Query2Property: Semantic retrieval of IFC properties for natural language BIM queries

Rabindra Lamsal[1], Sisi Zlatanova[1]

[1]GRID Lab, School of Built Environment, UNSW Sydney, NSW 2052, Australia - (r.lamsal,s.zlatanova)@unsw.edu.au

Keywords: Text Embeddings, Vector Search, Natural Language Querying, Building Information Modeling, Industry Foundation Classes

Abstract

IFC models store detailed building information, but their complex schema and deeply nested property sets make querying difficult for non-expert users and challenging for large language models (LLMs) to handle directly. Current LLM-based approaches are inefficient because prompts often include entire IFC schemas, many properties of which are irrelevant to the user's query, leading to higher inference costs and potential errors. This paper presents *Query2Property*, a semantic retrieval system that maps natural language queries to the most relevant IFC properties. By embedding both property descriptions and user queries in a shared vector space, the system retrieves contextually relevant properties for dynamic and concise prompt construction in LLM-driven workflows. Evaluation on 55 representative BIM queries achieves a top-1 accuracy of 87.3% and top-3 accuracy of 100%, demonstrating effective alignment with user intent. Query2Property simplifies LLM-based workflows over BIM data, supporting semantic search and natural language exploration of complex building information.

1. Introduction

The Architecture, Engineering, and Construction (AEC) industry increasingly relies on Building Information Modeling (BIM) (Eastman, 2011) to manage and exchange data about built environments (Wang et al., 2019). BIM stores detailed information about building geometry, materials, and element-specific attributes, serving as a foundation for design, analysis, and facility management workflows (Irizarry et al., 2013, Kim et al., 2015, Won and Cheng, 2017, Hasan et al., 2019), as well as wayfinding and localization (Isikdag et al., 2011, Liu et al., 2021, Diakite et al., 2022, Zhu et al., 2025). Among BIM data standards, the Industry Foundation Classes[1] (IFC) schema is the most widely adopted open standard, which captures both the geometric and semantic aspects of building components. IFC files store rich property information (e.g., material, fire rating, dimensions) and spatial relationships for entities such as walls, doors, slabs, and spaces.

Despite their richness, IFC models are challenging to query and interpret without specialized software or technical expertise (Ma and Liu, 2018). Extracting simple information, such as “What is the fire rating of door X?”, often requires writing IFC-specific scripts or using complex modeling tools. The structure of IFC data is deeply nested, and property sets may vary across element types, making querying difficult for non-technical users.

Recent advances in large language models (LLMs) (Zhao et al., 2023), such as GPT (Achiam et al., 2023) and LLaMA (Touvron et al., 2023), have enabled natural language interaction with structured data (Zhao et al., 2024). Users can now query and explore complex data conversationally. However, directly supplying an entire IFC model to an LLM is infeasible due to input token limits and computational costs. Retrieval-Augmented Generation (RAG) frameworks (Cuconasu et al., 2024) mitigate this by retrieving relevant data subsets before response generation. In such systems, a system prompt (Yang et al., 2024) defines the task and available data sources, including their schemas, and is included in each LLM invocation. When querying a relational database derived from IFC, the prompt may include schemas for several tables (e.g., IfcDoor, IfcWall, IfcSpace) and their properties. Including all possible properties for each element type is both inefficient and unnecessary, as large IFC projects can define hundreds of properties per class.

Consider these two queries: *What is the fire rating of door X?* and *What is the acoustic rating of wall Y?* In both cases, the LLM only needs access to a small, relevant subset of IFC property definitions, such as *FireRating* and *AcousticRating*. Providing just these in the system prompt allows the LLM to generate precise structured queries (e.g., SQL) in one step, using the correct property names from the schema, rather than semantically guessing terms like “Acoustic Rating” or “NoiseRating”. Furthermore, when LLM operates in iterative workflows, i.e., refining queries, verifying results, or exploring reasoning chains, this inefficiency compounds rapidly. This motivates the need for a semantic retrieval mechanism that maps user queries to the most relevant IFC property definitions, which allows dynamically constructing concise and context-aware system prompts.

To address this limitation, we developed *Query2Property*, a semantic retrieval system that automatically maps user queries to the most relevant IFC property. The approach projects both IFC property descriptions and user queries in the same vector space, enabling retrieval beyond simple keyword matching. For example, when a user asks, “What is the fire rating of wall Y?”, the system retrieves the *FireRating* property from the *Pset_WallCommon* property set, demonstrating its ability to translate natural language intent into formal IFC semantics. This capability not only supports dynamic prompt construction for LLM-based reasoning over BIMs but also enables semantic search and navigation of IFC standards based on conceptual intent rather than exact terminology. However, Query2Property is primarily intended as a modular component within LLM-based BIM querying pipelines, where it retrieves the most relevant properties for prompt construction prior to query generation.

[1] `https://technical.buildingsmart.org/standards/ifc/`

The approach does not aim to solve full end-to-end BIM question answering, multi-hop reasoning, geometric reasoning, or cross-object logical inference.

The paper is organized as follows. Section 2 covers the related literature, Section 3 introduces the retrieval system's workflow, Section 4 presents the experimental results and discussion, and Section 5 concludes the paper.

2. Literature Review

In this section, we briefly discuss the relevant literature on LLMs for BIMs, as well as the state of the art in text embeddings and vector search.

