Kalpa Publications in Computing

Volume 22, 2025, Pages 623–633

Proceedings of The Sixth International Conference on Civil and Building Engineering Informatics

Automatic Compliance Checking of BIM Models for Architecture and Fire Protection Based on Knowledge Graphs and Machine Learning

Sihao Li[1], Guangyao Chen[1], Yangze Liang[1] and Zhao Xu[1]

[1] Southeast University, Jiangsu, China.

lish@seu.edu.cn, chengy@seu.edu.cn, lyangz@seu.edu.cn, xuzhao@seu.edu.cn

Abstract

Compliance checking of Building Information Modeling (BIM) models is a critical process throughout the construction lifecycle, particularly during the design phase. Building design often involves the integration of multiple disciplines and complex spatial relationships, leading to errors. The growing volume and complexity of information embedded in BIM models have further complicated compliance checking. Traditional manual methods are not only time-consuming but also prone to mistakes. To address these challenges, this study proposes an integrated conceptual framework for automated BIM compliance checking, leveraging knowledge graph (KG) and machine learning. The framework aims to convert unstructured clauses in Chinese building standards into structured, interpretable, and extractable data, enabling the automatic detection of design errors in BIM models. The framework incorporates several key components. First, it constructs a knowledge graph by developing ontologies for Chinese building standards and training semantic role annotation models. A data extraction pipeline is designed using the Dynamo module in Revit to retrieve relevant information from BIM models. Finally, compliance checking logic is defined using Java to establish rules for matching the extracted building standard knowledge with BIM model information. The feasibility of this automated compliance-checking framework was validated using BIM models from two real-world projects, demonstrating its potential to streamline the compliance process and reduce errors in building design.

I. Hkust, J. Cheng and Y. Yantao (eds.), ICCBEI 2025 (Kalpa Publications in Computing, vol. 22), pp. 623–633

1 Introduction

The design review process in building construction has evolved from hand-drawn blueprints to CAD (Balachandran et al., 1991) and BIM (Liu et al., 2022; Solihin et al., 2020), which is now integral to project approvals in many cities. In China, the rapid growth of BIM data supports digital advancements and smart city initiatives (Dimyadi and Amor, 2013). BIM compliance checks ensure safety, regulatory adherence, and public interest compliance but face challenges due to the complexity of BIM data. Manual reviews are inefficient, require high expertise, and can introduce biases (Li et al., 2024).

Automated Compliance Checking (ACC) offers a solution to these challenges (Noardo et al., 2022). Researchers have focused on standardizing regulations (Ismail et al., 2017) and integrating model data (Zhang and El-Gohary, 2017), leading to tools like Solibri Model Checker (Solihin et al., 2020) and Autodesk Navisworks (Hjelseth, 2015). These systems handle tasks like clash detection and stairway inspections but struggle with complex logic and interrelated data. Enriching model information remains a challenge, and current systems rely on hard-coded rules, requiring extensive programming and frequent updates to match evolving standards (Zhang and El-Gohary, 2017). While semantic inspection methods offer flexibility (Beach et al., 2015; Liebich et al., 2004), they fall short in automating rule construction and updates.

Knowledge graphs (KG) (Singhal, 2012), first introduced by Google in 2012, excel in semantic reasoning and flexible knowledge representation (Li et al., 2021). KGs can structure and formalize building standards, transforming unstructured clauses into computable data for machine interpretation. Soft-coding techniques and Information Extraction (IE) enable automated rule library creation, empowering domain experts to define rules without programming, improving BIM compliance efficiency.

However, existing KGs have limited scope, poor transferability, and focus primarily on representation rather than reasoning. Addressing these gaps, this study proposes an ACC framework combining ontology, NLP, and deep learning to automate KG construction for Chinese building standards. Logical relationships within KGs are used to interpret standards, while processes in Dynamo and Java establish links between BIM models and standards, automating compliance checks. An intelligent platform further enhances BIM review efficiency and accuracy.

2 Method

This study proposes an integrated conceptual framework for building engineering that combines NLP, machine learning, and knowledge graph techniques to automate the compliance checking of BIM models. The aim is to improve the efficiency and accuracy of this process through automation. The methods used in this research are designed to be easily updated and transferable, providing a pathway for the future development of knowledge graphs across various building engineering disciplines. An overview of the proposed methodology is shown in Fig. 1, along with explanatory images for key concepts and steps. The process consists of three interrelated parts: knowledge graph construction, BIM model information extraction, and matching BIM models to the knowledge graph for compliance checking.

