Ontology-based Knowledge Graph for Industrial Standards with Hierarchical and Conditional Structuring

Anonymous ACL submission

Abstract

Industrial standards documents contain complex conditional statements and table-based rules, making them challenging to interpret using conventional text-based approaches. Although large language model–based document understanding methods have been actively studied, they often fail to capture hierarchical and conditional document structures, limiting their ability to model complex conditions and table-driven rules. To address this challenge, we propose an Ontology-based knowledge graph (KG) construction method that integrates ontologies with conditional-structure-based triplet extraction, and evaluate its effectiveness using question answering datasets constructed from multiple industrial standards documents. Experimental results show statistically significant performance improvements over baseline models, with particularly notable gains on reasoning tasks involving table-based rules and multi-condition reasoning, demonstrating that the proposed Ontology-based KG effectively captures document structure for reliable question answering. Code is available at: https://anonymous.4open.science/r/ontology_based_kg_paper-F5CE

1 Introduction

Industrial standards documents are used as authoritative references to ensure the quality of materials and processes in decision-making processes such as design reviews, technical comparisons, and contract drafting across various industries, including shipbuilding, construction, and energy. These documents typically follow a hierarchical structure in the form of SECTION–SUBSECTION–CLAUSE, and they incorporate diverse forms of expression such as tables, multiple conditions, unit constraints, and footnotes. As a result, the meaning of rules is determined within the global hierarchical context of the document rather than in isolation. Due to this structural complexity, the interpretation of industrial standards documents inherently requires domain expertise (Kabzhan et al., 2025). In real industrial settings, experts manually review large volumes of such documents, which entails significant operational costs.

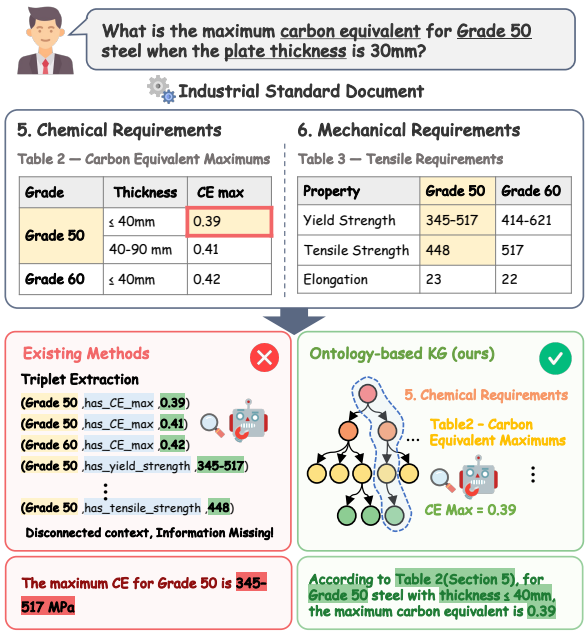

Figure 1: Comparison between Conventional KG and Ontology-based KG. In industrial standards documents, the same term (e.g., Grade 50) may appear in different sections with different meanings. Conventional approaches lose section-level context and conditional information during triplet extraction, which often leads to incorrect answers. In contrast, the proposed Ontology-based KG preserves the document's hierarchical structure and conditions, enabling accurate reasoning.

With the rapid advancement of large language models (LLMs), cost efficiency has been introduced to a wide range of industrial domains, leading to the active adoption of LLMs for industrial standards documents in tasks such as retrieval-augmented generation (RAG)–based question answering (QA) and contract drafting (Boecking et al., 2025). Most existing LLM-based approaches process text at the sentence or chunk level. How-

ever, unlike general-domain text, the document structure itself is a core component of semantic interpretation in industrial standards. Specifically, the same term may denote entirely different rules depending on the section, and certain rules require the combination of conditional statements scattered across the document. Consequently, the meaning of a rule is not determined independently at the sentence level but is defined through the document's hierarchical structure and the composition of conditions (Kabzhan et al., 2025). Because existing methods perform retrieval after removing document structure, hallucinations frequently occur, such as mixing rules from different sections or generating answers with missing conditions (see Figure 1). Moreover, conventional triplet extraction methods are designed to extract `subject-relation-object` independently from individual sentences (Zhang and Soh, 2024a), which makes it difficult to faithfully represent table-based rules that combine row and column conditions, if–then conditional statements, or exception clauses. As a result, the loss of document structural information leads to critical errors in downstream tasks and significantly degrades the reliability of RAG-based systems (Tufek, 2023).

To address these limitations, this study introduces ontology concepts tailored to industrial standards documents in order to represent rules while preserving both the hierarchical structure and the document-level semantics. Specifically, we propose an Ontology-based pipeline consisting of three stages. **(1) Hierarchical Ontology Modeling** — formalizing the document's hierarchical structure as an ontology schema. **(2) Sentence-level Conditional Structure Modeling** — extracting triplets by modeling conditional structures derived from conditional statements and tables. **(3) Ontology-based KG Construction** — performing synonym normalization and pruning on the extracted triplets, and linking them to the ontology schema to construct an Ontology-based knowledge graph (KG). We apply the proposed pipeline to three representative industrial standards documents (ASTM A578/A578M, API_2W, and ASTM A6/A6M) to construct Ontology-based KG. To evaluate the performance of the constructed KGs, we generate QA datasets covering rule, multi-hop, and table-based queries, and conduct comparative experiments against existing approaches (Chen et al., 2025; Wen et al., 2024).

The contributions of this study are summarized as follows:

- We propose a novel Ontology-based KG construction method that structures industrial standards documents according to their hierarchical organization and rule semantics.
- We provide a domain-specific benchmark by constructing QA datasets of rule, multi-hop, and table types based on real-world industrial standards documents.
- We demonstrate, through both quantitative and qualitative evaluations, that the proposed Ontology-based KG-RAG pipeline outperforms existing LLM-, RAG-, and KG-based systems in reasoning and QA quality.

2 Related Works

2.1 Ontology

Previous studies have defined an ontology as a formal representation of core concepts used within a specific domain and the relationships among those concepts (Miller, 1995; Ashburner et al., 2000). Building on this definition, research on ontology learning, which aims to automatically construct ontologies within a domain, has primarily evolved around concept discovery and relation extraction. More recently, attempts have emerged to leverage LLMs to expand schemas without predefined structures or to generate ontologies in an end-to-end manner (Funk et al., 2023; Lo et al., 2025). However, while these approaches mainly focus on lexical or taxonomic relationships (e.g., *is-a* relations), this study distinguishes itself from prior ontology research by proposing a domain-specific ontology that explicitly reflects the hierarchical structure and rule-oriented concepts inherent in industrial standards documents.

2.2 Knowledge Graph

KG represents entities and relations in the form of triplets, enabling structured knowledge storage and reasoning. Early extracted triplets using rule-based or statistical methods (Etzioni et al., 2004; Yates et al., 2007), and the advent of neural models significantly improved relation extraction performance at the sentence level (Wei et al., 2020; Qi et al., 2018). More recently, LLM-based approaches have been proposed to extract triplets or to construct KGs via prompt-driven methods that can be applied across diverse domains (Huguet Cabot and Navigli, 2021; Zhang and Soh, 2024b; Hu et al., 2024; Wang et al.,

2025). However, while these approaches are effective at modeling simple sentence structures, they struggle to adequately represent compound sentences in which multiple conditions are nested or combined. In contrast, this study structures intra-sentence conditions and their logical compositions based on an ontology, and constructs a KG that reflects these structures, thereby enabling the accurate modeling of complex sentences with multiple overlapping conditions.

3 Method

This section presents a pipeline for transforming industrial standards documents into structured knowledge by formalizing the document's hierarchical structure as an ontology and extracting triplets based on condition–consequence rules. The proposed ontology is designed to jointly represent the document hierarchy and conditional rules, and the overall pipeline consists of three stages, as illustrated in Figure 2.

3.1 Hierarchical Ontology Modeling

Industrial standards documents have a hierarchical structure organized into SECTION–SUBSECTION–CLAUSE levels, and the meanings of various elements, such as tables, conditional statements, and exception rules, are determined by this hierarchical context. Accordingly, this study defines each document section as a core concept and constructs an ontology that captures the relationships among sections. To this end, we design the following two-step procedure.

Hierarchical Structural Decomposition of Documents Sections, clauses, tables, and footnotes in industrial standards documents are identified, and heterogeneous numbering schemes are normalized using rule-based methods to form a consistent hierarchical structure. Tables and footnotes are assigned to their corresponding sections based on their positions, thereby preserving the logical structure of the document. Through this process, the document is represented as the following hierarchical ontology structure:

$$\mathcal{H} = \{\text{Section} \rightarrow \text{Subsection} \rightarrow \text{Subsubsection} \rightarrow (\text{Text} \cup \text{Table}) \rightarrow \text{Footnote}\}. \quad (1)$$

This hierarchical structure enables the valid scope of each rule to be constrained in subsequent stages, providing a foundation for preserving the structural semantics unique to industrial standards documents without loss.

Knowledge Unit Construction Based on the normalized hierarchical structure, we transform the document into knowledge units U_i in the form of title–text pairs:

$$\mathcal{D} = \{U_1, U_2, \ldots, U_N\}, \quad U_i = (t_{\text{title}}^{(i)}, t_{\text{text}}^{(i)}), \quad (2)$$

where N denotes the total number of title–text pairs generated according to the document's hierarchical structure. The title serves as an anchor that represents the hierarchical context (e.g., section, subsection, or clause) to which the rule belongs. The text contains the rule content, including descriptions, table cell entries, conditional statements, and numerical constraints within the corresponding section (see Figure 3). This structure preserves the rule applicability scope during triplet extraction, thereby capturing the characteristics of industrial standards documents in which the meaning of the same term varies depending on the rule's valid scope.

3.2 Sentence-level Conditional Structure Modeling

Industrial standard documents extensively rely on conditional statements of the form "if A then B" to specify manufacturing constraints and applicability criteria. Tabular specifications can likewise be interpreted as conditional structures, where rows and columns define conditions and the corresponding cell values represent consequences, forming a unified **Condition–Consequence** abstraction. However, existing triplet extraction methodologies are limited in their ability to represent such conditional relationships. To address this limitation, we introduce **Sentence-level Conditional Structure Modeling**, which explicitly incorporates conditional characteristics into the triplet extraction process.

