*buildings*

MDPI

*Article*

# Automated Checking of Highway Bridge BIM Models Based on Large Language Models

**Yongyi Yang [1], Xiaoping Jing [1,*] and Yan-Ming Liu [2]**

[1] School of Emergency Management, Xihua University, Chengdu 610039, China; 0120200012@mail.xhu.edu.cn
[2] School of Civil Engineering, Southwest Jiaotong University, Chengdu 610031, China; liuyanming@my.swjtu.edu.cn
* Correspondence: jingxiaoping@stu.xhu.edu.cn

**Abstract**

Building Information Modeling (BIM) is widely used in highway bridge engineering, making compliance with complex design specifications crucial. Existing checks rely heavily on manual review, which is time-consuming and inefficient. This study proposes an automated framework using large language models (LLMs) to parse unstructured design specifications and extract structured rules with 79.5% accuracy, stored in a knowledge graph. IFC-formatted BIM component attributes are then compared with these rules to check structural completeness and compliance, achieving 84.4% precision. The results indicate that the framework offers an effective solution for automated rule extraction and has the potential to improve compliance-checking efficiency and accuracy in engineering practice.

**Keywords:** building information modeling; large language model; highway bridge; knowledge graph; automated model checking

## 1. Introduction

With the rapid development of information technology, Building Information Modeling (BIM) has been widely adopted in the engineering sector and has become a core approach driving the digital transformation and intelligent management of construction projects [1]. In highway bridge engineering, BIM is gradually replacing traditional 2D drawings, becoming the primary digital medium for design and construction. BIM models not only provide improved visualization but also integrate geometry, component attributes, and inter-component relationships. However, the increasing complexity of BIM models and the growing volume of design specifications pose significant challenges for efficient and accurate automated checking.

Automated checking of BIM models involves using computational tools to evaluate whether design models comply with engineering codes and standards. Considerable progress has been made internationally and domestically. Eastman et al. [2] pioneered rule-based compliance checking in building design, outlining a four-stage process: rule interpretation and logical representation, model preparation, rule execution, and report generation. Early information extraction methods were mainly rule-based, relying on manually created rules to extract entities and relationships from texts. They required substantial human effort, and their performance depended heavily on the quality of the rules. El-Gohary et al. [3–5] conducted in-depth research on specification information extraction, employing rule-based methods, deep learning, and fully automated compliance frameworks integrating semantic natural language processing (NLP) with logical

check for updates

Academic Editor: Henry Abanda

Received: 27 August 2025
Revised: 14 September 2025
Accepted: 22 September 2025
Published: 25 September 2025

**Citation:** Yang, Y.; Jing, X.; Liu, Y.-M. Automated Checking of Highway Bridge BIM Models Based on Large Language Models. *Buildings* **2025**, *15*, 3465. https://doi.org/10.3390/buildings15193465

reasoning. Xue et al. [6] applied deep learning to part-of-speech tagging in regulatory documents, Machine learning–based methods gradually emerged, such as Support Vector Machines (SVM) [7], Hidden Markov Models (HMM) [8] and K-Nearest Neighbors (KNN) [9]. Peng et al. [10] proposed a framework based on BIM and knowledge graphs (KGs), employing machine learning for information extraction to enable automated compliance verification. However, these methods still require substantial manual involvement in feature engineering, and the effectiveness of information extraction is closely dependent on the quality of the manually constructed features. Moreover, maintaining these systems was technically demanding, and their scalability was limited. These constraints reduced their applicability to large and evolving design specifications.

Despite considerable progress in specification information extraction and automated checking of BIM models, challenges remain in extracting complex rules from engineering specifications, representing them effectively for automated reasoning, and integrating them with BIM model attributes. To address these challenges, this study focuses on the following two main research questions:

**RQ1.** *How can rules be extracted from highway bridge design specifications?*

**RQ2.** *How can the extracted rules be used for automated compliance and structural integrity checking of BIM models?*

To address these challenges, this study employs large language models (LLMs) and natural language processing (NLP) techniques to extract information from design specifications. Unlike existing methods that rely heavily on manual feature engineering and rule coding, our approach directly extracts structured rules from the specifications. This reduces manual work, lowers the operational threshold, and achieves a rule extraction accuracy of 79.5%. The extracted rules are stored in a domain-specific knowledge graph to support intelligent rule representation and efficient querying. The framework can handle multiple specifications with complex conditional logic, providing strong scalability and ease of maintenance. Component attributes from IFC-formatted BIM models are automatically compared with the rules in the knowledge graph to perform structural completeness and compliance checking. Performance evaluation shows an accuracy of 65.2%, precision of 84.4%, recall of 70.7%, and F1 score of 76.9%. These results indicate that the method offers a practical and promising solution for automated rule extraction and can potentially improve the efficiency of compliance-checking processes.

A technical framework for automated checking of highway bridge BIM models is proposed, consisting of four key stages as illustrated in Figure 1: (1) structured transformation of design specifications, (2) extraction of BIM model data, (3) compliance checking via a graph database, (4) automated generation of checking reports. This framework addresses both the technical challenges of specification information extraction and the practical needs of BIM-based bridge design, providing a foundation for digital transformation and high-quality delivery of highway bridge projects.

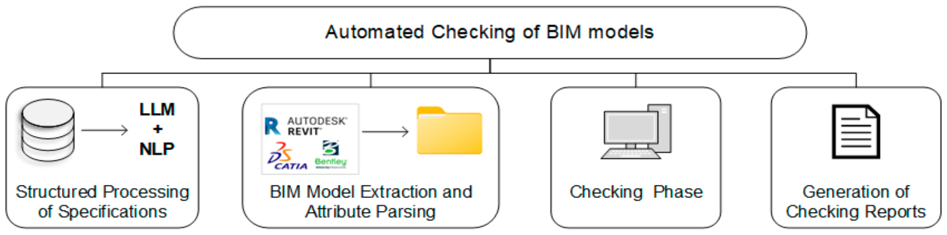

Automated Checking of BIM models
LLM
+
NLP
Structured Processing of Specifications
AUTODESK REVIT
CATIA
Bentley
BIM Model Extraction and Attribute Parsing
Checking Phase
Generation of Checking Reports

**Figure 1.** Framework for automated checking of BIM models.

## 2. Semantic Structuring of Engineering Specifications Using LLMs

### *2.1. Overview of Language Modeling Development*

As illustrated in Figure 2, the development of language modeling can be broadly categorized into four stages. Initially, research focused on language translation and basic grammar parsing, relying on hand-crafted rules with limited adaptability. With the advancement of computational power and the availability of large-scale corpora, Statistical Machine Translation (SMT) [11,12] gradually supplanted rule-based systems. The breakthroughs in deep learning, particularly the introduction of attention mechanisms and the Transformer architecture, significantly accelerated progress in machine translation. In recent years, NLP research has gradually shifted toward cross-modal fusion and reinforcement learning, achieving more comprehensive language understanding [13,14], and evolving from statistical language models to LLM [15].