The first step is constructing the building standard knowledge graph, which begins with analyzing standards and studying relevant texts in the field. Based on this analysis, an ontology of architectural and fire protection elements is created. The building standard documents are preprocessed, with semantic role labels applied to them. A corpus of building standards is then built and annotated with these labels. A deep learning model, BERT-BiLSTM-CRF, is trained to automatically label the building standard texts. Syntactic analysis is performed using CFG, and knowledge graph triples are

extracted from the annotated texts. Finally, Neo4j is used to construct and store the knowledge graph, where nodes and attributes are designed to preserve the logical relationships from the building standards, enhancing the graph's reasoning capabilities.

Next, BIM model checking data is predefined, and a data extraction method based on Dynamo is created to obtain the necessary files for checking. A matching algorithm is then developed to represent the logical rules of the knowledge graph triples, leading to an automated compliance checking method for BIM models. An intelligent BIM model checking platform is built and validated through two engineering case studies. The platform successfully performs automated compliance checking for both architectural and fire protection aspects of BIM models, generating comprehensive checking reports. Since the building standards used in this study are based on Chinese regulations, the research focuses on Chinese texts, with comparative English translations provided for clarity.

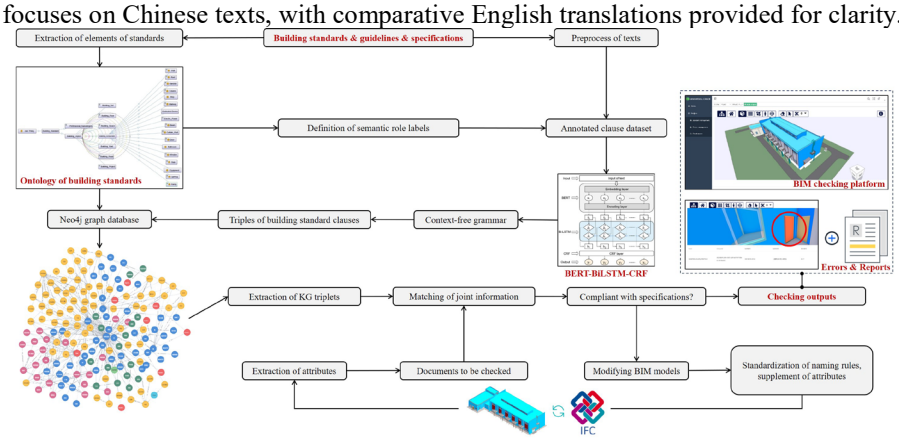

Figure 1: Overview of the proposed methodology framework

Building design and construction standards form the foundation for compliance checks. This study selects several widely used standards in architecture and fire protection (in Chinese) with broad coverage, comprehensive content, and extensive applicability. These standards include the "Unified Standard for Civil Building Design (GB 50352-2019)", "Residential Building Code (GB 50368-2005)", "Design Code for Residential Buildings (GB 50096-2011)", "Fire Protection Design Code for Buildings (GB 50016-2014)", "Technical Standard for Fire Emergency Lighting and Evacuation Indication Systems (GB 51309-2018)", "Fire Protection Design Code for Automobile Garages, Repair Garages, and Parking Lots (GB 50067-2014)", and "Fire Protection Design Code for Interior Decoration of Buildings (GB 50222-2017)". These standards serve as the data foundation for constructing the ontology of building standards, which is then used to build the knowledge graph (KG) through information extraction and syntactic analysis.

The process of constructing the KG in this study follows these steps: First, the ontology of building standards is developed. Then, the building standard texts undergo preprocessing. Based on this ontology, custom semantic role labels are designed, and the preprocessed texts are annotated with these labels to create a labeled dataset. A semantic role labeling model is trained using this dataset to automatically label the building standard texts. Afterward, syntactic analysis is performed on the annotated texts, which are parsed to extract data triples for the knowledge graph. Finally, the knowledge graph is constructed by combining the pattern layer and data layer.