LLMs for BIMs: LLMs are being extensively used in managing BIMs. For instance, (Wang et al., 2024) proposed a method to automatically optimize BIM using an LLM. Their method generated and corrected BIM components, particularly from complex IFC text files, by extracting relevant data instances, annotating them, and guiding the LLM to generate accurate IFC texts. (Du et al., 2025) explored the effectiveness of LLMs for robotic tasks in construction. Their proposed framework utilized tailored tools and structured prompt templates in performing tasks such as bricklaying and 3D concrete printing. (Du et al., 2025) used LLMs to automatically match human queries to BIM data and domain-specific code functions, which is a complex and error-prone process. Furthermore, LLMs have also been used in a multi-agent environment for generating building models from natural langauge instructions (Du et al., 2024).

Recently, RAG frameworks (Cuconasu et al., 2024) have been effectively applied to develop state-of-the-art BIM querying systems. For instance, (Iranmanesh et al., 2025) proposed a graph-based RAG approach in which an IFC model is transformed into a graph, and an LLM is instructed to query this graph to answer user questions. Their method provides the LLM with the graph schema, formatting rules, and few-shot examples as part of the prompt instructions. Similarly, BIMConverse (Pacheco and Berkmiller, 2024) adopts the Graph RAG concept, implementing a comprehensive set of instructions that include the graph schema, few-shot examples, and detailed task-specific guidance for the LLM. These instructions are concatenated with the user's query before being passed to the model. RAG-based BIM querying systems rely heavily on carefully engineered system prompts to provide the LLM with sufficient contextual information for accurate reasoning. To further advance this area, it is essential to design mechanisms for dynamic prompting (Choi et al., 2025), which allows LLMs to achieve stronger reasoning performance while minimizing token usage. Therefore, this study focuses on retrieving only the IFC properties relevant to user queries to construct concise and context-aware prompts.

While RAG frameworks enable LLMs to reason over BIM data, their effectiveness depends on retrieving semantically relevant context. This motivates the use of text embeddings and vector search (Pan et al., 2024) for efficient semantic retrieval of IFC properties.

Text Embeddings and Vector Search: Text Embeddings are fixed-length low-dimensional vector representations of arbitrary-length texts (Wang et al., 2022). They have applications across numerous natural language processing tasks, including information retrieval, search systems and text mining (Muennighoff et al., 2022). Some embedding models are tailored for specific tasks, such as CrisisTransformers (Lamsal et al., 2024), while others aim to produce *universal embeddings* (Muennighoff et al., 2022, Cao, 2024) that generalize across multiple applications, including classification, clustering, and semantic similarity tasks. The quality of embeddings has traditionally been assessed using benchmark datasets such as SemEval[2], SentEval (Conneau and Kiela, 2018), and BEIR (Thakur et al., 2021). However, these benchmarks were limited in scope and task diversity, which motivated the introduction of unified evaluation frameworks with 100+ embedding tasks — Massive Text Embedding Benchmark (MTEB) (Muennighoff et al., 2022) and Massive Multilingual Text Embedding Benchmark (MMTEB) (Enevoldsen et al., 2025). In this study, we employ an embedding model from the Qwen3 Embedding model family (Zhang et al., 2025), as all their models rank among the top five overall performers in the MMTEB benchmark[3].

Designed to efficiently index and retrieve text embeddings based on semantic similarity, vector databases are critical to modern retrieval systems (Pan et al., 2024). Such databases rely on Approximate Nearest Neighbor (ANN) search algorithms (Arya et al., 1998), which enable fast and scalable indexing and retrieval in high-dimensional spaces. Common ANN methods include Hierarchical Navigable Small World (HNSW) graphs (Malkov and Yashunin, 2018) and Inverted File Index with Product Quantization (IVF-PQ) (Jegou et al., 2010). Popular vector databases such as FAISS (Douze et al., 2024), ChromaDB[4], Pinecone[5], pgvector[6], Weaviate[7], and Qdrant[8] implement these ANN techniques and additionally support operations like metadata filtering, hybrid search, and CRUD functionalities. This study implements a FAISS (CPU) and HNSW setup, which is sufficient for a search space of IFC properties ($\leq$ 4k vectors), while keeping the system simple and reproducible.

3. Method

The workflow of the proposed retrieval system, Query2Property, consists of three main steps: (1) curating an IFC properties corpus, (2) generating text embeddings, and (3) performing vector-based similarity search. An overview of the system is shown in Figure 1. We briefly describe each step in this section.

3.1 IFC Properties Corpus

Given the complete schema of IFC property and quantity sets, we use these definitions to construct a corpus suitable for text embedding and subsequent vector search. Our approach extracts and organizes the following property attributes:

– **id**: A unique identifier that combines the property set name and property name (e.g., *Pset_DoorCommon.FireRating*).
– **pset**: The property set name, which groups related properties together (e.g., *Pset_DoorCommon*).
– **property**: The specific property name within the property set (e.g., *FireRating*).
– **description**: A human-readable explanation of what the property represents and how it should be used.

[2] https://semeval.github.io/
[3] https://huggingface.co/spaces/mteb/leaderboard
[4] https://github.com/chroma-core/chroma
[5] https://www.pinecone.io/
[6] https://github.com/pgvector/pgvector
[7] https://github.com/weaviate/weaviate
[8] https://github.com/qdrant/qdrant