Sentence-level Conditional Structure Modeling Algorithm We adopt a triplet extraction algorithm tailored to conditional structures to model conditions at the sentence and table-cell levels. As illustrated in Algorithm 1, each input sentence or cell is decomposed into a condition component and a consequence component, from which triplets are generated based on the structural pattern. For a structure consisting of a single element (Algorithm 1, line 11), a base relation (e.g., `[has_condition]`) is used to generate a triplet. When multiple elements are connected by the same

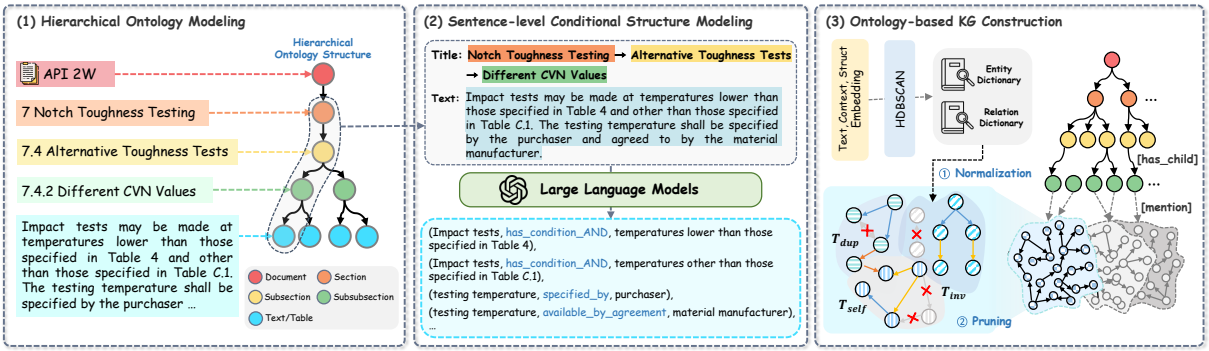

Figure 2: (1) Hierarchical Ontology Modeling normalizes the document into a hierarchical schema, thereby assigning an explicit ontology structure. (2) Sentence-level Conditional Structure Modeling leverages an LLM to extract condition–consequence–based triplets from both textual descriptions and tables. (3) Ontology-based KG Construction refines the final graph through synonym-dictionary–based normalization and pruning.

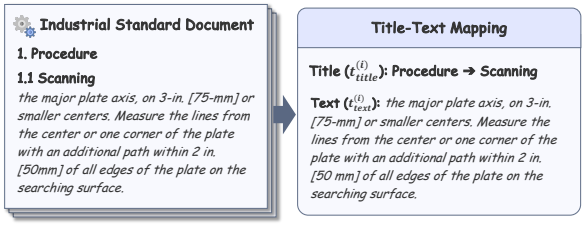

Figure 3: Example of Title–Text Mapping.This figure illustrates how title and text blocks extracted from the industrial standards document A578/A578M are organized according to the hierarchical structure.

logical operator (Algorithm 1, line 13), a relation reflecting the operator (e.g., `[has_condition_AND]`) is applied. In cases where different logical operators are combined (Algorithm 1, line 16), the structure is decomposed into substructures and the same rules are applied recursively. In this process, a `logic_group` node is introduced to represent composite conjunctions. Examples of triplet extraction produced by this algorithm are provided in Figure 7 and Appendix C.

Conditional-aware Triplet Extraction Finally, we construct a prompt $\mathcal{P}$ that integrates the previously defined triplet extraction rules for conditional structures with additional rules for non-conditional sentences. To ensure consistency in numerical expressions, we introduce conversion rules to the International System of Units (SI) and provide few-shot examples based on real-world standard documents. The prompt is presented in Appendix I. Subsequently, the constructed prompt $\mathcal{P}$ and a knowledge unit U_i are jointly provided to an LLM, converting implicit relationships in the text into triplets in the form of `subject-relation-object`. This process can be expressed as follows:

$$T_i = ExtractTriplets(\mathcal{P}, U_i), \tag{3}$$

where $ExtractTriplets$ denotes a function that extracts triplets from both conditional and non-conditional sentences, and T_i represents the set of triplets extracted from U_i. The complete set of triplets for a document D is defined as:

$$T(D) = \bigcup_{i=1}^{N} T_i. \tag{4}$$

The resulting document-level triplet set $T(D)$ serves as the fundamental input for constructing an Ontology-based KG.

3.3 Ontology-based KG Construction

In this section, a three-stage procedure is applied to consolidate redundant expressions from the previously extracted triplets and to remove structural noise, resulting in a coherent KG.

3.3.1 Entity–Relation Synonym Dictionary Construction

To ensure conceptual consistency by unifying synonymous expressions of entities and relations, we construct a synonym dictionary to normalize representations within the KG. In this study, entities and relations are clustered by jointly considering not only text similarity but also contextual and graph-structural information. For relations, we further apply selectional preference based on the subject–object entity type distributions. The centroid of each cluster is selected as the canonical representation to normalize the KG. Detailed construction procedures are provided in Appendix D.

3.3.2 KG Refinement and Pruning

First, we perform conceptual normalization by unifying entities and relations into their canonical forms using the constructed synonym dictionary,

Algorithm 1 Triplet Extraction from Conditional Sentences

Input: Sentence or table cell s
Output: Triplet set T
Initialize an empty triplet set T
Create a case node c
Parse s into a condition structure C and a consequence structure R
$T \leftarrow T \cup MCT(c,$ `has_condition`$, C)$
$T \leftarrow T \cup MCT(c,$ `has_consequence`$, R)$
▷ *MCT denotes ModelConditionalTriplets*
return T

Procedure $MCT(p, rel, S)$
Initialize an empty triplet set T_S
if S contains only one condition or consequence **then**
    $T_S \leftarrow T_S \cup \langle p, rel, S \rangle$
else if all items in S are combined using the same logical operator **then**
    $op \leftarrow$ shared operator of S
    $T_S \leftarrow T_S \cup \langle p, rel\_op, S \rangle$
else
    Separate S into grouped parts and remaining parts
    $op \leftarrow$ logical operator connecting these parts
    for each grouped part G **do**
        ▷ *Introduce a logic_group node to connect grouped substructures to the parent node*
        Create a logic_group node g
        $T_S \leftarrow T_S \cup \langle p, rel\_op, g \rangle$
        $T_S \leftarrow T_S \cup MCT(g, rel, G)$
    end for
    for each remaining part e **do**
        ▷ *Directly connect remaining elements to the parent node*
        $T_S \leftarrow T_S \cup \langle p, rel\_op, e \rangle$
    end for
end if
return T_S

thereby reducing conceptual redundancy and preventing the dispersion of semantically identical nodes. Next, to improve structural efficiency, we apply a multi-stage structural pruning procedure that removes low-information or redundant elements. In this process, nodes are represented as a set V containing subject and object concepts, while semantic relations between nodes are represented as a set R. Each knowledge triplet is expressed in the form (u, r, v), consisting of a subject node $u \in V$, a relation $r \in R$, and an object node $v \in V$. The pruning procedure is conducted in three stages as follows:

(i) ***Self-loop Pruning***: Self-referential relations $T_{\text{self}} = \{(u, r, v) \in T \mid u = v\}$ are semantically uninformative and are therefore removed.

(ii) ***Reciprocal Edge Pruning***: When bidirectional edges coexist, the edge with the lower occurrence frequency is removed, as defined in Eq. 5.

$$T_{\text{inv}} = \Big\{ \underset{t \in \{(u,v),(v,u)\}}{\arg\min} w(t) \mid (u, v), (v, u) \in T \Big\}. \quad (5)$$

If the weights w are identical, no principled directional preference exists; thus, both edges are retained to avoid arbitrary information loss.

(iii) ***Duplicate Triplet Pruning***: When identical triplets are extracted multiple times from the same document, only a single instance is retained:

$$T_{\text{dup}} = \{(u, r, v)_1, \ldots, (u, r, v)_k \mid (u, r, v)_i \in T,\ k > 1\}. \quad (6)$$

After these steps, isolated nodes are removed, yielding the final triplet set as defined in Eq. 7:

$$T_{\text{final}} = T \setminus (T_{\text{self}} \cup T_{\text{inv}} \cup T_{\text{dup}}). \quad (7)$$

The resulting T_{final} is subsequently used for KG construction.

3.3.3 Schema Design for Integrated Ontology-based KG

We design a connection schema to integrate the hierarchical ontology of documents and the KG into a unified graph structure. Each section in a document is defined as a structural unit, and the containment relationship between parent and child sections is modeled using the `[has_child]` relation. In addition, to explicitly connect content-based knowledge contained within each section, we define a `[mention]` relation between section nodes and KG entity nodes referenced in the corresponding section. This relation aligns individual knowledge entities with specific structural locations in the document, enabling a linkage between contextual positional information and Ontology-based structural information that is not provided by conventional KGs. The resulting Ontology-based KG, which unifies hierarchical document structures with content-based knowledge, is implemented in the graph database management system `Neo4j` using the proposed integrated schema.

4 Experiments

This experiment evaluates how effectively the proposed Ontology-based KG captures the complex rules and document structure of industrial standards documents to improve LLM-based reasoning performance. To this end, we conduct comprehensive baseline comparisons and an ablation study.

4.1 Ontology-based KG-RAG

Since the Ontology-based KG constructed in this study does not have an explicit ground-truth set, we evaluate it indirectly through QA performance. To this end, we design an Ontology-based KG-RAG that uses an Ontology-based KG as its knowl-

Table 1: Statistics of the KG constructed from three industrial standards.

	A578/A578M	API_2W	A6/A6M
Nodes	606	1320	10976
Relations	52	121	216
Triplets	1986	5128	53508

Table 2: `IndusSpec-QA` dataset statistics. The dataset is constructed from three industrial standard documents and comprises three types of QA—rule, multi-hop, and table —derived from the original texts.