2.1 Ontology Construction

The clauses in building standards define specific constraints for engineering design and construction and serve as the foundational knowledge for building the ontology and knowledge graph in this study. These clauses are categorized into mandatory and non-mandatory types. Mandatory

clauses are those that must be followed, while non-mandatory clauses represent guidelines that should generally be adhered to under typical circumstances. Mandatory clauses take precedence over non-mandatory ones. In this study, a hierarchical analysis of the building standard knowledge and the logical relationships between various construction elements was conducted. A seven-step method was employed to construct the building standard ontology, which forms the basis for semantic role labeling and knowledge graph development.

The focus of this research is on the compliance check of BIM models, with the ontology's scope defined as architecture and fire control. By integrating concepts, terminology, and attributes from existing ontologies such as Ontolingua and DAML, the construction of the building standard ontology in this study is enriched to improve its efficiency and completeness. The key terms included in the selected criteria are counted to provide a list of all terms.

The ontology development in this study follows a top-down approach. Initially, key concepts are extracted from the selected standard texts to form the classes within the ontology. Next, a hierarchical class structure is created through a stepwise subdivision from top to bottom. The building code is designated as the parent class, which is divided into two main categories: professional standards and building objects. These categories are then further subdivided into subclasses, each containing various underlying objects. Some of these underlying objects can be further classified into different types of instances, while others are directly treated as instances that do not require further subdivision. The professional standards category includes building standards and fire control standards, while the building objects category is divided into several subclasses such as project, building, space, site, storey, and component. A partial view of the constructed ontology for building standard texts is shown in Fig. 2.

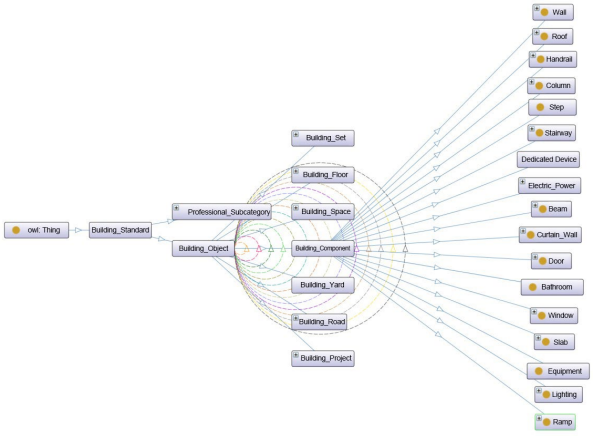

Figure 2: Ontology classes for building standards

2.2 Recognition of Semantic Role Labels

Based on the constructed ontology and Chinese word segmentation method, 6 semantic role labels have been customized, as shown in Table 1.

Abbreviation	Definition	Interpretation
JD	Building object	Including building project, building site, building, building floor, building component, building space, building road, etc., referring to the semantic entities in the triples.
DS	Object property	Elements lower than JD, which are object properties of JD, including containment relationships and spatial relationships, referring to the semantic relationships between entities in the triples.
SS	Data property	Elements lower than JD, which are data properties of JD, including numerical properties, non-numerical properties, and measures, referring to the semantic properties in the triples.
SXZ	Property value	Requirements and conditions connected to property SS, with data formats such as Float, String, Boolean, referring to the semantic relationships between entities and properties in the triples.
BJ	Comparative term	Comparative relationship between property SS and requirement SXZ, including "greater than", "equal to", "less than", "should not be less than", and other comparative terms, referring to the semantic relationships between properties and property values in the triples.
LJ	Logical term	Expressing logical semantic relationships such as "parallel", "optional" including "and", "both", "except for", etc.

Table 1: Labels of semantic roles

Selected building code provisions are annotated using the BIO (Begin-inside-outside) sequence labeling method, where "B" indicates the beginning, "I" represents the middle position, and "O" denotes that the token does not belong to any label type. For example, the sentence "Residential buildings with a height greater than 33m should be equipped with fire elevators" is annotated, as shown in Figure 3. In this study, a total of 11,174 lines of data were annotated, and the deep learning model's training, development, and test datasets were constructed with an 8:1:1 ratio.