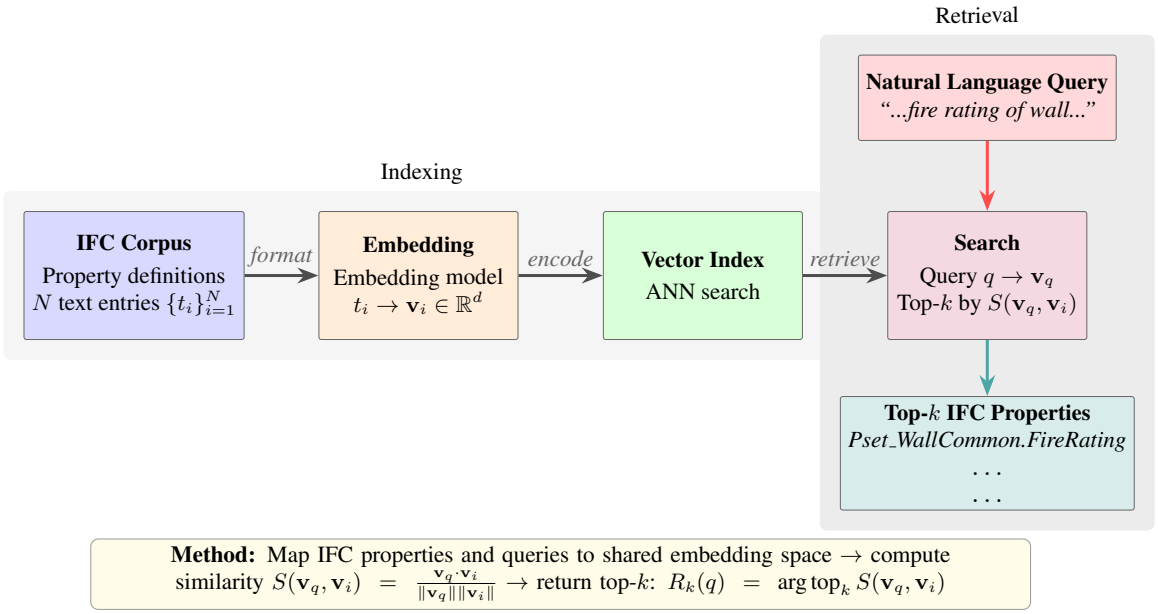

Figure 1. Overview of Query2Property for semantic retrieval of IFC properties.

– **applies_to**: A list of IFC entity types to which the property applies (e.g., [“IfcDoor”, “IfcDoorType”]).
– **type**: The classification type of this metadata entry. *PropertySet* or *QuantitySet*.
– **source**: The IFC schema version from which this property definition originates (e.g., *IFC4.3*).
– **aliases**: Alternative names or synonyms for the property (e.g., *fire resistance* for *FireRating*).
– **semantic_representation**: A formatted text string that combines all available property information.

The final text t (*semantic_representation*), used for embedding is created by concatenating the *pset*, *property*, *description*, *applies_to*, and *aliases* fields into a single coherent text sequence.

3.2 Text Embeddings

To enable semantic search, each text entry t_i from the IFC properties corpus is transformed into a numerical vector representation using an embedding model $E(\cdot)$. The embedding model maps variable-length texts into a fixed-dimensional vector space $\mathbb{R}^d$:

$$\mathbf{v}_i = E(t_i), \quad \mathbf{v}_i \in \mathbb{R}^d$$

where t_i is the input text and $\mathbf{v}_i$ is its corresponding embedding vector. The embedding function $E(\cdot)$ projects semantically similar texts to nearby points in the vector space, i.e., for two texts t_i and t_j,

$$\text{similarity}(t_i, t_j) \propto \cos(\mathbf{v}_i, \mathbf{v}_j)$$

where $\cos(\mathbf{v}_i, \mathbf{v}_j)$ denotes the cosine similarity between their embeddings. This property allows conceptually related IFC property definitions (e.g., “Pset_WallCommon.FireRating”) and queries (e.g., “... fire rating of wall ...”) to appear close to each other in the embedding space. An illustration of a simplified two-dimensional query-property embedding space is shown in Figure 2.

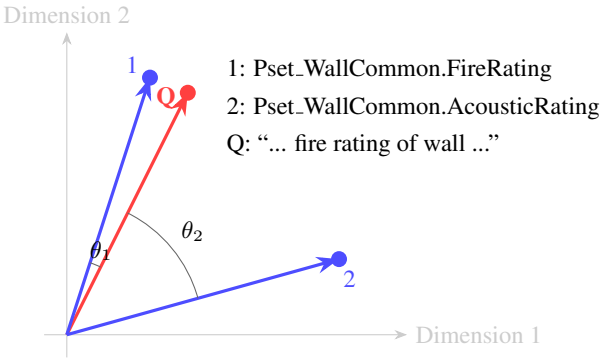

Figure 2. A simplified two-dimensional query–property embedding space. The query vector (Q) and IFC property vectors (1–2) are embedded in the same space; similarity is indicated by the angle at the origin. A smaller angle θ_1 denotes higher cosine similarity, so *Pset_WallCommon.FireRating* (1) ranks above *Pset_WallCommon.AcousticRating* (2).

3.3 Vector Search

Once all text entries t_i are embedded, we perform semantic retrieval using vector similarity search. Given a query q, we first compute its embedding vector:

$$\mathbf{v}_q = E(q)$$

Then, we measure its similarity to each property vector $\mathbf{v}_i$ in the corpus using cosine similarity:

$$S(\mathbf{v}_q, \mathbf{v}_i) = \frac{\mathbf{v}_q \cdot \mathbf{v}_i}{\|\mathbf{v}_q\| \, \|\mathbf{v}_i\|}$$

The top-k most similar properties are retrieved according to their similarity scores:

$$R_k(q) = \arg\text{top}_k \; S(\mathbf{v}_q, \mathbf{v}_i), \quad \forall i \in \{1, \ldots, N\}$$

This contribution has been peer-reviewed. The double-blind peer-review was conducted on the basis of the full paper.
https://doi.org/10.5194/isprs-annals-XI-4-2026-323-2026 | © Author(s) 2026. CC BY 4.0 License.

where N is the total number of property entries in the corpus, and $R_k(q)$ denotes the set of top-k results most semantically related to the query. This enables users to search for IFC properties using natural language, synonyms, or partial descriptions rather than relying on exact string matches.