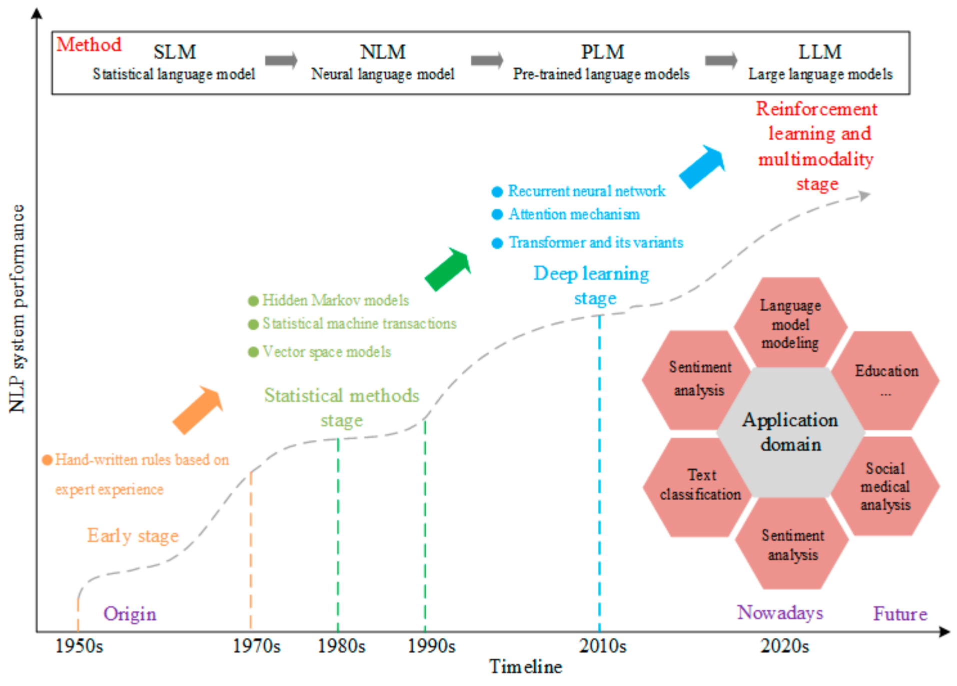


**Figure 2.** Overview of NLP development stages and their applications.

LLMs have billions to trillions of parameters. They can analyze language structures and semantic representations, making them well-suited for processing complex and lengthy engineering specifications. Transformer-based models, in particular, use self-attention mechanisms to capture long-range dependencies in input sequences. This significantly improves the model's ability to handle language representation and long-text modeling. The GPT series [16], a leading example of large models, performs exceptionally well in this regard. Li et al. [17] used LLMs to extract structured data from engineering specifications, achieving 71.6% accuracy. Dagdelen [18] fine-tuned GPT-3 and LLaMA-2 models to extract structured information from the scientific literature. These studies show that LLMs can convert unstructured engineering specifications into structured, machine-readable rules.

### *2.2. LLM-Based Structuring of Design Specifications*

The rapid development and widespread application of large language models (LLMs) have made it possible to decompose engineering specifications and extract structured information from them. This enables the construction of domain-specific knowledge bases, which provide essential support for the automated checking of BIM models. As shown in Figure 3, this study proposes an LLM-driven technical framework for structuring design specifications. The process consists of four main stages: (1) text preprocessing; (2) prompt engineering; (3) knowledge extraction via LLMs; (4) structured data storage.

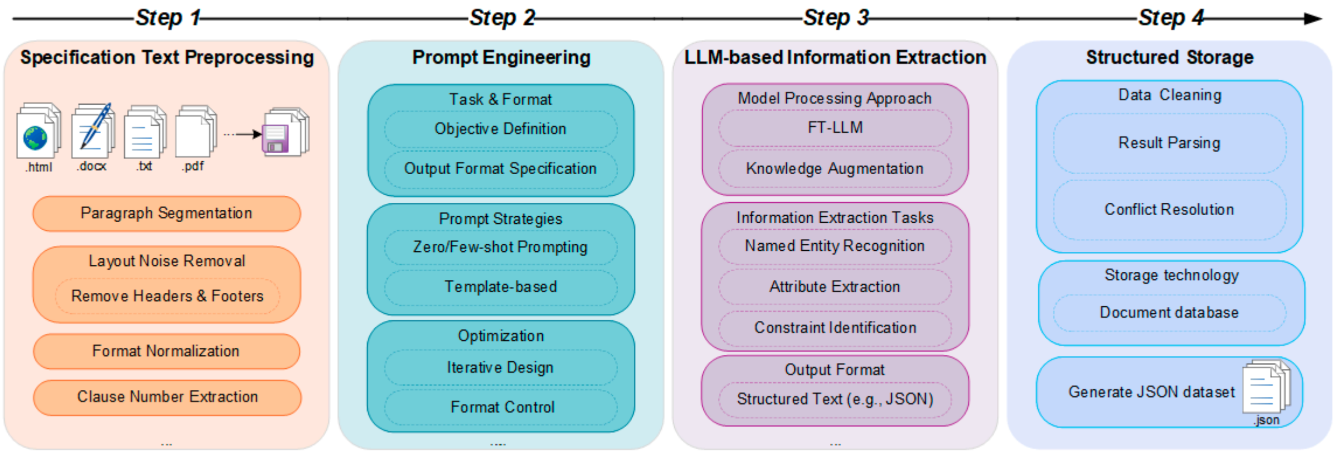


**Figure 3.** LLM-based framework for specification data structuring.

### 2.2.1. Text Preprocessing

Text preprocessing is the first step in converting engineering specifications into structured information that can be processed by LLMs. The specific process involves the following key steps:

1. Format Conversion: Convert engineering specification texts from various sources (e.g., PDF, Word, HTML) into a standardized text format to ensure data consistency and operability.
2. Removal of Irrelevant Content: Eliminate unrelated parts, such as covers, tables of contents, and copyright statements, retaining only the clauses and main body of the text.
3. Clause Segmentation: Automatically divide the text into independent clauses based on the clause numbers in the specification, ensuring clear boundaries for each clause and preventing information confusion.
4. Separation of Comments and Main Text: Separate clause comments from the main body of the specification. Comments are often crucial for understanding the clauses, and separating them facilitates further processing.
5. Text Standardization: Standardize various expressions, units, and symbols to ensure consistency in the text, making subsequent information extraction easier.

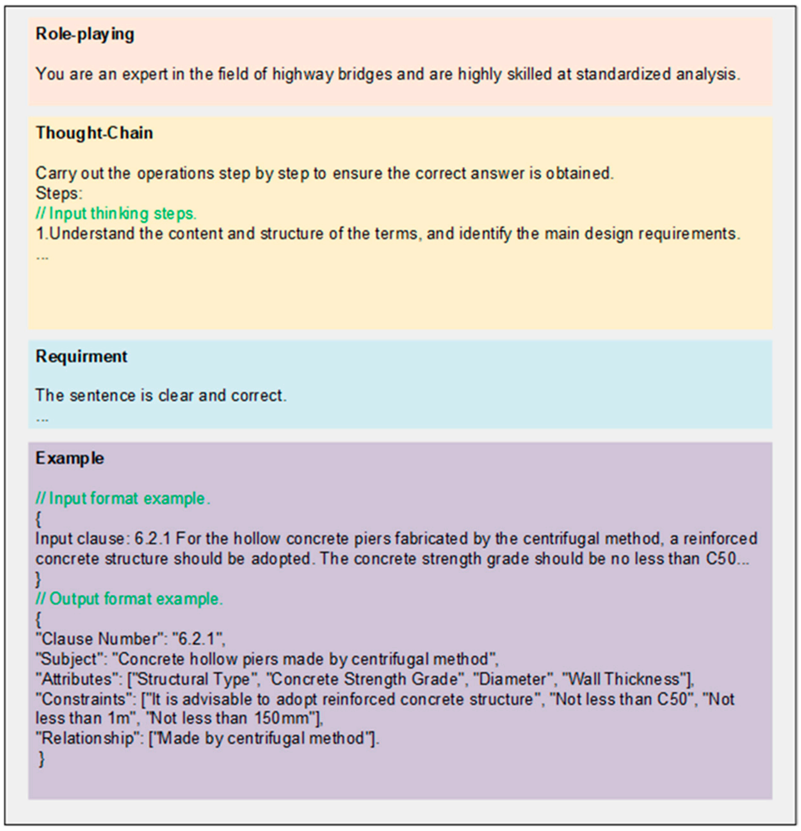
### 2.2.2. Prompt Engineering

To ensure that the LLMs can accurately extract structured information from the specifications, this study employs a structured prompt design method called "Role-playing—Thought-Chain—Requirements—Example" [19,20], as illustrated in Figure 4. The prompt is further enhanced through iterative optimization to improve its effectiveness.

1. Role-playing: The prompt design involves role-playing, where the LLM's role is that of a "highway bridge engineering expert" proficient in the use and analysis of specifications. The task is to extract structured data from the bridge design specifications, particularly information related to design parameters and constraints.
2. Thought Chain: The thought chain technique enables the LLM to better generate the required results. When processing each clause step by step, the model should follow this thought chain:

- Understand the content and structure of the clause and identify the main design requirements.
- Extract key attributes (such as dimensions, strength, etc.), constraints (such as size limits, ratio requirements), and scope from the clause.
- Organize the extracted structured information into a processable format (e.g., JSON, tables, etc.).

3. Requirements: Detailed requirements ensure more accurate content.
   - Accuracy: The extracted information must strictly conform to the specification requirements, avoiding any misunderstandings or omissions.
   - Consistency: Information from different clauses should follow the same structure to ensure the model can uniformly process all clauses.
   - Clarity: Each extracted attribute and constraint should have a clear definition and boundary.
4. Example: Provide input–output examples to help the LLM better generate the desired content. Giving templates for the LLM to learn from is a common and effective method for adjusting large models.