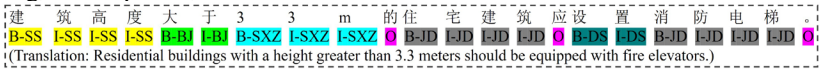

Figure 3: BIO sequence labeling data

After constructing the dataset, the BERT-BiLSTM-CRF model is introduced to automatically label the semantic roles of building standard texts. The model architecture consists of three levels. First, the pre-trained BERT layer is used to generate embedding vectors and output encoded sequence vectors T=(t1,t2,…,tn). Then, the Bi-LSTM layer extracts semantic features from the text data by integrating contextual information, and the resulting bidirectional output sequence vector hi indicates the probability of each position being recognized with a BIO label. Finally, the CRF layer adjusts and optimizes the sequence vectors, outputting the most probable label sequence Y=(y1,y2,…,yn).

2.3 Extraction of the Triplets

In the triples, entities and attributes correspond to JD, DS, and SS labels, which can be achieved by extracting entities under each label based on the output of the statistical model. Additionally, Context-Free Grammar (CFG) is used to perform syntactic analysis on the labeled results, helping to identify and refine the extraction of more complex relationships in the triples.

CFG is a formal language grammar that describes all possible string combinations generated by a probabilistic process, with sentence generation following predefined rules. By analyzing common

semantic representations in building standard texts, four types of rules are classified after CFG parsing and adjustment:

<JD-SS-BJ-SXZ>: Constraints on certain numerical attributes of building objects. For example, "The fire resistance grade of a high-rise factory should not be lower than Grade II."

<JD-DS-SXZ>: Constraints on measures or non-numeric attributes of data. For example, "The floor of the toilet should adopt waterproof construction measures."

<JD-DS-JD>: Constraints on the existence of certain building objects, equipment, or systems, or constraints on category matching for building objects. For example, "Residential buildings should have a lighting power supply system."

<JD-JD-DS>: Constraints on the spatial relationship between certain building objects. For example, "Electrical wiring pipes should not be set next to the toilet."

Taking the first rule as an example, the automatic semantic labeling result is shown in Figure 4. By directly applying CFG, the syntactic tree is generated. According to the part-of-speech labeling rules, "high-rise factory" is a noun phrase (NP) terminal, where "high-rise factory" is the entity to be extracted. Both "high-rise factory" and "fire resistance grade" are labeled as nouns (N), making it difficult to distinguish between the entity and attribute. At the same time, "should not be lower than" is split into "should not" and "be lower than," corresponding to the auxiliary verb (Aux) and verb (V). For the knowledge graph triple, "should not be lower than" should be extracted as a whole, representing the relationship between the attribute and its value.

高	层	厂	房	的	耐	火	等	级	不	应	低	于	二	级	。
B-JD	I-JD	I-JD	I-JD	O	B-SS	I-SS	I-SS	I-SS	B-BJ	I-BJ	I-BJ	I-BJ	B-SXZ	I-SXZ	O

(Translation: The fire resistance rating of high-rise buildings should not be lower than level 2.)

Figure 4: Automatic annotation results of semantic labels

Therefore, the triples cannot be directly obtained from the syntactic tree at this point. By combining the semantic role labels with the syntactic tree, the semantic labels were changed. The concept of JD corresponds to the entity in the triple, SS corresponds to the data attribute, and SXZ corresponds to the attribute value connected to the SS attribute. The knowledge graph triple for this rule and its minimal unit triple are shown in Figure 5. Using a similar approach, adjustments are made to the labeled results, and the triples for the standard text are obtained.

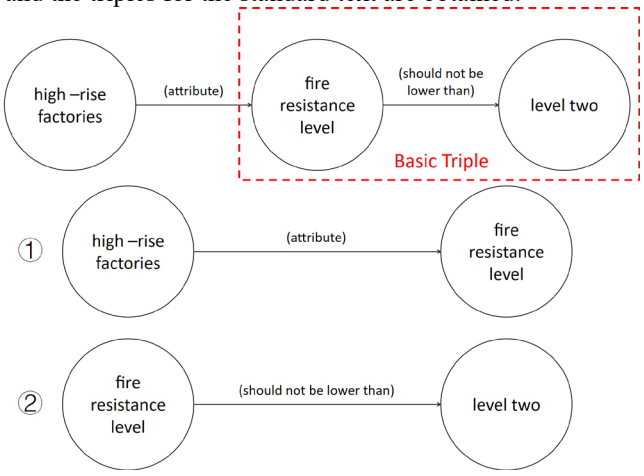

Figure 5: Knowledge graph triplet

2.4 Generation of Knowledge Graph

After obtaining the triples, the knowledge graph is generated and stored using the Neo4j database. The ontology triples constructed in Protégé software are parsed and converted into ".turtle files ,"

which are then imported into Neo4j to create the knowledge graph network. To preserve logical relationships and meet the requirements for subsequent automatic review of BIM models, the knowledge triples of the clauses are organized. Additional relationships are created, such as (standard) -[clause]-> (clause number), (clause number) -[clause type]-> (mandatory/non-mandatory), and (clause number) -[clause constraint]-> (building object). Entity nodes corresponding to each clause are created individually, as shown in Figure 6.