4. Experimentation and Discussion

4.1 System Setup

IFC Properties Corpus: We obtained the IFC property definitions from the official *IFC Documentation*[9]. The definitions are provided as XML files, which include both property and quantity set specifications. We parsed these XML files to extract the property attributes discussed in Section 3.1, including identifiers, property set names, descriptions, and applicable entities. Each extracted property entry was structured according to our corpus schema (see Section 3.1) and converted into a JSON format suitable for downstream embedding and vector search experiments. An example of one such extracted property definition is shown below:

```
{
  "id": "Pset_DoorCommon.FireRating",
  "pset": "Pset_DoorCommon",
  "property": "FireRating",
  "description": "Fire rating for this object. It is
  ↪  given according to the national fire safety
  ↪  classification.",
  "applies_to": [
    "IfcDoor",
    "IfcDoorType"
  ],
  "type": "PropertySet",
  "source": "IFC4.3",
  "aliases": [],
  "semantic_representation":
  ↪  "Pset_DoorCommon.FireRating - Fire rating for
  ↪  this object. It is given according to the
  ↪  national fire safety classification. Applicable
  ↪  to: IfcDoor, IfcDoorType."
}
```

The *semantic_representation* field aggregates all the textual definitions and contexts provided for each property in the official IFC documentation. Therefore, the resulting vector representation of this field can be considered an exhaustive encoding of a property’s semantic meaning. The field can be further enriched by also including aliases (e.g., “heat resistance”, “heat resisting capacity” for properties such as *FireRating*). At this stage of development, we do not use aliases.

We use *Qwen3-Embedding-0.6B*[10] as the embedding model for generating vector representations. This model contains 0.6 billion parameters, supports a context length of 32k tokens and generates embeddings of 1,024 dimensions. Due to its moderate parameter size, it can be efficiently executed on a standalone device, making it suitable for local experimentation in this study. The 32k context length is sufficient to encode the full property definitions, while the 1,024-dimensional embeddings enable efficient vector search operations even in low-resource environments. The statistics for *semantic_representation* at word and token levels are provided in Table 1.

[9] https://ifc43-docs.standards.buildingsmart.org/
[10] https://huggingface.co/Qwen/Qwen3-Embedding-0.6B

	words	tokens
min	5	18
max	341	467
median	19	50
25th percentile	14	38
75th percentile	30	66

Table 1. Statistics of the *semantic_representation* field.

Vector Search: We used the *Langformers* Python library (Lamsal et al., 2025) to implement embeddings generation and vector search environments. Our vector search implementation was based on the FAISS vector database combined with HNSW indexing. Since the search space in our experiments consists of $\leq 4k$ vectors, optimizing the indexing strategy is considered outside the scope of this study.

For each IFC property, its *semantic_representation* field was encoded into an embedding vector. Its complete property definition, represented as a Python dictionary, was used as associated metadata and stored together with the embedding in the FAISS database. Such metadata can be used to refine search results based on attributes such as “pset”, “applies_to”, etc.

4.2 Results and Analysis

To systematically evaluate our method, we manually curated a benchmark consisting of 55 natural language queries that represent common requests in BIM workflows. Each query was paired with its corresponding expected IFC property (ground truth), covering a wide range of building element classes (e.g., doors, walls, windows, materials) and property domains (e.g., fire safety, acoustics, thermal performance, material characteristics, maintenance). For each query, the system retrieves the top-k most semantically similar IFC properties from the embedded property space. Performance was evaluated using top-1 accuracy, which measures whether the correct property appeared as the highest-ranked result, and top-3 accuracy, which measures whether it appeared among the top three retrieved candidates. The queries, ground-truth properties, and system retrieval results are summarized in Table 2.

Results show that, out of 55 queries, in 48 instances (i.e., 87.27%) the system retrieved the correct expected property as the top-ranked result. In the remaining 7 cases (refer Table 3), the correct property appeared within the top three retrieved candidates; for 6 queries as the second result and for 1 query as the third. We obtained a top-3 accuracy of 100%, indicating that the system consistently ranked the correct property among the most semantically relevant results. With an average search time of $\approx$0.16 seconds per query, the system demonstrates both feasibility and a strong semantic alignment between user intent and IFC property definitions. These results highlight the suitability of the approach for integration into LLM-BIM workflows, where it can support natural-language querying of IFC models; an example of such integration is shown in Figure 3.

To further explore the system’s behavior, we analyzed the 7 queries where the correct property was not ranked first (Table 3). In most of these cases, the top-1 prediction corresponded to a semantically similar but contextually adjacent property. For example, for “Is column C1 a load-bearing column?”, the model ranked *Pset_MemberTypePost.LoadBearingCapacity* higher than the correct *Pset_ColumnCommon.LoadBearing*, reflecting confusion between structurally related property sets. Similarly, “Is window W6 using a tempered glass?” prioritized *IsWired* over