**Role-playing**

You are an expert in the field of highway bridges and are highly skilled at standardized analysis.

**Thought-Chain**

Carry out the operations step by step to ensure the correct answer is obtained.
Steps:
// Input thinking steps.
1.Understand the content and structure of the terms, and identify the main design requirements.
...

**Requirment**

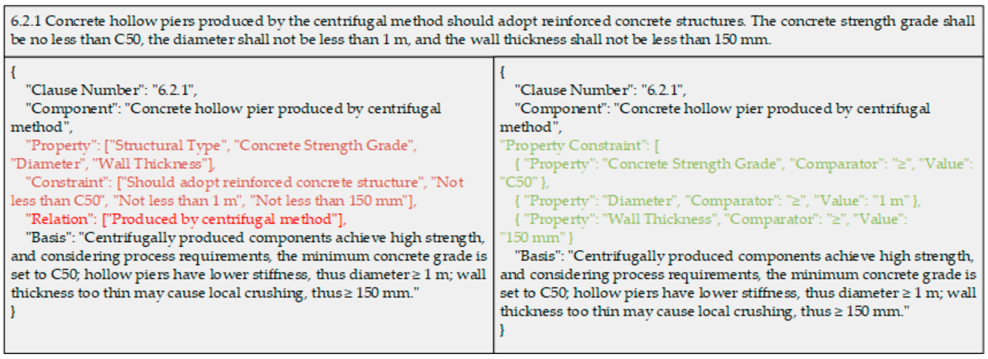
The sentence is clear and correct.
...

**Example**

```
// Input format example.
{
Input clause: 6.2.1 For the hollow concrete piers fabricated by the centrifugal method, a reinforced concrete structure should be adopted. The concrete strength grade should be no less than C50...
}
// Output format example.
{
"Clause Number": "6.2.1",
"Subject": "Concrete hollow piers made by centrifugal method",
"Attributes": ["Structural Type", "Concrete Strength Grade", "Diameter", "Wall Thickness"],
"Constraints": ["It is advisable to adopt reinforced concrete structure", "Not less than C50", "Not less than 1m", "Not less than 150mm"],
"Relationship": ["Made by centrifugal method"].
}
```

**Figure 4.** Structured design of prompts.

Additionally, during prompt design, initial prompts may not achieve the desired results. To enhance the accuracy and consistency of structured information extraction by LLMs, we introduced an iterative optimization process consisting of five iterations. In each iteration, the role was adjusted, for example, from a specification writing expert to a highway bridge engineering expert. The thought chain and requirements were progressively refined to be more specific and precise. Most importantly, examples were continuously improved, as illustrated in Figure 5, by splitting and categorizing information to better structure the extracted data.



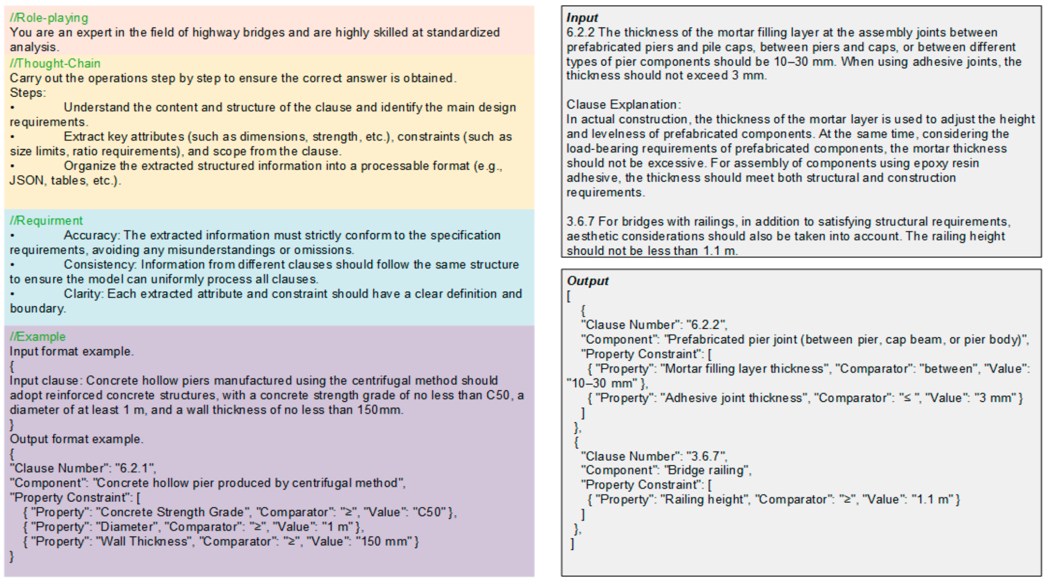
6.2.1 Concrete hollow piers produced by the centrifugal method should adopt reinforced concrete structures. The concrete strength grade shall be no less than C50, the diameter shall not be less than 1 m, and the wall thickness shall not be less than 150 mm.

```
{
   "Clause Number": "6.2.1",
   "Component": "Concrete hollow pier produced by centrifugal
method",
   "Property": ["Structural Type", "Concrete Strength Grade",
"Diameter", "Wall Thickness"],
   "Constraint": ["Should adopt reinforced concrete structure", "Not
less than C50", "Not less than 1 m", "Not less than 150 mm"],
   "Relation": ["Produced by centrifugal method"],
   "Basis": "Centrifugally produced components achieve high strength,
and considering process requirements, the minimum concrete grade is
set to C50; hollow piers have lower stiffness, thus diameter ≥ 1 m; wall
thickness too thin may cause local crushing, thus ≥ 150 mm."
}
```

```
{
   "Clause Number": "6.2.1",
   "Component": "Concrete hollow pier produced by centrifugal
method",
"Property Constraint": [
   { "Property": "Concrete Strength Grade", "Comparator": "≥", "Value":
"C50" },
   { "Property": "Diameter", "Comparator": "≥", "Value": "1 m" },
   { "Property": "Wall Thickness", "Comparator": "≥", "Value":
"150 mm" }
   "Basis": "Centrifugally produced components achieve high strength,
and considering process requirements, the minimum concrete grade is
set to C50; hollow piers have lower stiffness, thus diameter ≥ 1 m; wall
thickness too thin may cause local crushing, thus ≥ 150 mm."
}
```

**Figure 5.** Example refinement process in iterative prompt optimization; by modifying the structure of 'property' and 'constraint', the generated data becomes more structured, facilitating subsequent operations. The 'relation' field was removed to reduce computational load.

To clearly illustrate the structured prompt design for rule extraction, we provide a detailed example in Figure 6.



//Role-playing
You are an expert in the field of highway bridges and are highly skilled at standardized analysis.
//Thought-Chain
Carry out the operations step by step to ensure the correct answer is obtained.
Steps:
- Understand the content and structure of the clause and identify the main design requirements.
- Extract key attributes (such as dimensions, strength, etc.), constraints (such as size limits, ratio requirements), and scope from the clause.
- Organize the extracted structured information into a processable format (e.g., JSON, tables, etc.).

//Requirment
- Accuracy: The extracted information must strictly conform to the specification requirements, avoiding any misunderstandings or omissions.
- Consistency: Information from different clauses should follow the same structure to ensure the model can uniformly process all clauses.
- Clarity: Each extracted attribute and constraint should have a clear definition and boundary.

//Example
Input format example.
{
Input clause: Concrete hollow piers manufactured using the centrifugal method should adopt reinforced concrete structures, with a concrete strength grade of no less than C50, a diameter of at least 1 m, and a wall thickness of no less than 150mm.
}
Output format example.
```
{
"Clause Number": "6.2.1",
"Component": "Concrete hollow pier produced by centrifugal method",
"Property Constraint": [
   { "Property": "Concrete Strength Grade", "Comparator": "≥", "Value": "C50" },
   { "Property": "Diameter", "Comparator": "≥", "Value": "1 m" },
   { "Property": "Wall Thickness", "Comparator": "≥", "Value": "150 mm" }
}
```

***Input***
6.2.2 The thickness of the mortar filling layer at the assembly joints between prefabricated piers and pile caps, between piers and caps, or between different types of pier components should be 10–30 mm. When using adhesive joints, the thickness should not exceed 3 mm.