	A578/A578M	API_2W	A6/A6M
Rule	414	179	200
Multi-hop	79	86	63
Table	8	119	400
Total	501	384	663

edge base and extend two existing KG-RAG approaches—MindMap (Wen et al., 2024) and KG-Retriever (Chen et al., 2025)—into Ontology-based variants with different internal mechanisms and retrieval strategies. The detailed pipeline and implementation specifics are described in Appendix A.

4.2 Setups

Dataset To evaluate how the proposed method operates under different rule structures, we use three industrial standard documents with varying lengths, table structures, and levels of conditional complexity. These documents are widely used in industrial practice, and their structural differences naturally induce varying levels of reasoning difficulty.

- ASTM A578/A578M[1] is a short, rule-centric document that primarily consists of sentence-based rules, making it suitable for evaluating basic rule QA.
- API 2W[2] contains a large number of table-based rules, resulting in a higher level of difficulty for table reasoning.
- ASTM A6/A6M[3] is the most extensive document, encompassing exceptions, multi-hop conditional rules, and large-scale tables, and is therefore suitable for evaluating complex reasoning.

Using these three documents, we construct an Ontology-based KG as shown in Table 1 by applying the proposed method described in Section 3.

[1] `https://standards.globalspec.com/std/4055316/astm-a578-a578m-17`

[2] `https://standards.globalspec.com/std/921032/api-spec-2w`

[3] `https://standards.globalspec.com/std/14621695/astm-a6-a6m-22`

In addition, to validate the constructed Ontology-based KG as described in Section 4.1, we build a QA dataset, `IndusSpec-QA` (see Table 2). The dataset is created by designing question scopes based on the titles (t_{title}) and the main text and tables (t_{text}) of the units U_i defined in Section 3.1. Question drafts are generated using an LLM and then finalized through validation by domain experts. The final dataset covers three types: rule, multi-hop, and table. Specifically, rule refers to queries that explicitly ask about a single sentence or a single condition; multi-hop refers to queries that require stepwise combination of multiple sections or multiple conditions to answer; and table refers to queries that require deriving answers based on information contained in tables.

Baselines To evaluate how effectively the Ontology-based KG reflects the rules and document structure inherent in industrial standards documents, we define four categories of baselines with different levels of knowledge utilization. (1) **LLM-only** generates answers solely from the query without using any external knowledge. (2) **LLM w/full doc** directly injects the entire document into the prompt without retrieval, providing the maximum available information and serving as an upper-bound baseline. (3) **Text-only RAG** is a conventional RAG approach that performs text chunk–based retrieval without considering document structure, using `BM25` (Robertson and Zaragoza, 2009) and `Ada-002` (OpenAI, 2022). (4) **KG-RAG** leverages a KG constructed from the document but does not apply an ontology, representing a conventional graph-based approach; comparison with Ontology-based KG-RAG enables analysis of the contribution of the ontology. Implementation details of each baseline are provided in Appendix B.

Evaluation Metrics To evaluate response quality, we use F1, BERTScore, BLEU, and Rouge. F1 measures classification accuracy, BERTScore captures semantic similarity, and BLEU and Rouge assess n-gram– and phrase-level textual alignment.

4.3 Main Results

Table 3 presents the performance comparison results against four baseline model categories across three industrial standards documents. From this experiment, we derive the following key observations:

Table 3: Performance comparison across datasets on three industrial standard documents. Scores are weighted averages over rule-based, table-based, and multi-hop QA tasks, computed according to the data proportions. **Best** and second-best results are shown in bold and underlined, respectively.

Type	Method	A578/A578M				API_2W				A6/A6M			
		F1	BERTScore	BLEU	Rouge	F1	BERTScore	BLEU	Rouge	F1	BERTScore	BLEU	Rouge
LLM-only	GPT-4o-mini	0.013	0.806	0.004	0.020	0.010	0.800	0.003	0.014	0.002	0.790	0.002	0.003
	gemini-2.0-flash	0.000	0.802	0.000	0.004	0.000	0.797	0.000	0.002	0.003	0.790	0.002	0.003
	DeepSeek-v3.1	0.043	0.817	0.016	0.049	0.040	0.809	0.018	0.046	0.036	0.800	0.016	0.039
LLM w/full doc	GPT-4o-mini	0.308	0.867	0.157	0.357	0.298	0.864	0.120	0.361	0.152	0.826	0.066	0.174
	gemini-2.0-flash	0.282	0.861	0.159	0.315	0.276	0.853	0.142	0.314	0.206	0.834	0.083	0.273
	DeepSeek-v3.1	0.077	0.874	0.165	0.384	0.382	0.884	0.156	0.463	0.130	0.824	0.069	0.152
Text-only RAG	BM25+Qwen1.5-14B	0.306	0.876	0.129	0.381	0.306	0.881	0.124	0.431	0.154	0.837	0.054	0.208
	BM25+Mistral-7B	0.279	0.865	0.118	0.346	0.242	0.859	0.112	0.300	0.128	0.837	0.050	0.183
	BM25+GPT-4o-mini	0.366	0.880	0.193	0.433	0.308	0.868	0.140	0.357	0.160	0.831	0.070	0.195
	BM25+gemini-2.0	0.340	0.876	0.184	0.405	0.291	0.859	0.137	0.350	0.159	0.825	0.076	0.184
	Dense+Qwen1.5-14B	0.264	0.870	0.104	0.322	0.372	0.891	0.134	0.475	0.179	0.852	0.067	0.223
	Dense+Mistral-7B	0.255	0.859	0.100	0.318	0.277	0.864	0.112	0.348	0.152	0.840	0.059	0.223
	Dense+GPT-4o-mini	0.314	0.871	**0.499**	0.370	0.413	0.888	0.160	0.476	0.175	0.828	0.079	0.205
	Dense+gemini-2.0	0.309	0.870	0.167	0.359	0.348	0.875	0.164	0.471	0.167	0.825	0.077	0.203
KG-RAG	MindMap (Wen et al., 2024)	0.370	0.879	0.224	0.426	0.334	0.887	0.145	0.412	0.223	0.851	0.113	0.255
	KG Retriever (Chen et al., 2024)	0.319	0.874	0.170	0.476	0.223	0.862	0.090	0.301	0.190	0.853	0.067	0.281
Ontology-based KG-RAG	Ontology-based MindMap	0.511	0.911	0.313	0.585	**0.494**	**0.915**	0.265	**0.569**	**0.441**	**0.892**	**0.186**	**0.518**
	Ontology-based KG-Retriever	**0.639**	**0.933**	0.424	**0.653**	0.448	0.909	**0.268**	0.527	0.191	0.854	0.084	0.194

- **Ontology-based KG-RAG achieves dominant performance across all datasets.** The average F1 score reaches 0.454, showing a substantial improvement over existing methods. This demonstrates that Ontology-based KGs are a critical factor for performance gains in specialized domains such as industrial standards documents.
- **LLM-only methods are limited in specialized domains.** With an average F1 of only 0.016, the results indicate that LLMs alone cannot generate meaningful answers in domains such as industrial standards without external knowledge.
- **LLM w/full doc suffers from information overload.** Although the average F1 improves to 0.235 compared to LLM-only, it remains far below the performance of the proposed Ontology-based KG-RAG. This suggests that naively injecting entire documents leads to information overload and fails to support effective reasoning.
- **KG-RAG shows limited gains over Text-only RAG.** The performance gap between Text-only RAG (average F1: 0.262) and conventional KG-RAG (average F1: 0.277) is relatively small, indicating that existing KG-RAG approaches do not fully exploit the potential of structural knowledge during KG construction. In contrast, Ontology-based KG-RAG achieves a 64% improvement over conventional KG-RAG (0.277 $\rightarrow$ 0.454), empirically validating the importance of domain-specific ontology design.
- **Ontology-based KG-RAG maintains stable performance regardless of document length.**

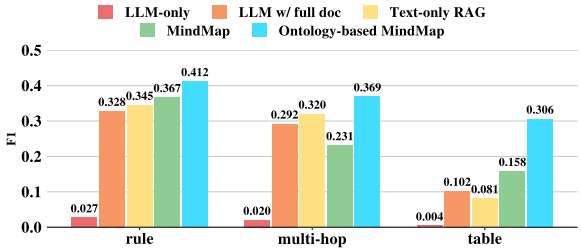

Figure 4: Comparison of F1 Performance by QA Data Type. For three documents, we compare the average F1 of five model categories across three QA data types: rule, multi-hop, and table.

Overall, performance is relatively lower on the longest document, A6/A6M (63 pages), compared to the shortest document, A578/A578M (5 pages), suggesting increased difficulty in evidence extraction as document length grows. Nevertheless, Ontology-based KG-RAG maintains comparatively stable performance and demonstrates robustness on longer documents.

In addition, Figure 4 presents a comparison of the average F1 performance of models across different methodologies, showing that model performance varies by task type. Ontology-based KG-RAG achieves the highest average F1 across all task types. In particular, it shows a 93.7% improvement in the table task over conventional KG-RAG (0.158 $\rightarrow$ 0.306), a 12.3% improvement in the rule task (0.367 $\rightarrow$ 0.412), and a 59.7% improvement in the multi-hop task (0.231 $\rightarrow$ 0.369). These results can be attributed to the accurate representation of row–column conditions and applicability scopes of table-based rules as structured relations, which preserves the logical composition among rules. This suggests that table-based rules

Table 4: Ablation study on API_2W rule data. This table presents a performance comparison of the LLM under four settings: with documents only, with ontology, with KG, and with both ontology and KG.

	F1	BERTScore	BLEU	Rouge
LLM w/doc	0.298	0.864	0.120	0.361
+ KG	0.334	0.887	0.145	0.412
+ Ontology	0.413	0.888	0.160	0.476
+ Ontology + KG	**0.494**	**0.915**	**0.265**	**0.569**

and hierarchical constraints, which frequently appear in industrial standards documents, are prone to distortion in text-based LLMs or simple RAG approaches, whereas structural knowledge representations such as Ontology-based KGs enable more accurate reasoning. The details of Figure 4 are provided in the Appendix Table 5,6,7. Overall, these results demonstrate that an Ontology-based KG approach is essential for specialized domains such as industrial standards documents. In particular, the performance gains over the LLM-only baseline clearly demonstrate the importance of integrating external knowledge and constructing a systematic, domain-specific KG.