#	Query	Expected Property	Result
1	What is the fire rating of door D1?	Pset_DoorCommon.FireRating	✓
2	How fire-resistant is wall W1?	Pset_WallCommon.FireRating	✓
3	What’s the sound insulation level of wall W2?	Pset_WallCommon.AcousticRating	✓
4	What’s the soundproofing value of wall W3?	Pset_WallCommon.AcousticRating	✓
5	What is the thermal transmittance (U-value) of window W1?	Pset_WindowCommon.ThermalTransmittance	✓
6	How energy-efficient is the window glazing?	Pset_DoorWindowGlazingType.ShadingCoefficient	✓
7	Is wall W4 a load-bearing one?	Pset_WallCommon.LoadBearing	✓
8	Is column C1 a load-bearing column?	Pset_ColumnCommon.LoadBearing	✓**
9	What is the thickness of slab S2?	Qto_SlabBaseQuantities.Depth	✓
10	What is the height of door D2?	Qto_DoorBaseQuantities.Height	✓
11	What is the width of window W4?	Qto_WindowBaseQuantities.Width	✓
12	Is window W6 using a tempered glass?	Pset_DoorWindowGlazingType.IsTempered	✓**
13	What is the fire door’s burn resistance?	Pset_DoorCommon.FireRating	✓
14	What is the fire resistance class of the main corridor door?	Pset_DoorCommon.FireRating	✓
15	What is the acoustic performance of floor slab F1?	Pset_SlabCommon.AcousticRating	✓
16	Give me the thermal conductivity of insulation layer I1?	Pset_MaterialThermal.ThermalConductivity	✓
17	What is the strength of concrete used in column C2?	Pset_MaterialConcrete.CompressiveStrength	✓
18	What is the U-value of roof R1?	Pset_RoofCommon.ThermalTransmittance	✓
19	What is the fireproof rating of the curtain wall C1?	Pset_CurtainWallCommon.FireRating	✓
20	What is the door’s acoustic isolation value?	Pset_DoorCommon.AcousticRating	✓
21	What is the flow rate of duct D3?	Pset_DistributionPortTypeDuct.VolumetricFlowRate	✓
22	What is the power rating of electrical circuit C1?	Pset_ElectricalDeviceCommon.Power	✓
23	What is the voltage rating of device D1?	Pset_ElectricalDeviceCommon.RatedVoltage	✓
24	What is the voltage rating of sensor S1?	Pset_AxleCountingEquipment.RatedVoltage	✓
25	What is the pressure range of pipe P1?	Pset_PipeSegmentTypeCommon.PressureRange	✓
26	What is the airflow of fan F1?	Pset_FanPHistory.DischargeVelocity	✓
27	What is the current aging of lamp L2?	Pset_LampTypeCommon.LampMaintenanceFactor	✓
28	What is the impermeability of concrete block material?	Pset_MaterialConcrete.WaterImpermeability	✓
29	What is the glazing thickness of window W7?	Pset_DoorWindowGlazingType.GlassThickness1/2/3	✓
30	What is the fire rate in beam B1?	Pset_BeamCommon.FireRating	✓**
31	What is the slope angle of beam B2?	Pset_BeamCommon.Slope	✓
32	What is the current status of roof R2?	Pset_RoofCommon.Status	✓
33	What is the durability class of timber beam B2?	Pset_MaterialWood.StrengthGrade	✓
34	What is stress capacity of steel frame F2?	Pset_MaterialSteel.UltimateStress	✓
35	What is the plastic strain of steel frame F3?	Pset_MaterialSteel.PlasticStrain	✓
36	Give me the structural grade of steel frame F4?	Pset_MaterialSteel.StructuralGrade	✓
37	Is wall W8 made from a material that burns?	Pset_(Curtain)WallCommon.Combustible	✓
38	What is the warranty period of the furniture F1?	Pset_Warranty.WarrantyPeriod	✓
39	When does the warranty start for Furniture F2?	Pset_Warranty.WarrantyStartDate	✓**
40	When was Element E4 installed?	Pset_ConstructionOccurence.InstallationDate, Pset_InstallationOccurrence.InstallationDate	✓
41	How much renewable energy is used by Element E5?	Pset_EnvironmentalImpactValues.RenewableEnergyConsumption	✓
42	How much water is used by Element E6?	Pset_EnvironmentalImpactValues.WaterConsumption	✓
43	Is railing R3 outside the building?	Pset_RailingCommon.IsExternal	✓***
44	Give me the monitoring strategy for the wall?	Pset_MaintenanceStrategy.MonitoringType	✓
45	Can ramp R2 be used as an exit during emergency?	Pset_RampCommon.FireExit	✓
46	Give me the upcoming assessment date of element E7.	Pset_Condition.NextAssessmentDate	✓
47	How frequently do I need to assess element E8?	Pset_Condition.AssessmentFrequency	✓
48	Is there any ramp which is slippery?	Pset_RampCommon.HasNonSkidSurface	✓**
49	List me ramps which are accessible for disabled people.	Pset_RampCommon.HandicapAccessible	✓
50	Count me the number of risers in stair S1.	Pset_StairCommon.NumberOfRiser	✓
51	Extract the length of tread of stair S1.	Pset_StairCommon.TreadLength	✓
52	What amount of solar is transmitted through shade S1?	Pset_ShadingDeviceCommon.SolarTransmittance	✓**
53	Give me the color of shade S1’s surface?	Pset_ShadingDeviceCommon.SurfaceColour	✓
53	Can I operate shade S1 manually?	Pset_ShadingDeviceCommon.MechanicalOperated	✓
54	Is beam B3 external to the building?	Pset_BeamCommon.IsExternal	✓
55	Please provide the permit start date.	Pset_Permit.StartDate	✓

Table 2. Evaluation results. The table reports system performance across 55 manually curated natural language queries. A check mark (✓) indicates that the correct property was ranked first, while ** and *** denote cases where it appeared in the second or third position within the top three results.