Clause Explanation:
In actual construction, the thickness of the mortar layer is used to adjust the height and levelness of prefabricated components. At the same time, considering the load-bearing requirements of prefabricated components, the mortar thickness should not be excessive. For assembly of components using epoxy resin adhesive, the thickness should meet both structural and construction requirements.

3.6.7 For bridges with railings, in addition to satisfying structural requirements, aesthetic considerations should also be taken into account. The railing height should not be less than 1.1 m.

***Output***
```
[
   {
   "Clause Number": "6.2.2",
   "Component": "Prefabricated pier joint (between pier, cap beam, or pier body)",
   "Property Constraint": [
    { "Property": "Mortar filling layer thickness", "Comparator": "between", "Value":
"10–30 mm" },
    { "Property": "Adhesive joint thickness", "Comparator": "≤ ", "Value": "3 mm" }
   ]
  },
  {
   "Clause Number": "3.6.7",
   "Component": "Bridge railing",
   "Property Constraint": [
    { "Property": "Railing height", "Comparator": "≥", "Value": "1.1 m" }
   ]
  },
 ]
```

**Figure 6.** Specific application of the prompt: input prompt; input specification text; output JSON data.

### 2.2.3. Knowledge Extraction and Structured Storage Based on Large Language Models

For the preprocessed specification text, this study uses ChatGPT-4.1 (temperature = 0, top-p = 0.9, frequency penalty = 0, presence penalty = 0) and DeepSeek-V3 (temperature = 0, top-p = 0.9, frequency penalty = 0, presence penalty = 0) to extract structured knowledge using the designed prompt templates. Max_token was set to 1.5 $\times$ input token count, with long specifications (>20,000 tokens) divided into segments and processed independently. The output is organized according to the format provided in the prompt examples. By comparing the outputs of two large language models, it becomes easier to identify both erroneous rules and omitted rules.

By extracting a subset of rule information from multiple highway bridge design specifications, a total of 967 JSON-formatted data entries were compiled within a single working day. This approach significantly reduced the time required compared with manual rule encoding. Subsequent manual verification identified 198 irrelevant or incorrect entries, yielding an accuracy rate of 79.5%. Most of the inaccuracies were attributable to semantic misinterpretation and numerical inconsistencies.

To avoid excessive edge generation in the knowledge graph due to an overabundance of nodes and to enhance query efficiency, a JSON-based data schema is employed to optimize the storage structure. This schema not only improves data processing readability but also boosts subsequent query performance. Table 1 illustrates the structure of this JSON schema.

**Table 1.** JSON schema for structured knowledge representation extracted from design specifications.

| Field Name | Meaning | Purpose |
|---|---|---|
| Component | Name of the structural component or entity | Used to generate a component node |
| Property Constraint | | |
| – Property | The design attribute is being constrained | Attached to the relationship as the subject of the constraint |
| – Comparator | Comparison operator (e.g., $\geq$, =, $\leq$) | Defines the logical condition in the relationship |
| – Value | The actual limit or required value | Used to generate a Value node |

The key information extracted by the LLM is then aggregated into a structured JSON file, which serves as input for the next phase of knowledge graph construction.

## 3. Knowledge Graph Construction Methods

### *3.1. Knowledge Graph Technology*

A knowledge graph is a semantic representation technique that models entities and their interrelationships through a graph structure. Its primary purpose is to achieve both the structural organization and computational usability of knowledge, enabling semantic reasoning and querying. Since the introduction of the Google Knowledge Graph in 2012 [21], this approach has become a foundational infrastructure in fields such as artificial intelligence, natural language processing, and complex systems modeling.

Structurally, a knowledge graph is formalized as a directed graph, consisting of nodes and edges: nodes represent entities or concepts, and edges denote semantic relationships between them. Common representation models include the Resource Description Framework (RDF) and the Labeled Property Graph (LPG), each offering different levels of granularity and data management strategies [22].

The fundamental unit of a knowledge graph is the triple, expressed as (subject, predicate, object), which captures the attribute relationships or semantic links between entities. Due to the directional nature of entity relations, a knowledge graph is inherently a directed graph. As illustrated in Figure 7, a head entity points to a tail entity through a specified relationship, forming a semantically constrained network of knowledge representations [23].

Graph databases store nodes and relationships as fundamental modeling units, making them particularly suitable for representing complex structures involving multiple hierarchical levels and diverse relationship types. Unlike traditional relational databases, which use foreign keys to maintain table connections, graph databases eliminate the need for predefined join paths. Instead, they store associations directly between entities, significantly enhancing semantic connectivity and query efficiency. Neo4j is a widely used graph database that employs a specialized query language called Cypher.

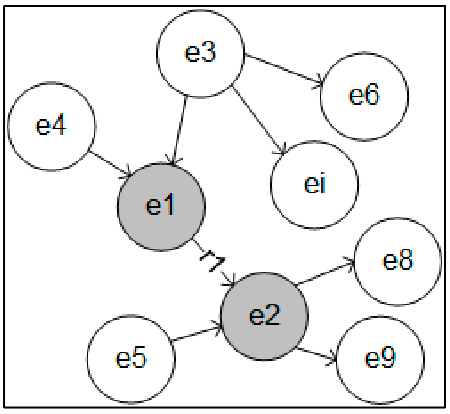


**Figure 7.** An example of a knowledge graph. In this graph, (e1, r1, e2) is a triplet that indicates that e1 and e2 are connected by relation r1 [23].

### 3.2. Knowledge Graph Construction Workflow

Building upon the prior discussion, knowledge graph technology not only facilitates intelligent querying and solution recommendation but also enables semantic rule checking by integrating domain codes with component-level semantic information. This functionality is crucial for the automated checking of BIM models. As illustrated in Figure 8, this study proposes a technical workflow for knowledge graph construction using structured specification data as input. The overall process consists of three stages: (1) data preparation, (2) ontology modeling, (3) graph construction.

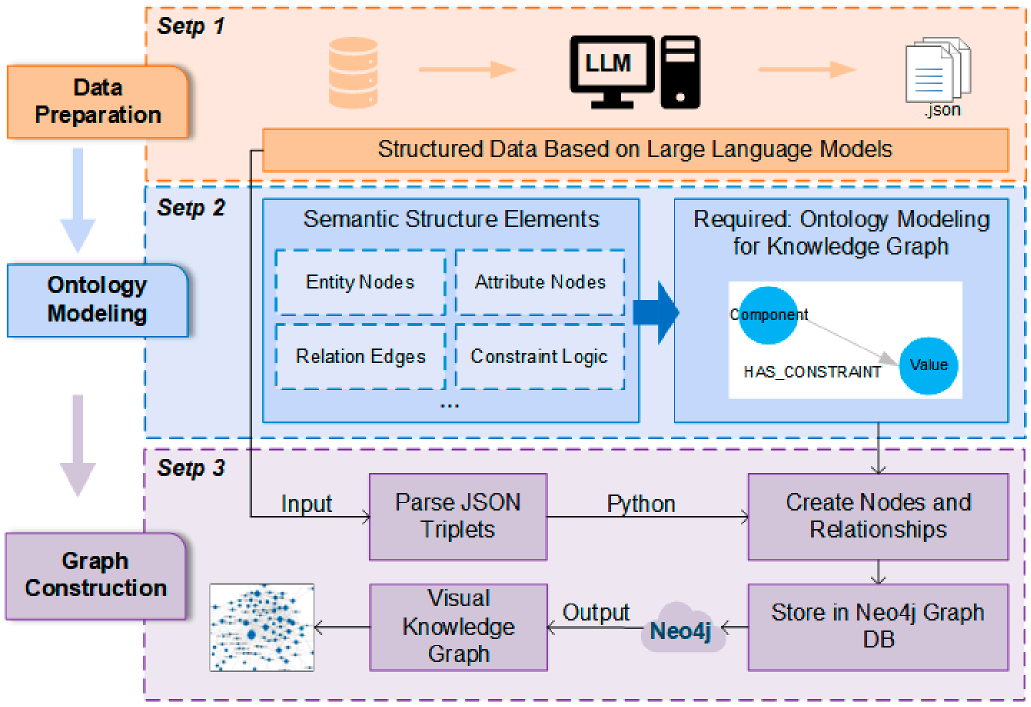


**Figure 8.** Technical workflow for knowledge graph construction.

## 4. Automated Checking of Highway Bridge BIM Models Based on Large Language Models

### 4.1. Research Methods and Technical Framework

Based on the previous analysis of the application of natural language processing, LLMs, and KGs in highway bridge design, this paper proposes an automated checking method for highway bridge BIM models based on large language models. The method leverages the strengths of LLMs in semantic understanding, rule extraction, and knowledge generation, converting large amounts of unstructured text from bridge design specifications into structured rules that can be processed by machines. A queryable knowledge graph system is constructed using the Neo4j graph database, supporting subsequent model checking.