4.4 Additional Analyses

Effect of KG and Ontology Integration To analyze the impact of using Ontology and KG individually and in combination, we conduct an ablation study on the API_2W dataset. Specifically, we compare four configurations: (i) an LLM using only document information, (ii) a KG-only setting, (iii) an ontology-only setting that captures document structural information, and (iv) a setting that jointly uses both the ontology and the KG. As shown in Table 4, the combined configuration achieves the highest performance across all evaluation metrics, with the F1 score improving from 0.298 (document-only) to 0.494, corresponding to an approximately 65% increase. Compared to the KG-only setting (F1: 0.334), the combined model attatins an improvement of approximately 47%. Notably, the ontology-only configuration outperforms the KG-only configuration. This observation suggests that explicitly encoding the hierarchical document structure and rule context through an ontology plays a more critical role in enhancing rule-based reasoning performance than relational information among individual entities alone. Overall, the KG complements the structural context provided by the ontology, and the most effective reasoning is achieved when both modules are used together.

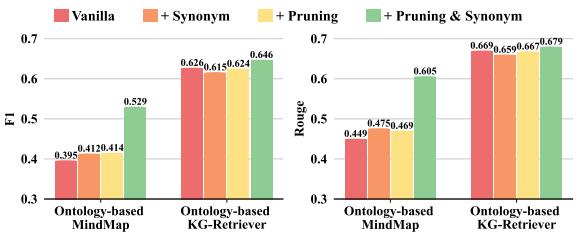

Figure 5: Effect of the synonym dictionary and pruning modules on QA performance (F1 and Rouge) on API_2W rule dataset. Ontology-based MindMap shows progressive performance gains with sequential module application, whereas Ontology-based KG-Retriever performs best only when both modules are applied together.

Effect of Synonym Dictionary and Pruning In this experiment, we compare the contributions of the synonym dictionary and pruning modules in the KG construction process using F1 and Rouge scores. As shown in Figure 5, the performance of Ontology-based MindMap improves gradually as the two modules are applied sequentially. In contrast, for Ontology-based KG-Retriever, a slight decrease in F1 or Rouge is observed when either the synonym dictionary or pruning is applied in isolation, suggesting that partial information loss may occur when individual modules are used alone. However, when both modules are applied jointly, performance improves most substantially, demonstrating that the two procedures operate in a complementary manner. These results support the necessity of combining synonym dictionary–based normalization and pruning in the KG refinement process.

5 Conclusion

This study presents an Ontology-based KG construction method that integrates hierarchical ontology modeling with logic-aware triplet extraction to transform industrial standards documents into structured knowledge. By explicitly encoding document hierarchy, conditional logic, and table-based constraints, the proposed approach addresses key limitations of conventional text-based methods. We further validate the effectiveness of the proposed pipeline through an Ontology-based KG-RAG system, which demonstrates strong performance gains, particularly in conditional reasoning and table interpretation. These results highlight the importance of structured domain knowledge and provide a foundation for future domain-specific RAG systems and automated standards interpretation in industrial applications.

Limitations

The proposed approach is optimized for documents with explicit structural cues, such as hierarchical sections, conditional rules, and tables, which effectively stabilize LLM-based reasoning. Consequently, its direct applicability to fully unstructured open-domain texts, where such structural signals are scarce, may be limited. Nevertheless, the pipeline is not confined to industrial standards and can be applied to other rule-centric documents with explicit structural units, including technical manuals, legal or contractual texts, and policy documents. Moreover, the proposed Sentence-level Conditional Structure Modeling provides a general mechanism for decomposing complex nested conditions and multi-exception rules into logical units, making it applicable across domains with complex rule structures. Finally, as the approach relies on LLM-based knowledge extraction, its performance depends on model stability. Future work will explore training domain-specific local models using the extracted ontology and rule patterns, thereby reducing reliance on external LLMs and improving reliability, reproducibility, and operational robustness.

Ethical statement

This study proposes an automated system for question answering based on industrial standards documents. In conducting this research, we carefully considered the following aspects to ensure compliance with research ethics. First, regarding data usage and copyright, the industrial standards documents used in the experiments were utilized strictly for academic research purposes, and neither distribution of the original data nor any form of commercial use was permitted. Second, concerning safety and potential misuse, although the proposed model demonstrates high accuracy, decision-making in industrial settings is closely tied to safety and therefore the model outputs should not be blindly trusted without expert review. The proposed system is designed as a decision-support tool for engineers, and we explicitly acknowledge that the possibility of errors caused by hallucinations cannot be entirely eliminated.

References

M.M. Ashburner, C.A.C. Ball, Judith Blake, David Botstein, Heather Butler, J.M.J. Cherry, Allan Peter Davis, Kara Dolinski, Selina Dwight, Janan Eppig, Midori Harris, D.P. Hill, Laurie Issel-Tarver, A Kasarskis, Suzanna Lewis, John Matese, J.E. Richardson, M Ringwald, and G.M. Rubin. 2000. Gene ontology: tool for the unification of biology. the gene ontology consortium. *Nat Genet*, 25:25–29.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, Binyuan Hui, Luo Ji, Mei Li, Junyang Lin, Runji Lin, Dayiheng Liu, Gao Liu, Chengqiang Lu, Keming Lu, and 29 others. 2023. Qwen technical report. *Preprint*, arXiv:2309.16609.

Mikhail Belkin and Partha Niyogi. 2001. Laplacian eigenmaps and spectral techniques for embedding and clustering. In *Proceedings of the 15th International Conference on Neural Information Processing Systems: Natural and Synthetic*, NIPS'01, page 585–591, Cambridge, MA, USA. MIT Press.

Nils Wakan Boecking, Parastou Azari Gargari, Sarah Reichel, Sven Hoffmann, Tobias Richter, and Volker Wulf. 2025. Chat with standards: An assistant for the provision of normative knowledge for practical use in welding. In *Proceedings of the 2025 ACM Designing Interactive Systems Conference*, DIS '25, page 2171–2188, New York, NY, USA. Association for Computing Machinery.

Weijie Chen, Ting Bai, Jinbo Su, Jian Luan, Wei Liu, and Chuan Shi. 2025. Kg-retriever: Efficient knowledge indexing for retrieval-augmented large language models. *Preprint*, arXiv:2412.05547.

DeepMind / Google. 2025. Gemini 2.0 flash. `https://deepmind.google/models/gemini/flash/`.

DeepSeek-AI, Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, and 1 others. 2025. Deepseek-v3 technical report. *Preprint*, arXiv:2412.19437.

Oren Etzioni, Michael Cafarella, Doug Downey, Stanley Kok, Ana-Maria Popescu, Tal Shaked, Stephen Soderland, Daniel S. Weld, and Alexander Yates. 2004. Web-scale information extraction in knowitall: (preliminary results). In *Proceedings of the 13th International Conference on World Wide Web*, WWW '04, page 100–110, New York, NY, USA. Association for Computing Machinery.

Maurice Funk, Simon Hosemann, Jean Christoph Jung, and Carsten Lutz. 2023. Towards ontology construction with language models. *Preprint*, arXiv:2309.09898.

Johannes Gasteiger, Aleksandar Bojchevski, and Stephan Günnemann. 2022. Predict then propagate: Graph neural networks meet personalized pagerank. *Preprint*, arXiv:1810.05997.

Yuelin Hu, Futai Zou, Jiajia Han, Xin Sun, and Yilei Wang. 2024. Llm-tikg: Threat intelligence knowledge graph construction utilizing large language model. *Computers & Security*, 145:103999.

Pere-Lluís Huguet Cabot and Roberto Navigli. 2021. REBEL: Relation extraction by end-to-end language generation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2370–2381, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Zarina Kabzhan, Alexandr Shakhnovich, Sergey Gorshkov, Yussuf Yemenov, Fedor Gorshkov, and Nazym Shogelova. 2025. Semantic and ontology-based analysis of regulatory documents for construction industry digitalization. *Frontiers in Built Environment*, 11:1575913.

Andy Lo, Albert Q. Jiang, Wenda Li, and Mateja Jamnik. 2025. End-to-end ontology learning with large language models. In *Proceedings of the 38th International Conference on Neural Information Processing Systems*, NIPS '24, Red Hook, NY, USA. Curran Associates Inc.

Leland McInnes, John Healy, and Steve Astels. 2017. hdbscan: Hierarchical density based clustering. *Journal of Open Source Software*, 2(11):205.

George A. Miller. 1995. Wordnet: a lexical database for english. *Commun. ACM*, 38(11):39–41.

Mistral AI team. 2023. Mistral 7b. `https://mistral.ai/news/announcing-mistral-7b`.

OpenAI. 2022. New and improved embedding model. `https://openai.com/index/new-and-improved-embedding-model/`.

OpenAI. 2025. GPT-5 System Card. `https://openai.com/index/introducing-gpt-5/`.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, and Lama Ahmad. 2024. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Siyuan Qi, Wenguan Wang, Baoxiong Jia, Jianbing Shen, and Song-Chun Zhu. 2018. Learning human-object interactions by graph parsing neural networks. *Preprint*, arXiv:1808.07962.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *Preprint*, arXiv:1908.10084.

Stephen Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389.

Nilay Tufek. 2023. Semantic information extraction from multi-modal technical document. In *2023 18th Iberian Conference on Information Systems and Technologies (CISTI)*, pages 1–4.

Qingwang Wang, Chaohui Li, Yi Liu, Qiubai Zhu, Jian Song, and Tao Shen. 2025. An adaptive framework embedded with llm for knowledge graph construction. *IEEE Transactions on Multimedia*, 27:2912–2923.

Zhepei Wei, Jianlin Su, Yue Wang, Yuan Tian, and Yi Chang. 2020. A novel cascade binary tagging framework for relational triple extraction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1476–1488, Online. Association for Computational Linguistics.