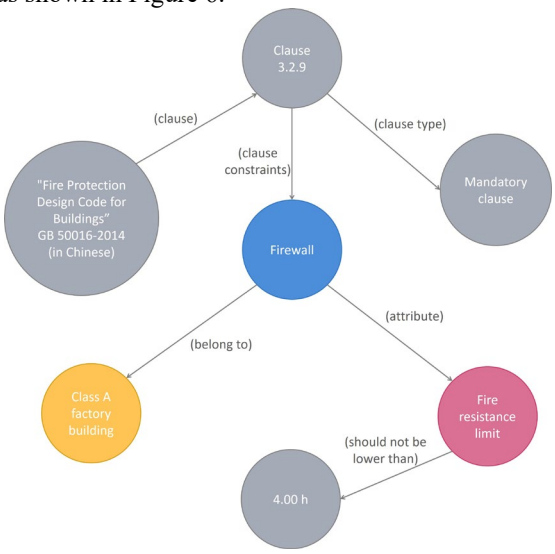

Figure 6: Example of a relational triplet

2.5 Extraction of Data

In this study, the process of automatically checking BIM models involves data extraction and matching. Dynamo is used to extract specific data from BIM models, covering tasks such as adding, extracting, and completing model data, as shown in Fig. 7 and Fig. 8. Afterward, the extracted data is matched with the knowledge stored in the knowledge graph (KG) using triples, enabling the compliance check of the model. Although Dynamo's visual nodes can extract certain parameters from model elements, their capabilities are limited, as some parameters cannot be directly accessed using the built-in nodes. To overcome this, this study integrates Dynamo's visual nodes with PythonScript to call the Revit API for data extraction and compliance checking.

Some parameters necessary for compliance checks are not included in the default element parameters. Therefore, it is essential to define and add custom attribute parameters based on the specific checking requirements to ensure comprehensive model information extraction. In Revit, project parameters are a useful tool, acting as containers for element data. These parameters can be customized to include various attributes, such as fire resistance level, combustibility, and mitigation measures. Additionally, project parameters allow users to flexibly define element attributes based on the specific requirements of the compliance checks. Revit elements are primarily categorized into Categories, Families, Family Types, and Family Instances. Family parameters are divided into type parameters and instance parameters. Type parameters are shared across all instances of a family type, meaning that modifying them will alter all instance parameters within that family. Conversely, modifying the instance parameters of a specific family instance will only affect that instance. In this study, custom parameters are added as instance parameters to facilitate the assignment and extraction of parameters for various components. The batch addition of these parameters follows a programming workflow in Dynamo. When creating a new family instance, the instance parameters will automatically include the custom parameters, allowing the required data to be input into the model.