To achieve information matching between design specifications and engineering models, this study selects IFC as the data format for highway bridge BIM models. IFC is an open and semantically rich BIM standard with a comprehensive entity system, covering components such as IfcColumn, IfcBeam, and IfcSlab, as illustrated in Figure 9. This format is well-suited for integration with the knowledge graph and effectively supports automated model checking.

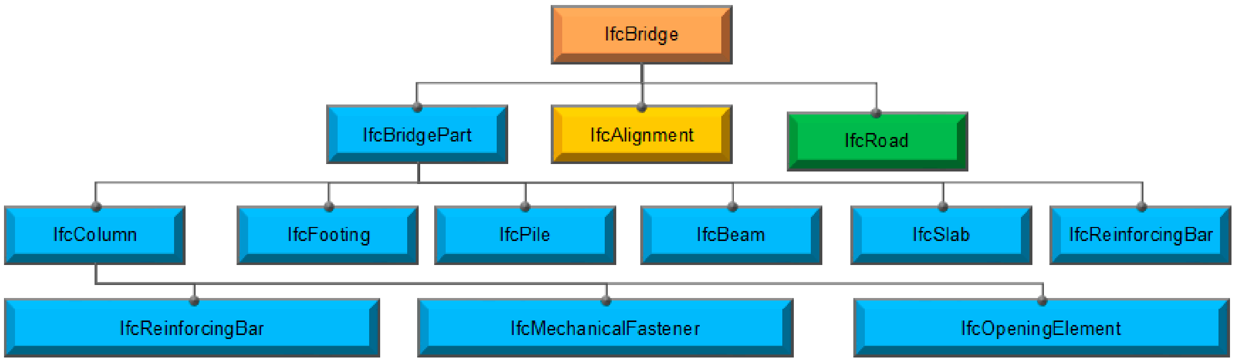


**Figure 9.** Partial examples of IfcBridge entities.

The overall technical framework, as illustrated in Figure 10, includes three core processes.

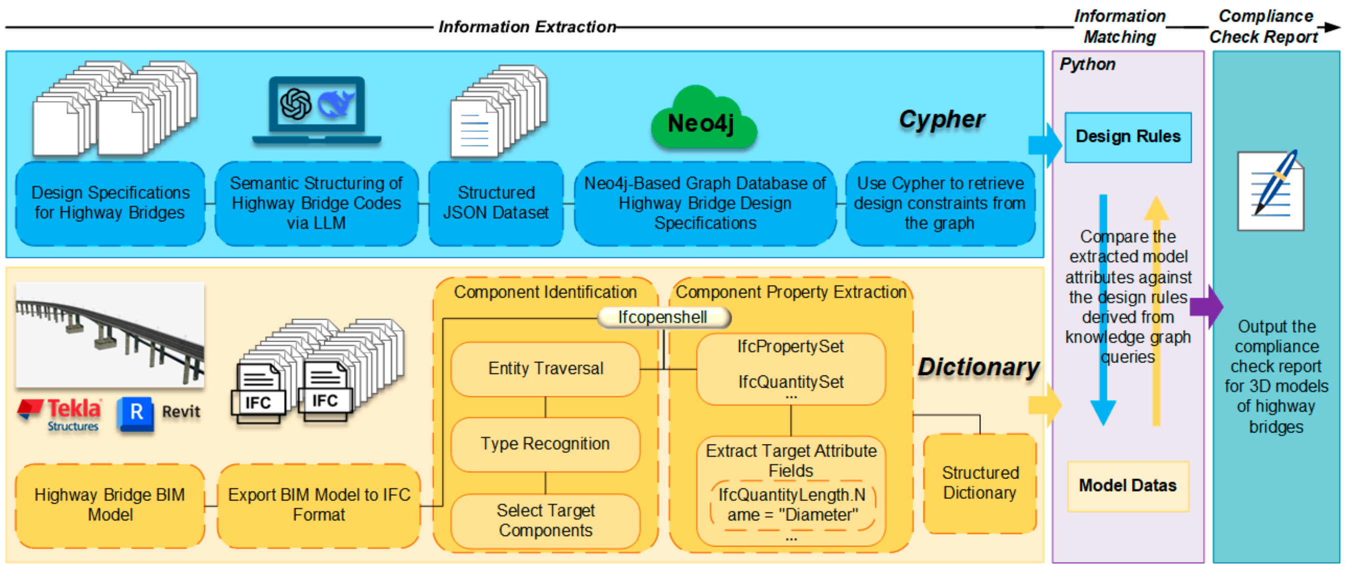

Information Extraction
Information Matching
Compliance Check Report
Python
Neo4j
Cypher
Design Specifications for Highway Bridges
Semantic Structuring of Highway Bridge Codes via LLM
Structured JSON Dataset
Neo4j-Based Graph Database of Highway Bridge Design Specifications
Use Cypher to retrieve design constraints from the graph
Design Rules
Compare the extracted model attributes against the design rules derived from knowledge graph queries
Model Datas
Output the compliance check report for 3D models of highway bridges
Tekla Structures
Revit
IFC
IFC
Highway Bridge BIM Model
Export BIM Model to IFC Format
Component Identification
Ifcopenshell
Entity Traversal
Type Recognition
Select Target Components
Component Property Extraction
IfcPropertySet
IfcQuantitySet
...
Extract Target Attribute Fields
IfcQuantityLength.Name = "Diameter"
...
Dictionary
Structured Dictionary

**Figure 10.** Technical workflow for automated checking of highway bridge BIM models based on LLMs.

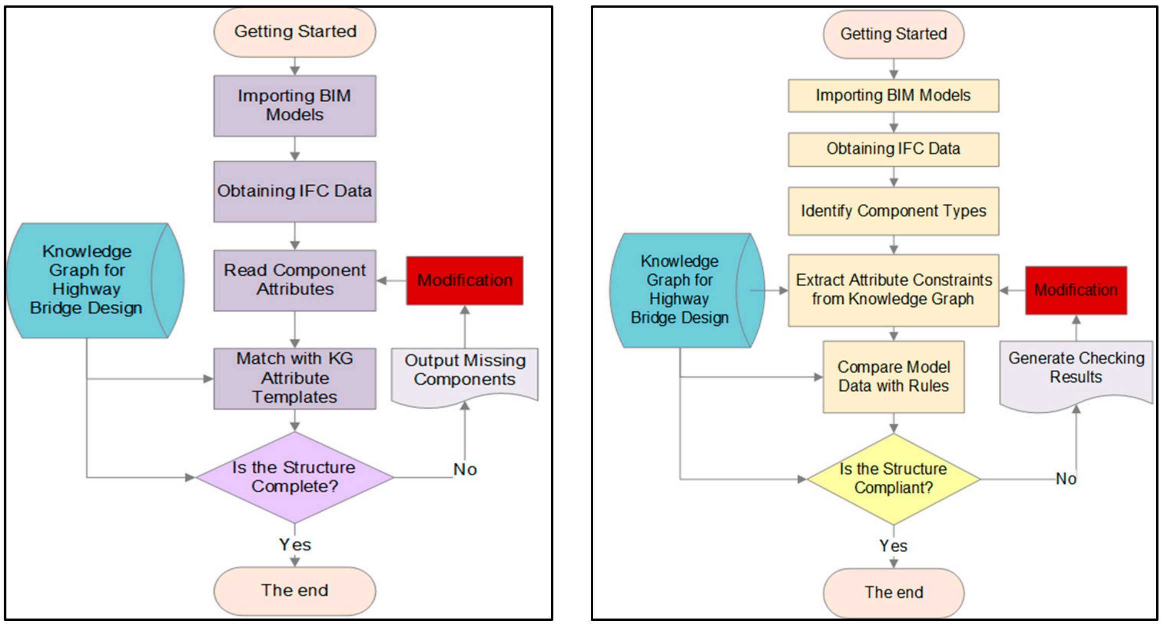
1. Information Extraction: Extract structured rules from bridge design specifications and construct the knowledge graph, while extracting target components and their attribute data from the BIM model.
2. Information Matching: Match the actual model data with the design requirements in the graph database.
3. Inspection Result Generation: Output the inspection results and provide feedback and design adjustment recommendations as necessary.