Yilin Wen, Zifeng Wang, and Jimeng Sun. 2024. MindMap: Knowledge graph prompting sparks graph of thoughts in large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10370–10388, Bangkok, Thailand. Association for Computational Linguistics.

Alexander Yates, Michele Banko, Matthew Broadhead, Michael Cafarella, Oren Etzioni, and Stephen Soderland. 2007. TextRunner: Open information extraction on the web. In *Proceedings of Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT)*, pages 25–26, Rochester, New York, USA. Association for Computational Linguistics.

Bowen Zhang and Harold Soh. 2024a. Extract, define, canonicalize: An LLM-based framework for knowledge graph construction. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 9820–9836, Miami, Florida, USA. Association for Computational Linguistics.

Bowen Zhang and Harold Soh. 2024b. Extract, define, canonicalize: An llm-based framework for knowledge graph construction. *Preprint*, arXiv:2404.03868.

Contents

A Ontology-based KG-RAG Details

A.1 Ontology-based MindMap

Ontology-based MindMap integrates the original MindMap pipeline with the Ontology-based KG, such that the entire process of evidence retrieval and reasoning is constrained by the hierarchical document structure and rule-level semantic context (see Figure 6). The proposed methodology consists of four stages: **Entity Extraction**, **Ontology-Constrained Evidence Graph Mining**, **Evidence Graph Aggregation**, and **LLM-integrated Reasoning**.

In the **Entity Extraction** stage, key domain entity candidates are identified from the query using an LLM-based extractor. Rather than being used directly, the extracted entities are mapped to actual KG entities via cosine similarity matching with pre-constructed Ontology-based KG entity embeddings. The resulting entity set serves as anchor points for subsequent graph exploration.

In the **Ontology-Constrained Evidence Graph Mining** stage, an evidence graph is explored starting from the matched entities, while all exploration processes are strictly constrained within the hierarchical ontology structure of the document. Specifically, candidate evidence is restricted to query-relevant document sections by leveraging the `[has_child]` and `[mention]` relations, effectively suppressing structural noise that may arise from global KG traversal. Neighbor search is then performed to collect local rules and entity relations, followed by shortest-path search between entity pairs. During this process, ontology relations that represent document structure are excluded from the reasoning paths, ensuring that only substantive rule-level and semantic relations are used as reasoning evidence. In particular, rules extracted from tables and conditional statements are preferentially collected in a form that preserves condition–consequence structures via the `[has_condition_AND/_OR]` and `[has_consequence_AND/_OR]` relations.

In the **Evidence Graph Aggregation** stage, the retrieved triplets are unified into a single reasoning graph through duplicate removal and structural normalization. Rather than simply enumerating triplets, evidence is aggregated at the level of condition–consequence units (i.e., case-level aggregation) to prevent fragmentation of the logical semantics of rules. The resulting reasoning graph thus simultaneously reflects both the hierarchical document context and the underlying rule structures.

Finally, in the **LLM-integrated Reasoning** stage, the reasoning graph is injected into the LLM prompt to perform question answering. The LLM conducts reasoning along ontology-refined evidence paths, suppressing structure-agnostic inference and generating consistent answers grounded in document structure. Through this process, Ontology-based MindMap realizes a collaborative reasoning pipeline that combines the structural strengths of KGs with the language-based reasoning capabilities of LLMs.

A.2 Ontology-based KG-Retriever

The proposed methodology consists of a two-stage evidence extraction process at the **Ontology KG-Level** and the **Global KG-Level**, followed by a final stage that integrates the extracted evidence and injects it into the LLM prompt.

At the **Ontology KG-Level**, each section title of a document is defined as a semantic unit, and each section is represented as a pair $(t^{(i)}_{\text{title}}, t^{(i)}_{\text{text}})$, where $t^{(i)}_{\text{title}}$ denotes the title of the i-th section and $t^{(i)}_{\text{text}}$ denotes the set of body sentences associated with the corresponding title. First, the sentences in $t^{(i)}_{\text{text}}$ are encoded using a sentence embedding function, Sentence-BERT (`all-MiniLM-L6-v2`), and averaged via mean pooling to generate a section embedding $\mathbf{d}_i$. The query q is likewise transformed into a

Table 5: A578/A578M Performance Comparison across different question types. Performance comparison of various QA models on table-, rule-, and multi-hop–based questions.

Type	Method	table				rule				multi-hop			
		F1	BERTScore	BLEU	Rouge	F1	BERTScore	BLEU	Rouge	F1	BERTScore	BLEU	Rouge
LLM-only	GPT-4o-mini	0.000	0.773	0.000	0.000	0.013	0.806	0.004	0.021	0.017	0.806	0.006	0.016
	gemini-2.0-flash	0.000	0.773	0.000	0.000	0.000	0.802	0.000	0.005	0.000	0.806	0.000	0.000
	DeepSeek-v3.1	0.000	0.773	0.000	0.000	0.048	0.819	0.018	0.055	0.021	0.814	0.005	0.025
LLM w/full doc	GPT-4o-mini	0.000	0.776	0.000	0.031	0.297	0.866	0.154	0.339	0.388	0.881	0.167	0.454
	gemini-2.0-flash	0.000	0.773	0.000	0.000	0.276	0.861	0.163	0.307	0.344	0.872	0.157	0.388
	DeepSeek-v3.1	0.000	0.766	0.000	0.195	0.033	0.876	0.174	0.383	0.317	0.890	0.139	0.407
Text-only RAG	BM25+Qwen1.5-14B	0.000	0.821	0.000	0.219	0.312	0.876	0.137	0.38	0.307	0.880	0.133	0.402
	BM25+Mistral-7B	0.000	0.776	0.000	0.179	0.276	0.865	0.121	0.331	0.320	0.874	0.114	0.441
	BM25+GPT-4o-mini	0.064	0.812	0.009	0.343	0.360	0.880	0.194	0.419	0.430	0.890	0.209	0.516
	BM25+gemini-2.0-flash	0.000	0.800	0.000	0.250	0.332	0.875	0.184	0.391	0.416	0.887	0.201	0.493
	Dense+Qwen1.5-14B	0.000	0.828	0.000	0.324	0.262	0.869	0.109	0.310	0.302	0.882	0.104	0.389
	Dense+Mistral-7B	0.011	0.775	0.001	0.185	0.251	0.859	0.101	0.305	0.290	0.867	0.105	0.390
	Dense+GPT-4o-mini	0.110	0.827	0.018	0.411	0.295	0.868	**0.560**	0.342	0.432	0.862	0.225	0.516
	Dense+gemini-2.0	0.000	0.803	0.000	0.281	0.296	0.868	0.161	0.337	0.407	0.887	0.216	0.480
KG-RAG	MindMap (Wen et al., 2024)	0.538	0.907	0.292	0.594	0.372	0.878	0.238	0.417	0.345	0.883	0.146	0.455
	KG Retriever (Chen et al., 2024)	0.118	0.822	**0.730**	0.750	0.338	0.878	0.170	0.403	0.253	0.859	0.119	0.529
Ontology-based KG-RAG	Ontology-based MindMap	**0.875**	**0.978**	0.725	**1.000**	0.511	0.911	0.329	0.573	0.476	0.907	0.226	0.607
	Ontology-based KG-Retriever	0.351	0.876	0.030	0.375	**0.639**	**0.934**	0.440	**0.654**	**0.670**	**0.935**	**0.380**	**0.676**

Table 6: API_2W Performance Comparison across different question types. Performance comparison of various QA models on table-, rule-, and multi-hop–based questions.

Type	Method	table				rule				multi-hop			
		F1	BERTScore	BLEU	Rouge	F1	BERTScore	BLEU	Rouge	F1	BERTScore	BLEU	Rouge
LLM-only	GPT-4o-mini	0.000	0.800	0.000	0.000	0.010	0.797	0.002	0.013	0.025	0.806	0.007	0.035
	gemini-2.0-flash	0.000	0.800	0.000	0.000	0.000	0.795	0.000	0.001	0.000	0.799	0.000	0.006
	DeepSeek-v3.1	0.005	0.803	0.001	0.005	0.005	0.809	0.003	0.058	0.069	0.818	0.017	0.079
LLM w/full doc	GPT-4o-mini	0.222	0.862	0.055	0.312	0.325	0.862	0.155	0.372	0.346	0.872	0.139	0.408
	gemini-2.0-flash	0.121	0.828	0.037	0.182	0.369	0.867	0.208	0.387	0.295	0.859	0.151	0.326
	DeepSeek-v3.1	0.352	0.888	0.078	0.472	0.412	0.884	0.208	0.471	0.360	0.879	0.155	0.435
Text-only RAG	BM25+Qwen1.5-14B	0.098	0.882	0.017	0.388	0.411	0.879	0.185	0.464	0.375	0.886	0.147	0.424
	BM25+Mistral-7B	0.053	0.836	0.015	0.125	0.352	0.870	0.174	0.402	0.273	0.866	0.119	0.330
	BM25+GPT-4o-mini	0.156	0.847	0.027	0.234	0.404	0.881	0.205	0.439	0.316	0.872	0.162	0.357
	BM25+gemini-2.0-flash	0.110	0.830	0.025	0.218	0.396	0.873	0.191	0.428	0.322	0.870	0.178	0.369
	Dense+Qwen1.5-14B	0.334	0.906	0.061	0.531	0.392	0.883	0.171	0.458	0.383	0.889	0.159	0.434
	Dense+Mistral-7B	0.141	0.854	0.024	0.249	0.359	0.869	0.163	0.406	0.294	0.869	0.127	0.364
	Dense+GPT-4o-mini	0.405	0.897	0.072	0.502	0.439	0.886	0.214	0.468	0.372	0.879	0.171	0.419
	Dense+gemini-2.0-flash	0.222	0.863	0.045	0.496	0.449	0.884	0.234	0.491	0.313	0.872	0.185	0.393
KG-RAG	MindMap (Wen et al., 2024)	0.468	0.929	0.173	0.610	0.303	0.873	0.157	0.358	0.214	0.860	0.081	0.256
	KG Retriever (Chen et al., 2024)	0.135	0.850	0.027	0.232	0.344	0.873	0.162	0.420	0.081	0.854	0.027	0.151
Ontology-based KG-RAG	Ontology-based MindMap	**0.531**	**0.933**	**0.211**	**0.621**	0.529	0.917	0.323	0.605	0.371	0.888	0.217	0.421
	Ontology-based KG-Retriever	0.135	0.879	0.036	0.317	**0.646**	**0.933**	**0.419**	**0.679**	**0.470**	**0.902**	**0.274**	**0.600**

query embedding $\mathbf{q}$ in the same embedding space, and the cosine similarity $\cos(\mathbf{q}, \mathbf{d}_i)$ is computed to measure section-level semantic relevance.

