowledge graph, where the terminology of object types and attributes follows the IFC schema, and the object relationships follow BOT. The geometry file address is stored under the corresponding nodes. An example of knowledge graph representation can be found in Fig 8 of Wang et al. (2024).

For data statistics, eight architectural object types are considered: site (32 instances), building (32), storey (65), door (295), slab (409), wall (1086), window (429), and railing (17). Feature engineering was adopted, as shown in Table 1. The first seven features are graph-based, where the first two describe the building topology, and the other five are computed based on graph algorithms related to the graph topology. Additionally, we compute 25 geometry features by following Utkucu et al. (2024). All the generated features are organized as tables for traditional machine learning algorithms, or inserted as node features in graphs for GNN.

4.2 Training and evaluation

We selected two algorithms for training and testing: one GNN algorithm and one traditional machine learning algorithm. As this experiment only considers node features without edge features, GraphSAGE was selected (Wang et al., 2021). Additionally, the random forest (RF) algorithm was selected for processing the tabular data, because it has demonstrated good performance for BIM object classification (Utkucu et al., 2024).

To ensure a fair comparison, we randomly divided 70% of the models to form the training dataset, and the remaining 30% of the models for the testing dataset. Accuracy and the F1 score were selected to evaluate performance. Accuracy is the ratio of correctly predicted instances to the total instances in the dataset. It provides an intuitive way to reveal performance as it does not consider class imbalances. Additionally, the F1 score is introduced to evaluate the ability of the algorithm on the imbalanced dataset, as it takes into account both false positives and false negatives of each class.

4.3 Results

This experiment aims to illustrate the performance of different algorithms with incremental features, specifically focusing on the trend of adding graph-based features. Two algorithms were used: GraphSAGE from GNNs and RF from traditional machine learning, with results presented in Table 2.

Table 2. Results of leveraging different features for machine learning.

No.	Format	Feature category*	Dimension	Algorithm	F1	Accuracy
0	Graph	-	0	GraphSAGE	0.49	43.7%
1	Graph	Building	2	GraphSAGE	0.74	81.1%
2	Graph	Topology	5	GraphSAGE	0.70	76.0%
3	Graph	Geometry	25	GraphSAGE	0.66	91.7%
4	Graph	Building + Geometry	27	GraphSAGE	0.67	92.8%
5	Graph	Building + Topology	7	GraphSAGE	0.81	84.8%
6	Graph	Building + Topology + Geometry	32	GraphSAGE	0.95	95.3%
7	Tabular	Building	2	RF**	0.46	75.3%
8	Tabular	Topology	5	RF	0.62	63.6%
9	Tabular	Geometry	25	RF	0.67	93.2%
10	Tabular	Building + Geometry	27	RF	0.81	94.5%
11	Tabular	Building + Topology	7	RF	0.74	81.5%
12	Tabular	Building + Topology + Geometry	32	RF	0.95	96.3%

* 'Building' refers to the two building relationship features. 'Topology' denotes the five graph topology features. Notably, the two feature categories are computed on graphs. ** RF: Random Forest.

As can be seen in Table 2, as the number of feature categories used increases, the performance of the algorithms improves significantly, and both algorithms achieve the most accurate and robust performance when using all features. For instance, the F1 score of GraphSAGE improves from 0.66 when using only a single feature category to 0.95 when combining all three feature categories (No. 6). Similarly, for RF, the accuracy improves by more than 30%, from 63.6% to 96.3%, and its F1 score reaches 0.95 when adopting all three feature categories (No. 12).

Both building and topology features are computed based on graphs. Comparing the adoption of graph-based features (No. 5 in GraphSAGE) with geometry features (No. 3), the graph-based features yield a higher F1 score (0.81 vs. 0.66), indicating a more robust prediction ability. RF shows a similar pattern, with lower accuracy but higher F1 scores when using graph-based features compared to only using geometry features.

Moreover, even without any designed features, GraphSAGE achieves an F1 score of 0.49 and an accuracy of 43.7% (No. 0), which is higher than the F1 score achieved by RF using two-dimensional building features (No. 7). This result reveals that a graph itself contains topology information that GNNs can leverage for prediction. For BIM graphs, the links between nodes represent practical meanings, which validates the advantage of the graph data structure in preserving BIM object relationships and contributing to algorithm performance.