### *4.2. Scope of Model Checking*

Within the framework proposed in this study, the automated checking of highway bridge BIM models focuses on two main aspects: structural completeness checking and regulatory compliance checking.

#### 4.2.1. Structural Completeness Checking

Structural completeness checking is used to verify whether all necessary structural components are present in the model. For example, a bridge pier must include a pier body, pier cap, and foundation element. As illustrated in Figure 11a, the system first reads the list of properties associated with each physical component and checks for their presence. It then retrieves the relevant attribute templates from the knowledge graph and performs matching and verification. This method effectively identifies missing components and helps to detect issues such as incomplete structural hierarchies or missing model elements.


Getting Started
Importing BIM Models
Obtaining IFC Data
Knowledge Graph for Highway Bridge Design
Read Component Attributes
Modification
Match with KG Attribute Templates
Output Missing Components
Is the Structure Complete?
No
Yes
The end
Getting Started
Importing BIM Models
Obtaining IFC Data
Identify Component Types
Knowledge Graph for Highway Bridge Design
Extract Attribute Constraints from Knowledge Graph
Modification
Compare Model Data with Rules
Generate Checking Results
Is the Structure Compliant?
No
Yes
The end

(**a**) (**b**)

**Figure 11.** Model checking workflow diagrams: (**a**) structural completeness checking; (**b**) regulatory compliance checking.

#### 4.2.2. Regulatory Compliance Checking

Regulatory compliance checking is used to verify whether the attributes of structural components conform to the relevant design standards. The checking process consists of two levels:

1. Single-Component Attribute Compliance: Determines whether the attribute values of each component (e.g., diameter, height, or concrete strength grade) comply with the specifications defined in the knowledge graph.
2. Inter-Component Relational Compliance: Evaluates whether the logical relationships between multiple components (e.g., the spacing between adjacent piers or the elevation difference between the abutment and road surface) satisfy the design requirements.

As illustrated in Figure 11b, this process begins by parsing the IFC model to identify various bridge component types. The system then extracts the relevant attribute constraint rules from the knowledge graph and compares them with the extracted model data. Based on the comparison results, the system outputs inspection feedback and provides design adjustment recommendations when necessary.

## 5. Case Study and Model Validation

### *5.1. LLM-Based Structuring of Highway Bridge Specifications*

To validate the effectiveness of the proposed method, this study selected representative clauses from multiple highway bridge design specifications and processed them using the LLM-based framework. Through text preprocessing, structured prompt design, and knowledge extraction, these unstructured clauses were successfully transformed into a structured JSON dataset.

### *5.2. Knowledge Graph Database Construction*

In this study, Neo4j is selected as the platform for visual representation of the knowledge graph related to highway bridge BIM models. The structured JSON dataset is imported into the Neo4j graph database using the py2neo library in a Python 3.10 environment. By establishing a connection between Python and the Neo4j database, the previously extracted structured data is parsed and stored, enabling the creation of a knowledge graph that includes entities, their relationships, and associated attribute constraints.

The design of the knowledge graph model revolves around three core components: subject, attribute, and constraint. The corresponding data structure is summarized in Table 2. Each attribute constraint results in the creation of a value node (which can be reused) and a HAS_CONSTRAINT relationship from the component node to the value node.

**Table 2.** Node and relationship design in the knowledge graph.

| Type | Description | Unique Attribute |
|---|---|---|
| Component | Node representing a structural component entity | Name |
| Value | Node representing a constraint value | Value |
| Relation: HAS_CONSTRAINT | Edge from component to value, indicating that the value is a constraint on a certain attribute | Carries two properties: attribute, comparator |

Based on the previously extracted structured JSON data, the knowledge graph for the highway bridge 3D model is constructed. The key steps involved in the implementation process are briefly introduced below:

1. Import the necessary libraries, such as py2neo and JSON.
2. Connect to the local Neo4j database and configure the connection parameters.
3. Load and parse the JSON file, which includes components and their attribute constraints.
4. Iterate through the data to create component nodes, value nodes, and HAS_CONSTRAINT relationships.

Through this process, a complete knowledge graph containing various components of the highway bridge, their relationships, and associated attribute constraints can be generated. Figure 12 shows an example of a constructed knowledge graph.

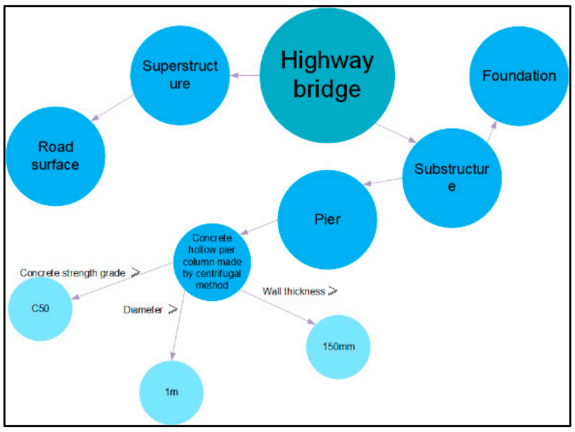


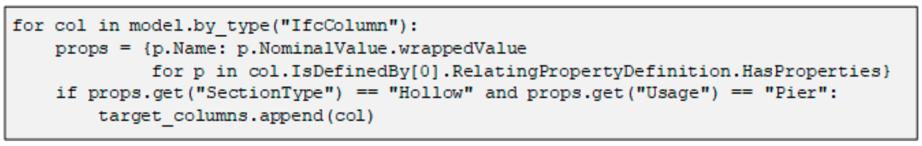
**Figure 12.** Partial knowledge graph of highway bridge BIM models.

### *5.3. Model Checking Based on the Graph Database*

#### 5.3.1. Component Localization and Attribute Extraction Based on the IFC Model

As an example, the BIM model of a highway bridge alignment is selected for demonstration. The open-source library, Ifcopenshell, is utilized to parse IFC model data and extract structured component information in two main stages:

- Component Localization: The process begins by identifying pier-related components in the IFC model using the IfcColumn type. In the IFC standard, IfcColumn represents vertical load-bearing components, suitable for modeling solid or hollow pier columns. Each the IfcColumn entity is further filtered based on key properties defined in its IfcPropertySet. This ensures accurate identification of hollow pier columns even when naming conventions vary. The updated component localization logic is illustrated in Figure 13.



```
for col in model.by_type("IfcColumn"):
    props = {p.Name: p.NominalValue.wrappedValue
             for p in col.IsDefinedBy[0].RelatingPropertyDefinition.HasProperties}
    if props.get("SectionType") == "Hollow" and props.get("Usage") == "Pier":
        target_columns.append(col)
```

**Figure 13.** Python—identification of hollow pier columns based on IFC classification and property sets.

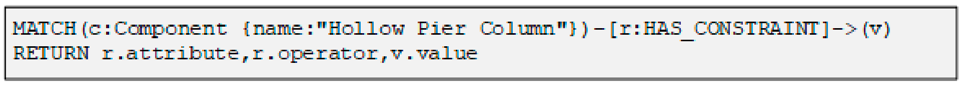
- Attribute Extraction: For each identified component, key structural and material properties are extracted. In IFC, component attributes are typically stored in IfcPropertySet entities, which are linked to components via IfcRelDefinesByProperties. This study focuses on attributes such as diameter, concrete strength grade, and wall thickness. The extracted results for each component are ultimately organized into a structured dictionary format as illustrated in Table 3.

**Table 3.** Structured attribute dictionary of a hollow pier column.

| Field | Value |
|---|---|
| Component Name | Hollow Pier Column-1 |
| Attributes | |
| – Diameter | 0.9 m |
| – Concrete Strength Grade | C50 |
| – Wall Thickness | 150 mm |

### 5.3.2. Validation of Model Attributes Against Knowledge Graph Rules

After extracting component attributes, the system compares the results with the corresponding design specifications stored in the knowledge graph for rule-based validation. Figure 14 shows the Cypher query used to retrieve constraint rules from the graph database.



```
MATCH(c:Component {name:"Hollow Pier Column"})-[r:HAS_CONSTRAINT]->(v)
RETURN r.attribute,r.operator,v.value
```

**Figure 14.** Cypher query—retrieval of constraint rules from the knowledge graph.

After retrieving the model attributes and the corresponding constraint rules from the knowledge graph, the system performs a logical comparison. The comparison function first removes units and converts values into numerical form. Then, it evaluates the compliance based on the comparison operator and the target value defined in the graph. If the actual value satisfies the specified design constraint, it is marked as "compliant"; otherwise, it is labeled as "non-compliant".