Based on this similarity-based ranking, the top three sections are selected and formed into a coarse-grained candidate set. Subsequently, a 1-hop search is first performed over the KG triplets contained within each section, and if sufficient evidence is not obtained, the search is progressively expanded up to 2-hop to collect additional evidence. Specifically, for each section, 20, 10, and 5 triplets are retrieved in descending order of importance, respectively, resulting in a total of 35 fine-grained triplets. This process enables the simultaneous incorporation of query-relevant local context and the hierarchical structure of the document.

At the **Global KG-Level**, section boundaries are disregarded and a global search is conducted over the entire KG. Using the query embedding $\mathbf{q}$ as the reference, both 1-hop and 2-hop neighbor searches are performed to collect a total of 30 triplets, thereby capturing global structural cues beyond local constraints. In addition, a Guided search module is applied, which directly computes cosine similarity between the query embedding and the embeddings of triplet textual representations (including table-based representations), and retrieves an additional top 20 triplets based on this similarity. The guided search operates as an independent module that complements global structure-based exploration from a semantic similarity perspective.

Finally, evidence collected from the Ontology KG-Level and the Global KG-Level is integrated to form a total of 65 candidate triplets. These candidates are then re-ranked based on semantic similarity to the query, and the top 30 triplets are selected

Table 7: A6/A6M Performance Comparison across different question types. Performance comparison of various QA models on table-, rule-, and multi-hop–based questions.

Type	Method	table				rule				multi-hop			
		F1	BERTScore	BLEU	Rouge	F1	BERTScore	BLEU	Rouge	F1	BERTScore	BLEU	Rouge
LLM-only	GPT-4o-mini	0.004	0.784	0.003	0.005	0.000	0.799	0.000	0.000	0.000	0.803	0.000	0.000
	gemini-2.0-flash	0.004	0.784	0.003	0.004	0.000	0.799	0.000	0.000	0.003	0.804	0.001	0.003
	DeepSeek-v3.1	0.007	0.786	0.003	0.008	0.091	0.824	0.044	0.099	0.047	0.817	0.011	0.053
LLM w/full doc	GPT-4o-mini	0.083	0.805	0.016	0.100	0.244	0.857	0.136	0.269	0.294	0.864	0.163	0.347
	gemini-2.0-flash	0.110	0.806	0.008	0.132	0.370	0.879	0.216	0.410	0.290	0.865	0.139	0.350
	DeepSeek-v3.1	0.006	0.792	0.003	0.014	0.343	0.876	0.184	0.386	0.243	0.862	0.121	0.294
Text-only RAG	BM25+Qwen1.5-14B	0.053	0.817	0.009	0.112	0.311	0.867	0.121	0.356	0.295	0.869	0.124	0.344
	BM25+Mistral-7B	0.036	0.819	0.007	0.085	0.266	0.862	0.107	0.321	0.274	0.871	0.138	0.37
	BM25+GPT-4o-mini	0.043	0.803	0.010	0.072	0.333	0.871	0.161	0.376	0.349	0.877	0.164	0.396
	BM25+gemini-2.0-flash	0.016	0.788	0.003	0.030	0.367	0.882	0.195	0.423	0.346	0.876	0.165	0.401
	Dense+Qwen1.5-14B	0.078	0.836	0.013	0.116	0.339	0.876	0.152	0.387	0.314	0.876	0.141	0.378
	Dense+Mistral-7B	0.064	0.827	0.012	0.145	0.292	0.858	0.131	0.336	0.264	0.864	0.127	0.362
	Dense+GPT-4o-mini	0.063	0.799	0.012	0.085	0.340	0.871	0.178	0.380	0.362	0.877	0.191	**0.414**
	Dense+gemini-2.0-flash	0.044	0.792	0.006	0.075	0.364	0.876	0.192	0.399	0.324	0.874	0.162	0.390
KG-RAG	MindMap (Wen et al., 2024)	0.117	0.834	0.034	0.130	0.404	0.877	0.255	0.458	0.322	0.880	0.167	0.405
	KG Retriever (Chen et al., 2024)	0.107	0.843	0.013	0.224	0.356	0.874	0.178	0.413	0.185	0.853	0.057	0.224
Ontology-based KG-RAG	Ontology-based MindMap	**0.468**	**0.898**	**0.135**	**0.562**	0.438	0.885	**0.285**	**0.472**	0.344	0.878	**0.192**	0.398
	Ontology-based KG-Retriever	0.124	0.842	0.029	0.106	**0.474**	**0.897**	0.163	0.303	**0.355**	**0.881**	0.185	0.405

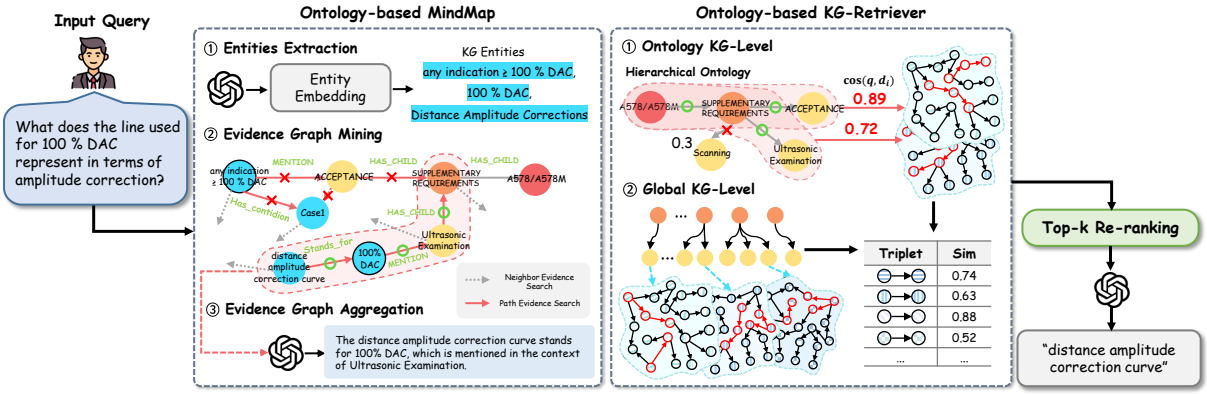

Figure 6: Ontology-based KG-RAG pipeline. (Left) Ontology-based MindMap selects evidence by reflecting the document's hierarchical ontology structure. (Right) Ontology-based KG-Retriever combines Ontology-based search with global KG exploration. Finally, it extracts triples through re-ranking and inserts them into the prompt.

and injected into the LLM (`GPT-4o-mini`) prompt for response generation. Through this multi-stage evidence extraction and re-ranking procedure, the Ontology-based KG-Retriever simultaneously reflects the semantic context of the query and the structural information of the KG, enabling more accuracy of downstream question answering tasks.

A.3 Implementation Details

The Ontology-based KG used in the Ontology-based MindMap and Ontology-based KG-Retriever pipelines was stored in the `Neo4j` Aura environment. Sentence-BERT (`all-MiniLM-L6-v2`) was employed to generate embeddings for queries, section titles, KG triplet texts, and table-based rule representations, and all embeddings were L2-normalized for cosine similarity computation. In the Ontology-based MindMap, the top three section paths were selected based on semantic similarity between the query and section titles. The retrieval budget was adaptively allocated to extract more evidence from the most relevant sections, and graph traversal following the document structure was allowed up to a maximum of 3 hops. In the Ontology-based KG-Retriever, the graph traversal depth was limited to a maximum of 2 hops. At the Ontology KG-level, retrieval was restricted to section-scoped exploration, while at the Global KG-level, global exploration over the entire KG was performed under the same hop constraint. For both pipelines, final evidence selection was conducted using Maximal Marginal Relevance (MMR), with the λ parameter set to 0.7. Up to 30 triplets were used as evidence for LLM input. Response generation was performed using the `GPT-4o-mini` model, and the temperature was fixed at 0.2 across all experiments to ensure response stability and reproducibility.

B Baselines & Implementation Details

- **LLM-only** generates answers directly from the LLM using only the query as input, without leveraging any external knowledge or document context. In this setting, `GPT-4o-mini`, `gemini-2.0-flash`, and `DeepSeek-v3.1` were used (OpenAI et al., 2024; DeepMind / Google, 2025; DeepSeek-AI et al., 2025). All queries were provided using an identical prompt template, and this configuration served as a baseline for evaluating the intrinsic reasoning capability of the models.

- **LLM w/full doc** provides the entire document together with the query as input context to the LLM. To accommodate input length constraints, documents were split into token-based segments, and partial responses were generated for each query–document pair. These partial responses were then aggregated into a single context, which was re-input to the LLM to generate the final answer. Depending on document length, between 2 and up to 20 segmented prompts were generated. The same models as in the LLM-only setting were used for response generation.

- **BM25 retriever + LLMs** is a traditional information retrieval approach that evaluates keyword relevance between queries and documents based on term frequency (TF) and inverse document frequency (IDF) (Robertson and Zaragoza, 2009). BM25 retrieval was implemented using LangChain's `BM25Retriever`. Documents were segmented at the section level based on section headers, and the top $k = 5$ documents were retrieved for each query. The retrieved documents were directly inserted into the LLM prompt for answer generation. For comparison, `Qwen1.5-14B`, `Mistral-7B`, `GPT-4o-mini`, and `gemini-2.0-flash` were used (Bai et al., 2023; Mistral AI team, 2023; OpenAI et al., 2024; DeepMind / Google, 2025).