5 Discussion

5.1 Advantages and disadvantages

The proposed graph-based approach offers several notable advantages and contributions. It illustrates the use of different types of graphs for various purposes. Knowledge graphs following ontologies are suitable as an as-is data repository for storage, while property graphs with more flexible formats can be used for computation. By leveraging multiple feature categories, the experiment demonstrates that algorithm performance significantly improves, especially with graph-based features. Furthermore, the experiment highlights that graph structures can effectively preserve building relationships and contribute to performance even without pre-defined features.

However, the approach also has several disadvantages. The process of constructing property graphs results in information loss. For example, all parsed IFC attributes were abandoned due to their diverse and complex language descriptions, which are challenging to represent in a uniform way for machine learning. Additionally, the geometry features used were artificially designed, whereas an ideal situation would allow machines to learn and generate geometry-based feature embedding directly. The experiment also considered a limited number of object classes, which do not represent the diversity of real BIM models. These limitations suggest a need for a more generic approach to graph representation learning, emphasizing rich, comprehensive, and full-scope data integration to address the highlighted challenges.

5.2 Challenges of graph representation learning for BIM

The experiments with graph representation learning for BIM aim to explore methods that leverage multi-modal design data, enabling machines to understand designs and support diverse BIM scenario tasks. In other words, this is part of an effort to seek a generic approach to process various types of design data to a machine-readable representation suitable for learning tasks.

The core of graph representation learning is to generate embeddings, by processing human-friendly design data into high-dimensional vectors suitable for computers. Different types of data may require different types of techniques for processing. For object attributes in natural language format, LLMs can leverage them to generate embeddings, effectively handling diverse and complex attribute descriptions. Additionally, learning algorithms can be used to process object geometries and generate embeddings directly. Incorporating vision features into the graph representation learning process can provide richer data context. This can be achievable by generating object images using virtual cameras and converting these images into feature embeddings through deep learning. Moreover, edge features representing relationships should also be considered for embedding.

Beyond generation, integrating these different embeddings poses another challenge. A straightforward method could involve concatenating all embeddings together. Alternatively, learning algorithms could be used to process these diverse embeddings and output a unified representation. These approaches need to be validated through experiments to determine their effectiveness. However, it appears likely that new AI algorithms will be needed for processing the new BIM data representations.

5.3 Towards general AI systems for BIM

The two questions we posed are driven by an ambitious goal of developing a general AI system that can understand BIM, much like LLMs understand human language. This level of intelligence would provide a versatile solution for many existing BIM-related tasks and open new possibilities, just as LLMs have enabled a wide array of NLP applications.

To achieve this ultimate goal, the first fundamental task is to develop effective and efficient embedding methods that can integrate and project complex, diverse, and heterogeneous BIM data into a vector space where algorithms can operate. This contains developing learning-suitable representations for BIM data, as explained in Section 5.2.

Next, it is crucial to develop self-supervised learning techniques compatible with BIM. The success of current LLMs is mainly due to pre-training with vast amounts of data in an efficient,

self-supervised manner, where data labeling for supervised learning becomes impractical (Radford et al., 2018). The key to self-supervised learning is to obtain learning signals from the raw input data. LLMs obtain these signals by predicting the next token in a sequence from vast amounts of text data, thus learning the structure and semantics of the language (Radford et al., 2018). For BIM data, self-supervised learning signals could be derived from the inherent semantic information within the BIM models. Additionally, domain-specific pretext tasks can be developed, such as predicting missing components or reconstructing parts of the BIM from partial information.

Building on effective embedding and self-supervised learning techniques, it becomes technically feasible to develop large models, i.e., general AI systems for BIM. This may involve self-supervised pre-training with a vast number of BIM models, followed by supervised fine-tuning to align the pre-trained model's behavior with the specific needs of target users, such as designers and engineers.

Lastly, a general AI system for BIM may need to be multimodal with the ability of understanding human language. Human language is the most natural way for people to communicate with computer systems. An alternative solution could involve developing converters to translate user instructions from human language into BIM representation formats accepted by the AI system. However, experience from NLP and computer vision indicates that naturally multimodal models generally perform better in terms of accuracy, cost, and inference speed (OpenAI, 2024).