Query	Top-3 retrieved properties	Similarity
Is column C1 a load-bearing column?	Pset_MemberTypePost.LoadBearingCapacity **Pset_ColumnCommon.LoadBearing** Pset_ReinforcementBarPitchOfColumn.ReinforcementBarType	0.7839 0.7761 0.7739
Is window W6 using a tempered glass?	Pset_DoorWindowGlazingType.IsWired **Pset_DoorWindowGlazingType.IsTempered** Pset_DoorWindowGlazingType.GlassThickness2	0.7914 0.7860 0.7718
What is the fire rate in beam B1?	Pset_SlabCommon.SurfaceSpreadOfFlame **Pset_BeamCommon.FireRating** Pset_WallCommon.SurfaceSpreadOfFlame	0.7908 0.7869 0.7869
When does the warranty start for Furniture F2?	Pset_Warranty.WarrantyPeriod **Pset_Warranty.WarrantyStartDate** Pset_Warranty.IsExtendedWarranty	0.8162 0.8069 0.7700
Is railing R3 outside the building?	Pset_RailingCommon.Diameter Pset_RampFlightCommon.ClearWidth **Pset_RailingCommon.IsExternal**	0.8051 0.7784 0.7654
Is there any ramp which is slippery?	Pset_RampCommon.RequiredSlope **Pset_RampCommon.HasNonSkidSurface** Pset_RampFlightCommon.CounterSlope	0.7888 0.7874 0.7702
What amount of solar is transmitted through shade S1?	Pset_MaterialOptical.SolarTransmittance **Pset_ShadingDeviceCommon.SolarTransmittance** Pset_DoorWindowGlazingType.SolarHeatGainTransmittance	0.8188 0.8135 0.7930

Table 3. Queries where the correct IFC property was not retrieved at the top position. Each row lists the user query along with the top three retrieved candidates. The correct property is highlighted in bold.

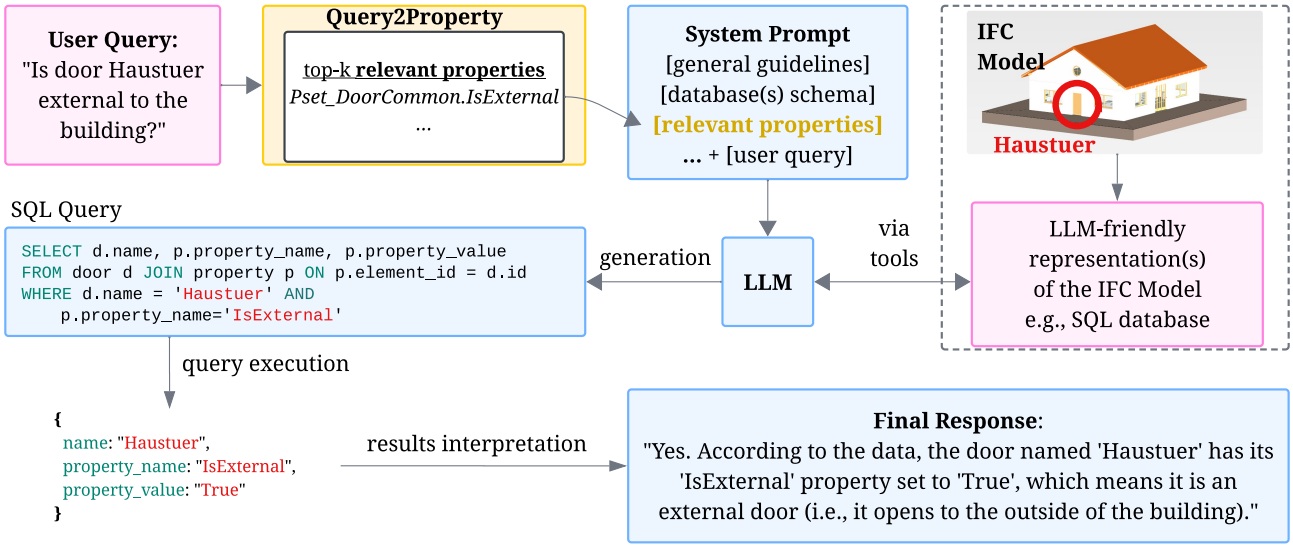

Figure 3. An illustration of integrating Query2Property into an LLM-BIM workflow. The IFC model is the “FZK House”, provided by the Institute for Automation and Applied Informatics (IAI) / Karlsruhe Institute of Technology (KIT). The LLM used is OpenAI’s GPT OSS 120B available at `https://huggingface.co/openai/gpt-oss-120b`. The database is implemented in SQLite, where IFC classes such as *IfcDoor*, *IfcSpace*, and *IfcWall* each correspond to their own tables (e.g., door, room, wall). A separate property table stores the property names and values for each building element. The LLM is instructed to generate and execute SQL queries to access the BIM data in the database and to produce a natural-language response as its final answer.

the correct *IsTempered* property, as both appear within the same glazing property domain and share lexical features. A particularly illustrative case is the query “What amount of solar is transmitted through shade S1?”, where the top-ranked property *SolarTransmittance* from *Pset_MaterialOptical* was selected instead *Pset_ShadingDeviceCommon*. These two properties are nearly identical in their textual descriptions, differing only in their applicable entities, as shown below[11]:

- ***Pset_MaterialOptical.SolarTransmittance****: The ratio of incident solar radiation that directly passes through a system (also named τ_e). Note the following equation: Asol + Rsol + Tsol = 1. Applicable to: IfcMaterial.*
- ***Pset_ShadingDeviceCommon.SolarTransmittance****: The ratio of incident solar radiation that directly passes through a system (also named τ_e). Note the following equation: Asol + Rsol + Tsol = 1. Applicable to: IfcShadingDevice, IfcShadingDeviceType.*

Such cases show that ranking inconsistencies often arise not from failure to capture semantics, but from property definition overlaps and missing lexical variants.