For example, Clause 6.2.1 of the Technical Specifications for Design of Prefabricated Concrete Highway Bridges (JTGT 3365-05—2022) [24] states: "For hollow piers fabricated using centrifugal methods, reinforced concrete structures are recommended. The concrete strength grade should not be lower than C50, the diameter should not be less than 1 m, and the wall thickness should not be less than 150 mm." In addition, according to Clause 6.2.1 of the Highway Bridge Foundation and Substructure Design Code (JTG 3363-2019) [25], the spacing between piles should not be less than 2.4 m (calculated as 2 * 1.2 m). Furthermore, for the cap, its thickness must not be less than 1.5 m. For the railing, according to Clause 3.6.7 of the General Specifications for Highway Bridges (JTGD60—2015) [26], the height should be no less than 1.1 m.

The batch inspection of bridge piers in the highway bridge alignment BIM model was conducted, and the results are shown in Table 4.

**Table 4.** Compliance checking results for bridge pier, cap, and railing.

| No. | Component Name | Attribute | Actual Value | Constraint |
|---|---|---|---|---|
| 1. | Hollow Pier Column-1 | Diameter | 0.98 m | ≥1 m |
| 2. | Hollow Pier Column-8 | Diameter | 0.90 m | ≥1 m |
| 3. | Hollow Pier Column-4 | Concrete Strength Grade | C25 | ≥C50 |
| 4. | Hollow Pier Column-7 | Concrete Strength Grade | C30 | ≥C50 |
| 5. | Cap-5 | Cap Thickness | 1.0 m | ≥1.5 m |
| 6. | Cap-10 | Cap Thickness | 1.2 m | ≥1.5 m |
| 7. | Railing-1 | Height | 1.0 m | ≥1.1 m |
| 8. | Hollow Pier | Pile Spacing | 1.8 m | ≥2.4 m |

After performing structural completeness checking on the highway bridge BIM model, we evaluated the performance of rule extraction based on LLMs. The total sample size was 112, with 65 rules correctly identified as compliant (true positive, TP), 8 rules correctly identified as non-compliant (true negative, TN), 12 rules incorrectly marked as compliant (false positive, FP), and 27 rules of compliant rules not identified (false negative, FN). Based on these data, four key performance metrics were calculated, as shown in Table 5.

**Table 5.** Performance metrics of structural completeness checking.

| Metric | Formula | Value |
|---|---|---|
| Accuracy | (TP + TN)/(TP + TN + FP + FN) | 65.20% |
| Precision | TP/(TP + FP) | 84.40% |
| Recall | TP/(TP + FN) | 70.70% |
| F1 Score | 2 * (Precision * Recall)/(Precision + Recall) | 76.90% |

Overall, the method demonstrates good performance in terms of precision and F1 score, while the recall is relatively lower, indicating room for improvement in capturing all compliant rules.

In addition, a structural integrity check is performed on the model attributes extracted from the knowledge graph, as shown in Table 6.

**Table 6.** Structural completeness check results of highway bridge BIM models.

| Parameter Name | IFC Type | Check Result |
|---|---|---|
| Deck | IfcSlab | Correct |
| Bridge Beams | IfcBeam | Correct |
| Bridge Slab | IfcSlab | Correct |
| Pier | IfcColumn | Correct |
| Bearing | IfcBearing | Correct |
| Abutments | IfcBridgeElement | Correct |
| Foundation | IfcFoundation | Correct |
| Cap Slab | IfcSlab | Correct |
| Piles | IfcPile | Correct |
| Transition Section | IfcBridgeElement | Absent |
| Expansion Joints | IfcBridgeElement | Absent |

## 6. Discussion

### *6.1. Innovation and Effectiveness of the Method*

This study proposes an automated checking framework for highway bridge BIM models based on large language models (LLMs). The framework is innovative in its approach to semantic parsing and structuring of design specifications. The structured data extracted by LLMs is stored in a knowledge graph and compared with BIM model data using Cypher queries in the Neo4j graph database. This method achieves an efficient and scalable automated checking process. Compared to traditional manual methods, this framework reduces reliance on human experts, increases checking efficiency, and enhances the automation and accuracy of compliance checks for highway bridge BIM models.

### *6.2. Practical Significance*

As BIM technology becomes more widely adopted in highway bridge design, the complexity of design models and specifications increases and traditional manual checking methods can no longer meet current demands. The proposed framework leverages LLMs for efficient semantic parsing and knowledge graphs for storing rules. It accelerates the parsing and comparison of design specifications, significantly improving the efficiency and accuracy of the checking process. This framework has broad application potential, not only in highway bridge design but also in other infrastructure fields such as tunnels, roads, and buildings, driving the digital transformation of engineering projects.

### *6.3. System Optimization, Limitations, and Future Work*

Although the framework has shown strong performance, several limitations remain. Future research can focus on the following areas:

- Accuracy of Semantic Parsing by LLMs: The accuracy of LLMs in parsing design specifications depends heavily on prompt formulation. Performance can vary across engineering domains and specification formats. Future work should focus on refining prompt design and expanding training data to improve the model's understanding of complex design rules. In practice, LLMs may produce outputs that do not match the specifications or generate inconsistent numerical values. To address this, we decomposed the specifications into individual clauses, clarified the reasoning steps and requirements, and provided structured examples. Prompts and outputs were iteratively refined to improve accuracy and consistency. The choice of LLM and parameter settings also affects the results. For example, in specification information extraction, ChatGPT-4.1 showed slightly higher speed and accuracy than DeepSeek-V3, but it produced more hallucinations.
- Domain-Specific Fine-Tuning: To further improve LLM performance in parsing bridge engineering specifications, future work could explore domain-specific fine-tuning. This involves retraining the model on bridge design specifications and historical BIM datasets. Such fine-tuning helps the model to better understand engineering terminology, numerical constraints, and structural rules, reducing errors in semantic parsing and improving reliability. For example, Kasimir Forth [27] proposed a method that uses semantic textual similarity (STS) and fine-tuned multilingual LLMs to automatically enrich missing information in BIM. Their study demonstrates that domain-specific fine-tuning can significantly enhance model performance in engineering BIM tasks, providing a solid reference for bridge specification parsing. In addition, recent AI applications in infrastructure engineering include deep learning for structural health monitoring, reinforcement learning for construction process optimization, and generative AI for design proposal generation [28]. These approaches illustrate the increasing integration of AI technologies into digital construction workflows. Although this study focuses on LLM-based automated checking, these examples provide context for potential future extensions.
- Knowledge Graph Storage and Query Efficiency: The knowledge graph effectively stores design specification rules but may face challenges in handling large-scale, cross-disciplinary specifications. Future research should address improving the scalability and query performance of knowledge graphs in multi-disciplinary applications.
- BIM Model Data Quality: BIM model data in real-world applications often suffer from incompleteness or inconsistency, requiring additional preprocessing. Future work should explore methods to enhance the robustness of the system by optimizing BIM model data preprocessing.
- Integration with BIM Workflows and Computational Considerations: The framework extracts IFC data using Python and the Ifcopenshell library. Key components, including piers, caps, beams, and slabs, are identified, and attributes such as diameter, concrete strength, wall thickness, and pile spacing are retrieved. These attributes are compared with design rules stored in the knowledge graph, and checking results are presented in tables and visual alerts. Users can import IFC data for batch checking and obtain structured outputs. Current interaction features are basic and not fully integrated into BIM workflows. Regarding computational cost, the current implementation is lightweight and can process medium-sized bridge models on standard workstations. Future work will focus on optimizing performance and implementing parallel processing to handle larger models efficiently. Additionally, BIM plugins or standalone applications will be developed to allow engineers and inspectors to access automated checking directly within common BIM environments, improving usability, streamlining workflows, and promoting practical adoption.

## 7. Conclusions

This study proposes an automated checking framework for highway bridge BIM models based on large language models. By converting design specifications into structured rules and storing them in a knowledge graph, the framework provides a practical and promising approach for automated rule extraction, with the potential to substantially improve the efficiency and accuracy of compliance checking in practical engineering applications.