5.4 Re-visiting the key questions

Back to the two questions posed at the beginning: 1) What formats are most suitable for BIM data representation? 2) What are the corresponding learning techniques needed for BIM? Note that we make no claim that graphs are the best data format, but have illustrated in this work that graphs are a useful approach as they can capture complex and diverse BIM data to support learning.

What is next? In section 5.2, we further discussed the exploration of processing the various types of BIM data as embeddings to support machine learning. This could be the potential answer to the first question. In section 5.3, we illustrate our latest understanding of designing suitable techniques and roadmaps towards general AI systems for BIM, aiming to fully resolve the two questions. We understand that the two questions could be interrelated, as formulating the data representation requires the development of new appropriate learning techniques. Lastly, we believe that further exploration of these questions, even if not fully achieving the ultimate goal, will provide deeper insights into the best practices for integrating machine learning with BIM.

6 Conclusion

BIM models contain diverse data. Current BIM machine learning studies typically adopt an inverse approach to process part of design data into a specific format to launch a pre-selected algorithm. We argue that new data representations and learning techniques are necessary to fully utilize the richness and scope of information contained in BIM models for machine learning applications. Therefore, we raised the two fundamental questions stated in the introduction: **1) What formats are most suitable for BIM data representation? 2) What are the corresponding learning techniques needed for BIM?**

As an exploratory study, we proposed a graph-based approach for representing building information and developed one experiment to illustrate the effectiveness of leveraging various design data. In the BIM object classification experiment, both machine learning and GNN algorithms performed more accurately and robustly as the richness of features was increased. Specifically, the two algorithms achieve accuracy and F1 at 0.95, after considering graph-based and geometry features. In contrast, if we remove all the designed features, GraphSAGE, the GNN algorithm, can leverage only the graph structure and achieve an F1 score of 0.49 in an eight-class classification task. This further validates that the graph data structure preserves semantics and can contribute to learning.

This work only scratches the surface of the first question about the data representation of BIM. Future research will involve exploring BIM graph representation learning by integrating and processing multi-modal design data into high-dimensional embeddings to support various downstream learning tasks. Self-supervised learning techniques will be tested on large BIM graph datasets with interactive training techniques towards a general AI system for BIM. We believe that raising these questions will foster avenues for both academia and industry to explore new paths of application of AI to BIM, eventually enabling a more intelligent future for building design.