- **Ada-002 embedding retriever + LLMs** uses an OpenAI embedding model to vectorize text and retrieves the top-k documents based on semantic similarity, capturing contextual meaning more effectively than BM25. In this experiment, OpenAI's `text-embedding-ada-002` model was employed (OpenAI, 2022). The same section-level documents as in BM25 were used as retrieval units; additionally, each document was further split into chunks with `chunk_size=500` and `overlap=50`. Cosine similarity between query and document chunk embeddings was computed, and the top $k = 5$ chunks were retrieved. The retrieved chunks were used as LLM input context, and `Qwen1.5-14B`, `Mistral-7B`, `GPT-4o-mini`, and `gemini-2.0-flash` were used for response generation.

- **MindMap** is a reasoning pipeline that integrates LLMs with a KG and was implemented using a KG stored in a Neo4j graph database. Entities were extracted from the query using an LLM (`GPT-4o-mini`) and matched to the top 5 KG entities based on cosine similarity with pre-built KG entity embeddings (`all-MiniLM-L6-v2`), using a similarity threshold of 0.70. Graph traversal was allowed up to 3 hops, with up to 3 shortest paths collected between entities, and up to 30 neighboring triplet candidates permitted during neighborhood exploration. Final evidence consisted of both path-based and neighbor-based triplets, which were inserted into the LLM context to generate the answer. `GPT-4o-mini` was also used for response generation (Wen et al., 2024).

- **KG-Retriever** consists of a doc-level stage that first selects documents relevant to the query and a kg-level retrieval stage that directly retrieves KG triplets semantically similar to the query from the selected documents. In this work, since documents are assumed to exist independently, the doc-level stage of the original KG-Retriever was omitted. The query and each triplet were vectorized using the same sentence embedding model, Sentence-BERT (`all-MiniLM-L6-v2`), and triplets were ranked based on cosine similarity. Among triplets with similarity above 0.30, up to 30 candidates were selected. Graph traversal was restricted to 1-hop neighbor exploration to utilize only factual information directly connected to the query. In the baseline KG without structural constraints, multi-hop expansion was considered to introduce unnecessary noise; therefore, the traversal depth

was limited to 1 hop. The selected triplets were inserted into the LLM (`GPT-4o-mini`) prompt to generate the final answer (Chen et al., 2025).

C Conditional Structure Modeling Details

Industrial standard documents commonly contain numerous conditional statements to specify manufacturing conditions, allowable ranges, and applicability criteria. Such conditions may be described in sentence form; however, they are also frequently presented in tables. In the case of tables, rows and columns define conditions, while the corresponding cell values represent consequences that apply when those conditions are satisfied. These tabular representations can therefore be interpreted as having the same **condition–consequence** structure. In this section, we illustrate how tables in industrial standard documents can be interpreted as conditional statements through an example, and describe in detail how sentence-level conditional structures are modeled based on this interpretation.

A single cell in a table can generally be decomposed into a condition part and a consequence part. For example, the cell highlighted in red in Figure 7(A) indicates that the specified regulation (permitted variation $= 3\,\text{mm}$) applies when both the column condition (thickness $\leq 5\,\text{mm} \;\vee\; \text{weight} \leq 70\,\text{kg/m}^2$) and the row condition ($200\,\text{mm} \leq \text{width} \leq 500\,\text{mm}$) are satisfied. This represents a prototypical conditional structure. However, conventional triplet extraction methods based on simple `subject-relation-object` representations typically decompose sentences into isolated fragments, such as "the width is greater than or equal to 200 mm" or "the thickness is less than or equal to 5 mm." As a result, they struggle to capture such composite condition–consequence structures.

To overcome this limitation, we newly define **Sentence-level Conditional Structure Modeling**, which systematically incorporates the conditional formats found in industrial standard documents into the triplet extraction process. In this modeling approach, conditional statements are decomposed into semantic units, and the logical relationships among conditions are explicitly represented in a structured manner, enabling more accurate encoding of condition–consequence relationships.

Concretely, the conditional statement in this example can be decomposed into the following semantic units:

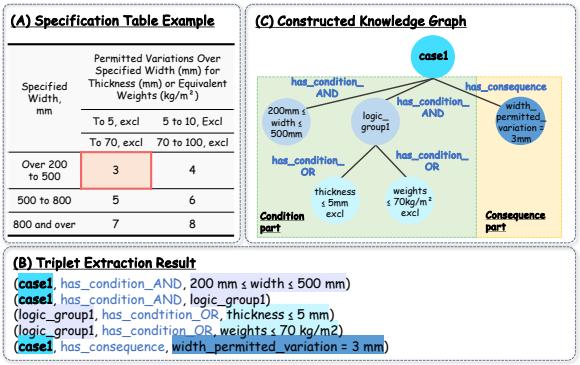

Figure 7: An example of triplet extraction and KG construction using sentence-level conditional structure modeling. (A) Table data from the A6/A6M document, (B) triplets extracted by modeling sentence-level conditional structures, and (C) an example of a KG constructed from the extracted triplets.

- C_1: $200\,\text{mm} < \text{width} \leq 500\,\text{mm}$
- C_2: $\text{thickness} \leq 5\,\text{mm}$
- C_3: $\text{equivalent weight} \leq 70\,\text{kg/m}^2$
- C_4: $\text{permitted variation in width} = 3\,\text{mm}$.

In this example, the thickness condition C_2 and the equivalent weight condition C_3 are connected by an `OR` relationship, forming a higher-level composite condition $C_2 \vee C_3$. The width condition C_1, on the other hand, must be satisfied jointly with this composite condition, resulting in an `AND` relationship. Accordingly, the cell can be represented by the following logical structure:

$$C_1 \;\wedge\; (C_2 \vee C_3) \rightarrow C_4. \tag{8}$$

In our pipeline, such a structure is represented as a single `case` node. When multiple logical operators (`AND`, `OR`) are mixed, we introduce intermediate `logic_group` nodes to explicitly structure the composition. Each `logic_group` node groups conditions combined by the same logical operator and is connected to the parent `case` node via a relation corresponding to that operator. By modeling conditional structures in this manner using a single table-cell example, table-based conditional statements can be interpreted in the same way as sentence-level conditional statements, while preserving the logical relationships among conditions in the extracted triplets. In this study, we construct a triplet extraction prompt that reflects this structure and employ `GPT-5-mini` (OpenAI, 2025) as the language model for triplet extraction. The triplets extracted from this example are subsequently used

as inputs for the Ontology-based KG construction stage.

D Synonym Dictionary Construction Details

In this appendix, we provide a detailed description of the methodology and implementation for synonym dictionary construction, focusing on the entity–relation synonym dictionary procedure introduced in Section 3.3.1 of the main text. To mitigate redundancy arising from identical or semantically similar concepts being expressed with different surface forms in the constructed KG, and to ensure overall representational consistency, we construct synonym dictionaries for both entities and relations. Rather than relying solely on surface-level text similarity, our approach aims to identify semantically equivalent concepts by additionally considering contextual usage patterns and structural roles within the graph.

First, for **entity dictionary** construction, we define a final embedding for each entity a by integrating textual semantics, local contextual information, and global structural information, as formalized in Eq. 9.

$$\mathbf{e}_{\text{final}}(a) = \big[\mathbf{e}_{\text{text}}(a) \;;\; \mathbf{e}_{\text{ctx}}(a) \;;\; \mathbf{e}_{\text{pos}}(a)\big], \quad (9)$$

where the textual semantic embedding $\mathbf{e}_{\text{text}}(a)$ is obtained using Sentence-BERT (`all-MiniLM-L6-v2`) (Reimers and Gurevych, 2019) to capture semantic similarity among entities. To reflect how an entity is actually used within the KG, a local contextual embedding $\mathbf{e}_{\text{ctx}}(a)$ is computed using Personalized PageRank (PPR) (Gasteiger et al., 2022), which encodes contextual relatedness and importance with respect to neighboring nodes. In addition, a global structural embedding $\mathbf{e}_{\text{pos}}(a)$ is derived via spectral embedding based on the eigenvectors of the graph Laplacian (Belkin and Niyogi, 2001), capturing the entity's overall structural position and role in the graph. By combining these two components, the resulting embedding jointly accounts for both local context and global structure.

The integrated entity embeddings $\mathbf{e}_{\text{final}}(a)$ are then clustered using `HDBSCAN` (McInnes et al., 2017) to automatically form synonym sets of entities that are semantically and structurally similar. Entity and relation representations in the KG exhibit uneven domain-specific frequency and distribution, and the size of synonym sets is difficult to predefine. `HDBSCAN` is therefore adopted for its ability to identify semantically dense clusters without requiring the number of clusters to be specified in advance, while effectively separating noisy representations from meaningful clusters.

For the **relation dictionary**, in addition to an embedding composition similar to that of entities, the functional characteristics of relations are explicitly incorporated. Specifically, for each relation r, we augment the textual semantic, contextual, and structural embeddings with a Selectional Preference embedding $\mathbf{e}_{\text{prefer}}(r)$, which captures the distribution of subject–object entity types that the relation predominantly connects:

$$\mathbf{e}_{\text{final}}(r) = \big[\mathbf{e}_{\text{text}}(r) \;;\; \mathbf{e}_{\text{ctx}}(r) \;;\; \mathbf{e}_{\text{pos}}(r) \;;\; \mathbf{e}_{\text{prefer}}(r)\big], \quad (10)$$

where $\mathbf{e}_{\text{prefer}}(r)$ is designed to capture not only semantic similarity but also functional similarity by modeling the type distribution of entity pairs (u, v) in which relation r frequently occurs as a probability distribution $p(t \mid u, v)$. Finally, the relation embeddings are also clustered using `HDBSCAN`, and within each cluster, the relation closest to the cluster centroid is selected as the canonical predicate to standardize relation representations. The resulting entity–relation synonym dictionary is subsequently used in KG normalization and refinement processes, contributing to the reduction of redundant expressions and the improvement of overall structural consistency of the graph.