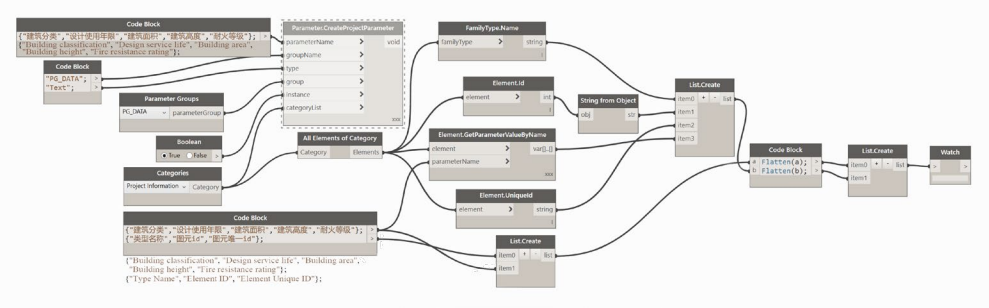

Figure 7: Extraction of project information parameters

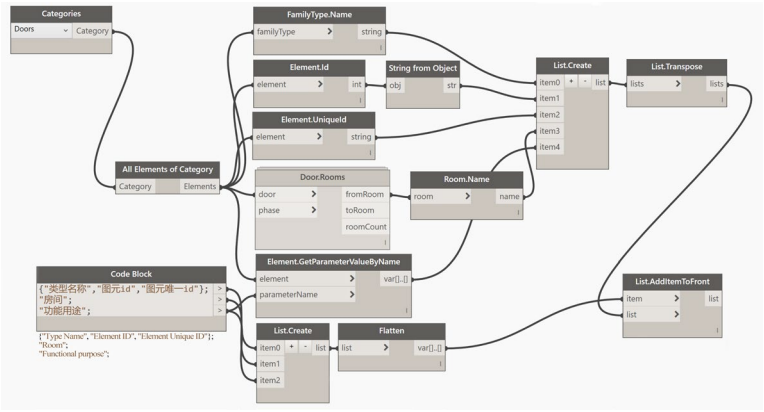

Figure 8: Information extraction of rooms associated with doors and windows

3 Results

The experimental environment for the BERT-BiLSTM-CRF model in Section 2.2 for automatic semantic role labeling consists of a DELL R730 server, 64-bit Linux system, Visual Studio Code (Version 1.71), TensorFlow 1.13.2 framework, and an NVIDIA T4 GPU. The performance evaluation criteria of the model are accuracy, precision, recall and F1 according to the confusion matrix, and the calculation formulas are shown in Formula (1), (2) and (3). The performance of the semantic role annotation model for extracting various semantic labels is shown in Table 2.

$$Precision = \frac{TP}{TP + FP} \tag{1}$$

$$Recall = \frac{TP}{TP + FN} \tag{2}$$

$$F1 = 2 \times \frac{precision * recall}{precision + recall} \tag{3}$$

Label	Precision (%)	Recall (%)	F1
BJ	100.00	100.00	1.000
DS	77.70	87.50	0.824
JD	96.55	95.45	0.960
LJ	100.00	95.24	0.976
SS	95.12	97.50	0.963
SXZ	95.65	95.65	0.957

Table 2: FID score statistics for different models

A web platform integrating the knowledge graph (KG) and the described algorithms has been developed to facilitate BIM model compliance checking. The platform adopts a Browser/Server (B/S) architecture and consists of three layers, including Data Access Layer (DAL), Business Logic Layer (BLL), and User Interface (UI) Layer. The DAL handles storage for standard ontologies, the KG, and BIM model databases, enabling access through web service interfaces, drivers, and open standards. The BLL incorporates both business functions and logical entities. Business functions include view, system, model, and project management, while logical entities encompass web servers, database servers, engines, and scripts. The platform utilizes HTML5 and WebGL for its base structure, with functionality implemented through CSS and JavaScript (JS) scripts. Users interact with BIM data on the visualization platform via the UI, performing operations such as adding, deleting, editing, querying, and generating compliance reports.

A factory engineering project was used to verify the feasibility of this method, as shown in Fig. 9. For the building project, elements such as "steps," "stairs," "ramps," "railings," "doors," "firewalls," "roof access points," and "rooms" were selected for automated architectural and fire safety checks. The relevant data from the models was extracted and then analyzed and assessed using matching algorithms.

Figure 9: Compliance check platform for BIM models

Finally, an automated compliance report is generated for each BIM model. Non-compliant components are identified in the visualized model by their unique IDs and highlighted in different colors. For instance, as shown in Fig. 10, a door in the industrial building is flagged as non-compliant. According to Article 6.2.7 of the "Fire Protection Code for Building Design" (Chinese), doors leading into electrical rooms must be Class A fire-resistant doors. In this model, the door opens into the building's interior in the electrical room, but its "functional" value is listed as "Class B fire-resistant door," which does not meet the standard.