References

- Bonduel, M., Oraskari, J., Pauwels, P., Vergauwen, M., & Klein, R. (2018). The IFC to linked building data converter - Current status. *CEUR Workshop Proceedings*, 2159, 34–43. <https://github.com/jyrkioraskari/IFCtoRDF-Desktop>
- Boroojerdi, S., & Rudolph, G. (2022). Handwritten Multi-Digit Recognition With Machine Learning. *2022 Intermountain Engineering, Technology and Computing, IETC 2022*, 1–6. <https://doi.org/10.1109/IETC54973.2022.9796722>
- 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>
- Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2018). *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. *Mlm*. <http://arxiv.org/abs/1810.04805>
- He, K., Zhang, X., Ren, S., & Sun, J. (2015). Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. *2015 IEEE International Conference on Computer Vision (ICCV), 2015 Inter*, 1026–1034. <https://doi.org/10.1109/ICCV.2015.123>
- He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep Residual Learning for Image Recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016-Decem*, 770–778. <https://doi.org/10.1109/CVPR.2016.90>
- IfcOpenShell. (2024). *IfcOpenShell*. <https://ifcopenshell.org/>
- Koo, B., Jung, R., & Yu, Y. (2021). Automatic classification of wall and door BIM element subtypes using 3D geometric deep neural networks. *Advanced Engineering Informatics*, 47, 101200. <https://doi.org/10.1016/j.aei.2020.101200>
- 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., & Shin, B. (2018). Applying novelty detection to identify model element to IFC class misclassifications on architectural and infrastructure Building Information Models. *Journal of Computational Design and Engineering*, 5(4), 391–400. <https://doi.org/10.1016/j.jcde.2018.03.002>
- Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet Classification with Deep Convolutional Neural Networks. In F. Pereira, C. J. Burges, L. Bottou, & K. Q. Weinberger (Eds.), *Advances in Neural Information Processing Systems* (Vol. 25). Curran Associates, Inc. https://proceedings.neurips.cc/paper_files/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf
- 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>
- Mikolov, T., Sutskever, I., Chen, K., Corrado, G., & Dean, J. (2013). Distributed Representations of Words and Phrases and their Compositionality. *Advances in Neural Information Processing Systems*, 1–9. <http://arxiv.org/abs/1310.4546>
- OpenAI. (2022, November 30). *Introducing ChatGPT*. <https://openai.com/index/chatgpt/>
- OpenAI. (2024). *Hello GPT-4o*. <https://openai.com/index/hello-gpt-4o/>
- OpenSeq2Seq. (2024, May). *Speech Recognition*. <https://nvidia.github.io/OpenSeq2Seq/html/speech-recognition.html#>
- Ouyang, B., Wang, Z., & Sacks, R. (2023). Semantic enrichment of object associations across federated BIM semantic graphs in a common data environment. *EWork and EBusiness in Architecture, Engineering and Construction - Proceedings of the 14th European Conference on Product and Process Modelling, ECPPM 2022*, 591–598. <https://doi.org/10.1201/9781003354222-75>
- Pauwels, P. (2021). *Linking BIM models with monitored data using Semantic Web technologies*. October.
- Pauwels, P., Costin, A., & Rasmussen, M. H. (2022). *Knowledge Graphs and Linked Data for the Built Environment* (pp. 157–183). Springer, Cham. https://doi.org/10.1007/978-3-030-82430-3_7

- Radford, A., Narasimhan, K., Salimans, T., & Sutskever, I. (2018). *Improving language understanding by generative pre-training*. OpenAI. <https://www.mikecaptain.com/resources/pdf/GPT-1.pdf>
- Rasmussen, M. H., Lefrançois, M., Schneider, G. F., & Pauwels, P. (2020). BOT: The building topology ontology of the W3C linked building data group. *Semantic Web*, 12(1), 143–161. <https://doi.org/10.3233/SW-200385>
- Sacks, R., Wang, Z., Ouyang, B., Utkucu, D., & Chen, S. (2022). Toward artificially intelligent cloud-based building information modelling for collaborative multidisciplinary design. *Advanced Engineering Informatics*, 53(June), 101711. <https://doi.org/10.1016/j.aei.2022.101711>
- Törmä, S. (2013). Semantic linking of building information models. *Proceedings - 2013 IEEE 7th International Conference on Semantic Computing, ICSC 2013*, 412–419. <https://doi.org/10.1109/ICSC.2013.80>
- Utkucu, D., Ying, H., Wang, Z., & Sacks, R. (2024). Classification of architectural and MEP BIM objects for building performance evaluation. *Advanced Engineering Informatics*, 61(September 2023), 102503. <https://doi.org/10.1016/j.aei.2024.102503>
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems, 2017-Decem(Nips)*, 5999–6009.
- Wang, Z., Ouyang, B., & Sacks, R. (2023a). CBIM: object-level cloud collaboration platform for supporting across-domain asynchronous design. *Proceedings of the European Conference on Computing in Construction*. <https://doi.org/10.35490/EC3.2023.189>
- Wang, Z., Ouyang, B., & Sacks, R. (2023b). Graph-based inter-domain consistency maintenance for BIM models. *Automation in Construction*, 154, 104979. <https://doi.org/10.1016/j.autcon.2023.104979>
- Wang, Z., Sacks, R., Ouyang, B., Ying, H., & Borrmann, A. (2024). A Framework for Generic Semantic Enrichment of BIM Models. *Journal of Computing in Civil Engineering*, 38(1), 1–18. <https://doi.org/10.1061/JCCEE5.CPENG-5487>
- 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., Yeung, T., Sacks, R., & Su, Z. (2021). Room Type Classification for Semantic Enrichment of Building Information Modeling Using Graph Neural Networks. *Proc. of the Conference CIB W78*, 773–781.