[11] Definitions are quoted from the IFC 4.3.2 Documentation.

w/ or w/o sample query	Top-Ranked Property	Similarity
(no sample query)	Pset_MemberTypePost.LoadBearingCapacity **Pset_ColumnCommon.LoadBearing** Pset_ReinforcementBar...Column.ReinforcementBarType	0.7839 0.7761 0.7739
+ Sample Query: “Is this column designed to bear loads?”	**Pset_ColumnCommon.LoadBearing**	**0.7990**
+ Sample Query: “Does this column carry weight in the building?”	**Pset_ColumnCommon.LoadBearing**	**0.8018**
+ Sample Query: “Is this column a structural column?”	**Pset_ColumnCommon.LoadBearing**	**0.8033**

Table 4. Impact of sample queries on retrieval performance for the query “Is column C1 a load-bearing column?”

Our overall observations show that it is necessary to enrich the IFC properties corpus with alias terms and contextual signals derived or curated from domain knowledge. For instance, *FireRating* property can have aliases such as *fire rating*, *fire resistance*, *fire protection level*, *fire load capacity*, *heat resistance*, and *heat resisting capacity*. More importantly, including contextualized usage descriptions (e.g., sample queries for a given property) can help disambiguate between properties that are semantically close but belong to different IFC contexts. To demonstrate the effectiveness of having sample queries, we made 3 variations of *Pset_ColumnCommon.LoadBearing* (refer Table 4). For the query “Is column C1 a load-bearing column?”, initially, the retrieval system did not rank *LoadBearing* from *Pset_ColumnCommon* first, as a competing but incorrect property *LoadBearingCapacity* from *Pset_MemberTypePost* achieved a slightly higher similarity score. However, after introducing sample queries, the correct property consistently emerged as the top-ranked result, with its similarity score increasing from 0.7761 to 0.8033. The presence of phrases such as “bear loads”, “carry weight” or “structural column” seems to provide additional linguistic cues that help disambiguate semantically close but contextually distinct properties. Similarly, we introduced the sample query “How much solar radiation passes through this shade?” to *Pset_MaterialOptical.SolarTransmittance*. For the query “What amount of solar is transmitted through shade S1?”, the similarity increased from 0.8135 to 0.8384, making the correct property the top retrieved choice. These observations provide a foundation for more systematic approaches to enhancing IFC property retrieval via knowledge-augmented representations.

5. Conclusion

In this study, we introduced Query2Property, a semantic retrieval system for mapping natural language queries to IFC properties using embedding-based similarity. To evaluate its effectiveness, we curated a benchmark of 55 representative BIM queries covering diverse element types and property domains. The method achieved a top-1 accuracy of 87.27% and top-3 accuracy of 100%. These results suggest that the method effectively captures the semantic alignment between user intent and IFC semantics. Failing queries mostly involved semantically related but contextually adjacent properties, pointing to the need for richer contextual and lexical representations.

As future work, we plan to expand the IFC properties corpus with alias terms, example queries, and entity-specific context to further improve retrieval precision in real-world BIM applications. The current study focuses on the property retrieval stage of an LLM-based BIM querying pipeline. While correct property retrieval is an important prerequisite for accurate SQL generation, the full end-to-end pipeline was not evaluated in this work. Although Query2Property currently uses the multilingual Qwen3 embedding model, our experiments so far have been limited to English property descriptions. A promising direction is to explore its performance on multilingual data, investigating whether a multilingual IFC properties corpus is needed to support accurate retrieval for queries in other languages. Furthermore, the area would benefit from a more comprehensive benchmark extending beyond the 55 natural language queries introduced in this work. Such a benchmark would allow comparative studies within the domain. In addition, IFC schema-specific embedding models could be trained and evaluated for their performance, and distilled models could be explored to improve search efficiency. Currently, the search time is $\approx$0.16 seconds per query.

Acknowledgements

This work was supported by the Australian Research Council (ARC) Industry Transformation Research Hub (ITRH) for Resilient and Intelligent Infrastructure Systems (RIIS) under Grant: ARC ITRP IH210100048, and was done in collaboration with FrontierSI.

References

Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F. L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S. et al., 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Arya, S., Mount, D. M., Netanyahu, N. S., Silverman, R., Wu, A. Y., 1998. An optimal algorithm for approximate nearest neighbor searching fixed dimensions. *Journal of the ACM (JACM)*, 45(6), 891–923.

Cao, H., 2024. Recent advances in text embedding: A Comprehensive Review of Top-Performing Methods on the MTEB Benchmark. *arXiv preprint arXiv:2406.01607*.

Choi, Y., Baek, J., Hwang, S. J., 2025. System Prompt Optimization with Meta-Learning. *arXiv preprint arXiv:2505.09666*.

Conneau, A., Kiela, D., 2018. Senteval: An evaluation toolkit for universal sentence representations. *arXiv preprint arXiv:1803.05449*.

Cuconasu, F., Trappolini, G., Siciliano, F., Filice, S., Campagnano, C., Maarek, Y., Tonellotto, N., Silvestri, F., 2024. The power of noise: Redefining retrieval for rag systems. *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 719–729.

Diakite, A. A., Díaz-Vilariño, L., Biljecki, F., Isikdag, Ü., Simmons, S., Li, K., Zlatanova, S., 2022. IFC2IndoorGML: An open-source tool for generating IndoorGML from IFC. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 43, 295–301.

Douze, M., Guzhva, A., Deng, C., Johnson, J., Szilvasy, G., Mazaré, P.-E., Lomeli, M., Hosseini, L., Jégou, H., 2024. The Faiss library.