**Author Contributions:** Conceptualization, Y.Y.; X.J. and Y.-M.L.; Software, Y.Y. and X.J.; Data Curation, X.J.; Writing—Original Draft Preparation, X.J.; Writing—Review and Editing, Y.Y. and Y.-M.L.; Funding Acquisition, Y.Y. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research was supported by the National Natural Science Foundation of China (NSFC), Grant No. 52378537; the Yunnan Province Major Science and Technology Special Plan Project, Grant No. 202302AD080009; the Yunnan Provincial Science and Technology Plan Project, Grant No. 202305AF150138; and the Yunnan Provincial Science and Technology Special Plan, Grant No. 202102AD080003.

**Data Availability Statement:** The data generated during the study are not publicly available due to privacy or ethical restrictions. Access to the data can be requested from the corresponding author.

**Acknowledgments:** The authors would like to thank the three anonymous reviewers for their valuable comments and suggestions, which have greatly contributed to the improvement of this manuscript.

**Conflicts of Interest:** The authors declare no conflicts of interest.

## Abbreviations

The following abbreviations are used in this manuscript:

| | |
|---|---|
| BIM | Building Information Modeling |
| LLMs | Large Language Models |
| KGs | Knowledge Graphs |
| IFC | Industry Foundation Classes |
| NLP | Natural Language Processing |
| RDF | Resource Description Framework |
| LPG | Labeled Property Graph |
| SMT | Statistical Machine Translation |
| LLaMA-2 | Large Language Model Meta AI |
| SVM | Support Vector Machines |
| HMM | Hidden Markov Models |
| KNN | K-Nearest Neighbors |
| STS | Semantic Textual Similarity |

## References

1. Zhou, X.; Zhao, J.C.; Wang, J.; Huang, X.; Li, X.; Guo, M.; Xie, P. Parallel Computing-Based Online Geometry Triangulation for Building Information Modeling Utilizing Big Data. *Autom. Constr.* **2019**, *107*, 102942. [CrossRef]
2. Eastman, C.; Lee, J.M.; Jeong, Y.S.; Lee, J.-K. Automatic Rule-Based Checking of Building Designs. *Autom. Constr.* **2009**, *18*, 1011–1033. [CrossRef]
3. Zhou, P.; El Gohary, N. Ontology-Based Automated Information Extraction from Building Energy Conservation Codes. *Autom. Constr.* **2017**, *74*, 103–117. [CrossRef]
4. Wang, X.; El-Gohary, N. Deep Learning-Based Relation Extraction from Construction Safety Regulations for Automated Field Compliance Checking. In Proceedings of the Construction Research Congress 2022, Arlington, TX, USA, 9–12 March 2022; pp. 290–297. [CrossRef]
5. Zhang, J.S.; El Gohary, N.M. Integrating Semantic NLP and Logic Reasoning into a Unified System for Fully-Automated Code Checking. *Autom. Constr.* **2017**, *73*, 45–57. [CrossRef]

6. Xue, X.R.; Zhang, J.S. Part-of-Speech Tagging of Building Codes Empowered by Deep Learning and Transformational Rules. *Adv. Eng. Inform.* **2021**, *47*, 101235. [CrossRef]
7. Lee, J.; Yi, J.-S. Predicting Project's Uncertainty Risk in the Bidding Process by Integrating Unstructured Text Data and Structured Numerical Data Using Text Mining. *Appl. Sci.* **2017**, *7*, 1141. [CrossRef]
8. Hassan, F.U.; Le, T.; Lv, X. Addressing Legal and Contractual Matters in Construction Using Natural Language Processing: A Critical Review. *J. Constr. Eng. Manag.* **2021**, *147*, 03121004. [CrossRef]
9. Hassan, F.U.; Le, T. Automated Requirements Identification from Construction Contract Documents Using Natural Language Processing. *J. Leg. Aff. Dispute Resolut. Eng. Constr.* **2020**, *12*, 04520009. [CrossRef]
10. Peng, J.; Liu, X. Automated Code Compliance Checking Research Based on BIM and Knowledge Graph. *Sci. Rep.* **2023**, *13*, 7065. [CrossRef] [PubMed]
11. Hearne, M.; Way, A. Statistical Machine Translation: A Guide for Linguists and Translators. *Lang. Linguist. Compass* **2011**, *5*, 205–226. [CrossRef]
12. Koehn, P.; Hoang, H.; Birch, A.; Callison-Burch, C.; Federico, M.; Bertoldi, N.; Cowan, B.; Shen, W.; Moran, C.; Zens, R.; et al. Moses: Open Source Toolkit for Statistical Machine Translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*; Ananiadou, S., Ed.; Association for Computational Linguistics: Prague, Czech Republic, 2007; pp. 177–180.
13. Koseki, S.; Kutsuzawa, K.; Owaki, D.; Hayashibe, M. Multimodal Bipedal Locomotion Generation with Passive Dynamics via Deep Reinforcement Learning. *Front. Neurorobot.* **2022**, *16*, 1054239. [CrossRef] [PubMed]
14. Qi, W.; Fan, H.Y.; Karimi, H.R.; Su, H. An Adaptive Reinforcement Learning-Based Multimodal Data Fusion Framework for Human–Robot Confrontation Gaming. *Neural Netw.* **2023**, *164*, 489–496. [CrossRef] [PubMed]
15. Zhao, W.X.; Zhou, K.; Li, J.; Tang, T.; Wang, X.; Hou, Y.; Min, Y.; Zhang, B.; Zhang, J.; Dong, Z.; et al. A Survey of Large Language Models. *arXiv* **2025**. [CrossRef] [PubMed]
16. GPT-4. Available online: https://openai.com/zh-Hans-CN/index/gpt-4/ (accessed on 14 September 2025).
17. Li, H.; Dong, Z.; Wang, S.; Zhang, H.; Shen, L.; Peng, X.; She, D. Extracting Formal Specifications from Documents Using LLMs for Automated Testing. In Proceedings of the 2025 IEEE/ACM 33rd International Conference on Program Comprehension (ICPC), Ottawa, ON, Canada, 27–28 April 2025.
18. Dagdelen, J.; Dunn, A.; Lee, S.; Walker, N.; Rosen, A.S.; Ceder, G.; Persson, K.A.; Jain, A. Structured Information Extraction from Scientific Text with Large Language Models. *Nat. Commun.* **2024**, *15*, 1418. [CrossRef] [PubMed]
19. Shanahan, M.; McDonell, K.; Reynolds, L. Role Play with Large Language Models. *Nature* **2023**, *623*, 493–498. [CrossRef] [PubMed]
20. Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; Ichter, B.; Xia, F.; Chi, E.; Le, Q.; Zhou, D. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. *arXiv* **2023**. [CrossRef]
21. Ehrlinger, L.; Wöß, W. Towards a Definition of Knowledge Graphs. In Proceedings of the Posters and Demos Track of 12th International Conference on Semantic Systems—SEMANTiCS2016 and 1st International Workshop on Semantic Change & Evolving Semantics (SuCCESS16), Leipzig, Germany, 12–15 September 2016.
22. Färber, M.; Bartscherer, F.; Menne, C.; Rettinger, A. Linked Data Quality of DBpedia, Freebase, OpenCyc, Wikidata, and YAGO. *Semantic Web* **2017**, *9*, 77–129. [CrossRef]
23. Peng, C.; Xia, F.; Naseriparsa, M.; Osborne, F. Knowledge Graphs: Opportunities and Challenges. *Artif. Intell. Rev.* **2023**, *56*, 13071–13102. [CrossRef] [PubMed]
24. *JTGT 3365-05—2022*; Technical Specifications for Design of Prefabricated Concrete Highway Bridges. Ministry of Transport of the People's Republic of China: Beijing, China, 2022.
25. *JTG 3363-2019*; Highway Bridge Foundation and Substructure Design Code. Ministry of Transport of the People's Republic of China: Beijing, China, 2019.
26. *JTGD60—2015*; General Specifications for Highway Bridges. Ministry of Transport of the People's Republic of China: Beijing, China, 2015.
27. Forth, K.; Borrmann, A. Semantic Enrichment for BIM-Based Building Energy Performance Simulations Using Semantic Textual Similarity and Fine-Tuning Multilingual LLM. *J. Build. Eng.* **2024**, *95*, 110312. [CrossRef]
28. Nyokum, T.; Tamut, Y. Artificial Intelligence in Civil Engineering: Emerging Applications and Opportunities. *Front. Built Environ.* **2025**, *11*, 1622873. [CrossRef]