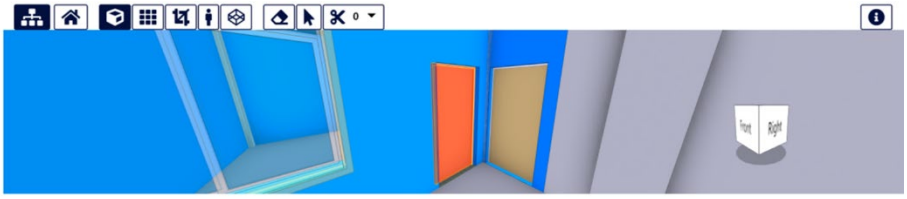

Figure 10: Highlighted door in the industry building

4 Conclusions

Based on the analysis of standards and BIM model data, an integrated framework for automated compliance checking of BIM models utilizing NLP and knowledge graphs has been developed. The feasibility and effectiveness of this approach were validated through two case studies. The theoretical value of the research lies in presenting a method for automating BIM model compliance checking and enhancing the automation and transferability of domain-specific knowledge graph construction. Its practical significance is reflected in the development and successful application of a BIM model checking system, offering both a theoretical foundation and practical reference for related research and applications. The main research contributions are as follows:

(1) Developed an ontology for the building standards domain using the "seven-step method" and created a semantic role labeling model for building standard texts based on BERT-BiLSTM-CRF.

(2) Integrated the semantic role labeling model with CFG-based analysis methods and constructed a knowledge graph stored in the Neo4j database, incorporating logical markers to strengthen the graph's reasoning capabilities.

(3) Developed script tools using Dynamo and the Revit API to enable parameter addition, extraction, and supplementation for BIM model data.

(4) Formulated logical rule expressions for information matching and proposed a method to match knowledge graph triples with BIM model data, enabling automated compliance checking.

5 Acknowledgments

The authors gratefully acknowledge the financial support provided by the "National Natural Science Foundation of China (72071043)," the "National Key Research and Development Program of China (2022YFC3803600)," and the "SEU Innovation Capability Enhancement Plan for Doctoral Students (CXJH_SEU 25089)." Additionally, the authors thank the author of the template for the manuscripts of the proceedings of the 30th International Symposium on Automation and Robotics in Construction and Mining (ISARC 2013).

References

Balachandran, M., Rosenman, M. A., and Gero, J. S. (1991). A knowledge-based approach to the automatic verification of designs from CAD databases. In J. S. Gero (Ed.), Artificial Intelligence in Design '91 (pp. 757-781). Butterworth-Heinemann.

Beach, T. H., Rezgui, Y., Li, H., et al. (2015). A rule-based semantic approach for automated regulatory compliance in the construction sector. Expert Systems with Applications, 42(12), 5219-5231.

Dimyadi, J., and Amor, R. (2013). Automated building code compliance checking: Where is it at? Proceedings of CIB WBC 2013, 172-185.

Hjelseth, E. (2015). BIM-based model checking (BMC). Building Information Modeling–Applications and Practices, 33-61.

Ismail, A. S., Ali, K. N., & Iahad, N. A. (2017). A review on BIM-based automated code compliance checking system. 2017 international conference on research and innovation in information systems (icriis) (pp. 1-6). IEEE.

Li, S., Wang, J., and Xu, Z. (2024). Automated compliance checking for BIM models based on Chinese-NLP and knowledge graph: An integrative conceptual framework. Engineering, Construction and Architectural Management.

Liebich, T., Wix, J., and Qi, Z. (2004). Speeding-up the submission process: The Singapore e-plan checking project offers automatic plan checking based on IFC. International Conference on Construction Information Technology (INCITE 2004), 245-252.

Liu, H., Cheng, J. C. P., Gan, V. J. L., et al. (2022). A novel data-driven framework based on BIM and knowledge graph for automatic model auditing and quantity take-off. Advanced Engineering Informatics, 54, 101757.

Li, X., Lyu, M., Wang, Z., et al. (2021). Exploiting knowledge graphs in industrial products and services: A survey of key aspects, challenges, and future perspectives. Computers in Industry, 129, 103449.

Noardo, F., Wu, T., Arroyo Ohori, K., et al. (2022). IFC models for semi-automating common planning checks for building permits. Automation in Construction, 134, 104097.

Singhal, A. (2012). Introducing the knowledge graph: things, not strings. Official google blog, 5(16), 3.

Solihin, W., Dimyadi, J., Lee, Y.-C., et al. (2020). Simplified schema queries for supporting BIM-based rule-checking applications. Automation in Construction, 117, 103248.

Zhang, J., and El-Gohary, N. M. (2017). Integrating semantic NLP and logic reasoning into a unified system for fully-automated code checking. Automation in Construction, 73, 45-57.