Du, C., Esser, S., Nousias, S., Borrmann, A., 2024. Text2BIM: Generating building models using a large language model-based multi-agent framework. *arXiv preprint arXiv:2408.08054*.

Du, S., Gao, Y., Tong, W., Weng, Y., 2025. Towards agent: Llm-based framework for robotic construction by leveraging ifc. *ISARC. Proceedings of the International Symposium on Automation and Robotics in Construction*, 42, IAARC Publications, 358–365.

Eastman, C. M., 2011. *BIM handbook: A guide to building information modeling for owners, managers, designers, engineers and contractors*. John Wiley & Sons.

Enevoldsen, K., Chung, I., Kerboua, I., et al., 2025. MMTEB: Massive Multilingual Text Embedding Benchmark. *arXiv preprint arXiv:2502.13595*. https://arxiv.org/abs/2502.13595.

Hasan, A. M., Torky, A. A., Rashed, Y. F., 2019. Geometrically accurate structural analysis models in BIM-centered software. *Automation in construction*, 104, 299–321.

Iranmanesh, S., Saadany, H., Vakaj, E., 2025. LLM-assisted Graph-RAG Information Extraction from IFC Data. *arXiv preprint arXiv:2504.16813*.

Irizarry, J., Karan, E. P., Jalaei, F., 2013. Integrating BIM and GIS to improve the visual monitoring of construction supply chain management. *Automation in construction*, 31, 241–254.

Isikdag, U., Zlatanova, S., Underwood, J., 2011. An opportunity analysis on the future role of BIMs in urban data management. *Urban and Regional Data Management–UDMS Annual*, 25–36.

Jegou, H., Douze, M., Schmid, C., 2010. Product quantization for nearest neighbor search. *IEEE transactions on pattern analysis and machine intelligence*, 33(1), 117–128.

Kim, J. B., Jeong, W., Clayton, M. J., Haberl, J. S., Yan, W., 2015. Developing a physical BIM library for building thermal energy simulation. *Automation in construction*, 50, 16–28.

Lamsal, R., Read, M. R., Karunasekera, S., 2024. CrisisTransformers: Pre-trained language models and sentence encoders for crisis-related social media texts. *Knowledge-Based Systems*, 296, 111916.

Lamsal, R., Read, M. R., Karunasekera, S., 2025. Langformers: Unified NLP Pipelines for Language Models. *arXiv preprint arXiv:2504.09170*.

Liu, L., Li, B., Zlatanova, S., van Oosterom, P., 2021. Indoor navigation supported by the Industry Foundation Classes (IFC): A survey. *Automation in Construction*, 121, 103436.

Ma, Z., Liu, Z., 2018. Ontology-and freeware-based platform for rapid development of BIM applications with reasoning support. *Automation in Construction*, 90, 1–8.

Malkov, Y. A., Yashunin, D. A., 2018. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE transactions on pattern analysis and machine intelligence*, 42(4), 824–836.

Muennighoff, N., Tazi, N., Magne, L., Reimers, N., 2022. Mteb: Massive text embedding benchmark. *arXiv preprint arXiv:2210.07316*.

Pacheco, L., Berkmiller, C., 2024. Bimconverse – graphrag for ifc natural language queries.

Pan, J. J., Wang, J., Li, G., 2024. Survey of vector database management systems. *The VLDB Journal*, 33(5), 1591–1615.

Thakur, N., Reimers, N., Rücklé, A., Srivastava, A., Gurevych, I., 2021. Beir: A heterogenous benchmark for zero-shot evaluation of information retrieval models. *arXiv preprint arXiv:2104.08663*.

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F. et al., 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Wang, H., Pan, Y., Luo, X., 2019. Integration of BIM and GIS in sustainable built environment: A review and bibliometric analysis. *Automation in construction*, 103, 41–52.

Wang, J., Yang, Q., Chen, Y., 2024. A large language model–based approach for automatically optimizing bim. *2024 43rd Chinese Control Conference (CCC)*, IEEE, 8518–8523.

Wang, L., Yang, N., Huang, X., Jiao, B., Yang, L., Jiang, D., Majumder, R., Wei, F., 2022. Text embeddings by weakly-supervised contrastive pre-training. *arXiv preprint arXiv:2212.03533*.

Won, J., Cheng, J. C., 2017. Identifying potential opportunities of building information modeling for construction and demolition waste management and minimization. *Automation in Construction*, 79, 3–18.

Yang, J., Jin, H., Tang, R., Han, X., Feng, Q., Jiang, H., Zhong, S., Yin, B., Hu, X., 2024. Harnessing the power of llms in practice: A survey on chatgpt and beyond. *ACM Transactions on Knowledge Discovery from Data*, 18(6), 1–32.

Zhang, Y., Li, M., Long, D., Zhang, X., Lin, H., Yang, B., Xie, P., Yang, A., Liu, D., Lin, J., Huang, F., Zhou, J., 2025. Qwen3 Embedding: Advancing Text Embedding and Reranking Through Foundation Models. *arXiv preprint arXiv:2506.05176*.

Zhao, W. X., Zhou, K., Li, J., Tang, T., Wang, X., Hou, Y., Min, Y., Zhang, B., Zhang, J., Dong, Z. et al., 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223*, 1(2).

Zhao, X., Zhou, X., Li, G., 2024. Chat2data: An interactive data analysis system with rag, vector databases and llms. *Proceedings of the VLDB Endowment*, 17(12), 4481–4484.

Zhu, J., Wong, M. O., Nisbet, N., Xu, J., Kelly, T., Zlatanova, S., Brilakis, I., 2025. Semantics-based connectivity graph for indoor pathfinding powered by IFC-Graph. *Automation in Construction*, 171, 106019.