E Additional Results

E.1 Performance on Toxic Clause Detection Task

The preceding QA experiments demonstrated that the proposed approach can accurately retrieve and reason over factual information in standards documents. However, in practical industrial document usage scenarios, it is crucial not only to generate correct answers but also to *precisely identify subtle modifications in document content or numerical manipulations*. In particular, in contracts and industrial standards documents, even minor changes in numerical values, conditions, or wording can directly lead to regulatory violations or toxic clauses. Accordingly, to verify **how robustly the proposed method can detect such rule-based semantic modifications**, we additionally conduct a Toxic Clause Detection Task.

Table 8: Performance on API_2W toxic clause detection task (F1, Accuracy, Recall).

Type	Method	F1	Acc	Recall
LLM-only	GPT-4o-mini	0.558	0.667	0.438
	gemini-2.0-flash	0.000	0.520	0.000
	DeepSeek-v3.1	0.591	0.685	0.471
LLM w/full doc	GPT-4o-mini w/full doc	0.812	0.833	0.752
	gemini-2.0-flash	0.527	0.667	0.364
	DeepSeek-v3.1	0.850	0.857	0.890
Text-only RAG	BM25+Qwen1.5-14B	0.870	0.869	0.895
	BM25+Mistral-7B	0.837	0.826	0.917
	BM25+GPT-4o-mini	0.878	0.879	0.886
	BM25+gemini-2.0-flash	0.887	0.888	0.895
	Dense+Qwen1.5-14B	0.871	0.872	0.893
	Dense+Mistral-7B	0.834	0.825	0.809
	Dense+GPT-4o-mini	0.851	0.861	0.826
	Dense+gemini-2.0-flash	0.851	0.864	0.818
KG-RAG	MindMap (Wen et al., 2024)	0.819	0.812	0.880
	KG Retriever (Chen et al., 2024)	0.900	0.905	0.893
Ontology-based KG-RAG	Ontology-based MindMap	0.828	0.832	0.883
	Ontology-based KG-Retriever	**0.910**	**0.913**	**0.926**

The toxic clause detection task is generally defined as the problem of identifying clauses in contracts that may be unfavorable or cause legal issues. In the context of contracts, statements that are inconsistent with the rules or conditions specified in the document can also be regarded as toxic clauses. In this study, we define the toxic clause detection task as a classification problem in which both an original sentence from the document and a modified sentence—with partial alterations in numerical values or key terms—are presented, and the model is required to determine whether the sentence complies with the rules and constraints of the original document.

For the experiments, we construct a dataset consisting of 252 classification instances using the API_2W document, and representative examples of the dataset are provided in Appendix Table 10.

As shown in Table 8, the Ontology-based KG-Retriever achieves the best performance across all evaluation metrics. In addition, the Ontology-based MindMap consistently outperforms the original MindMap without ontology integration. These results indicate that the proposed approach goes beyond simple retrieval and more accurately preserves rules that involve condition–consequence structures and numerical constraints.

E.2 Performance under Different k-Hop Neighborhoods

Figure 8 presents the experimental results with varying hop sizes for the two proposed methods that incorporate the ontology structure. Ontology-based MindMap achieves its best performance at 3-hop, while Ontology-based KG-Retriever attains the highest performance at 2-hop. These results suggest that the optimal hop size differs due to the distinct structural characteristics of the two exploration strategies. Specifically, Ontology-based KG-Retriever explores multidimensionally related nodes at both the ontology and KG levels, whereas Ontology-based MindMap follows a relatively linear exploration process. As a result of these structural differences, the optimal number of hops varies between the two methods. Accordingly, in this experiment, we apply the optimal hop size separately for each methodology.

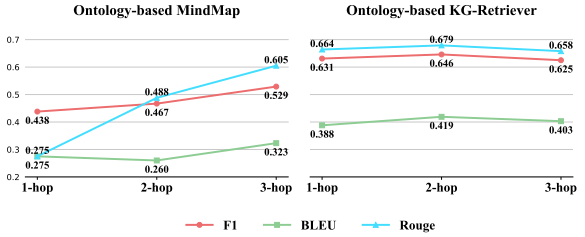

Figure 8: Performance across different k-hop settings using the API_2W rule data. Across all evaluation metrics, the Ontology-based MindMap achieves its best performance at 3-hop, whereas the Ontology-based KG-Retriever performs best at 2-hop.

F Dataset Details

Table 9: Representative Question Examples from IndusSpec-QA. This table presents representative questions extracted from the IndusSpec-QA dataset. Each question type (rule, multi-hop, table) reflects the structural characteristics and query patterns of industrial standard documents.

Type	Question	Answer
rule	What does CE stand for in the context of heat analysis?	carbon equivalent
multi-hop	What processes did TMCP evolve from, as discussed in the bibliographical reference regarding fine-grained steel?	controlled rolling processes
table	What is the maximum percentage of Nickel allowed in the chemical requirements for Grade 60?	1

Table 10: Representative Toxic Clause Question Examples. This table presents example questions from the toxic clause dataset. The questions reflect potentially risky clauses related to industrial standards, and the answers are derived based on the corresponding specifications. In particular, answers labeled as **No** indicate toxic clauses that conflict with the actual standards.

Question	Answer
Does the regulation allow specimens to be taken without being adjacent to tensile test coupons?	No
If a specimen fails during the drop-weight test, must two additional specimens from each plate be retested?	Yes
Are blisters considered acceptable surface imperfections even if they do not reduce the thickness below minimum?	No

G Synonym Dictionary Visualization

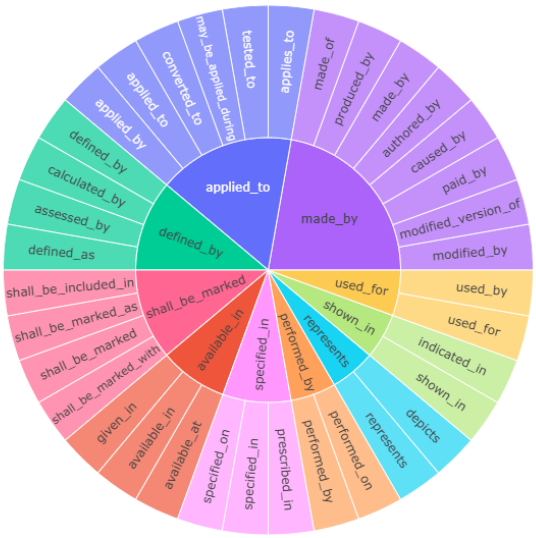

Figure 9: Relation synonym dictionary Visualization. This figure illustrates a subset of the hierarchical relationships between canonical predicates and their corresponding paraphrases using a sunburst diagram. Each canonical predicate encompasses a set of outer-level synonymous expressions, visually grouping semantically similar relation expressions into a single cluster.

H Ontology-based KG Visualization

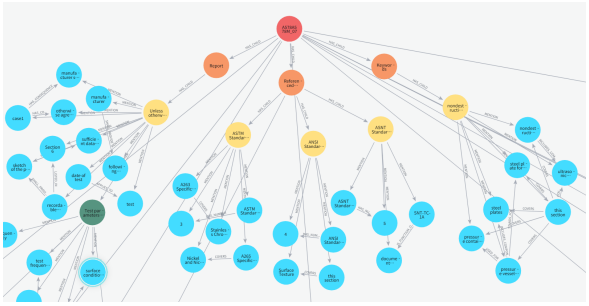

Figure 10: Visualization of an Ontology-based KG using `Neo4j`. Node colors indicate the hierarchical structure of the document: red nodes represent the document name, orange nodes denote sections, yellow nodes indicate subsections, and green nodes correspond to sub-subsections. Blue nodes represent concrete entities mentioned in the document. Edges, such as `[has_child]` and `[mention]`, explicitly encode relationships between the document structure and the extracted knowledge.

I Prompt Templates for Conditional-based triplet Extraction

Table-based triplet Extraction Prompt

Instruction
You are an expert in constructing knowledge graphs. Your task is to extract knowledge triplets (*subject–relation–object*) from the given tabular content. The input content may include:

- Tabular content (structured tables with conditions and permitted variations)
- Table titles
- Table captions appearing outside the table

Output Format

- Use the following structure for each triplet:
  `<triplet> {subject} <subj> {object} <obj> {relation_name}`
- `<subj>` and `<obj>` are explicit boundary markers indicating the end of the subject and object spans, respectively. They are not natural language tokens but structural delimiters.

Guidelines (Excerpt)

1. Extract only semantically meaningful and domain-relevant relations.
2. For enumerations or numerical constraints:
   - Capture requirement constraints (e.g., `thickness < 3 mm`).
   - Normalize all numerical values into SI units (e.g., millimeters, Celsius), even if the input uses mixed units.
3. For tabular data with conditions:
   - If the input starts with the `|` character, treat it strictly as table data: do not apply any plain text rules, and only use table-specific rules.
   - When processing tables, the ONLY allowed relations are:
     - If there is only one condition → `has_condition`
     - If there are multiple conditions → `has_condition_AND`, `has_condition_OR`
     - If there is only one consequence → `has_consequence`
     - If there are multiple consequences → `has_consequence_AND`, `has_consequence_OR`
     - Do not use any other relation names.
   - Each cell in a table must be represented as a rule node (`caseX`).
     - For every non-empty cell, create a new case node (`case1`, `case2`, ...).
     - Each case node must include:
       * all row labels as conditions,
       * all column labels as conditions,
       * and the cell value as the consequence.
   - When logical expressions contain both AND and OR, you must introduce a `logic_groupX` node.
     - Inside a `logic_groupX`, you must use ONLY one type of edge: either `has_condition_AND`, `has_condition_OR`, `has_consequence_AND`, or `has_consequence_OR`.
     - The `logic_groupX` itself must then connect to the case node (`caseX`) using the appropriate relation among the same four types.
     - Always preserve operator precedence by grouping first.
       Example: `(A OR B) AND C → A and B` are both connected to a `logic_groupX` with `has_condition_OR`; then this `logic_groupX` and C are connected to `caseX` with `has_condition_AND`.
     - Always include the column/row name together with the cell value when forming a condition or consequence.
4. Always include the corresponding row or column name when forming a condition or consequence.
5. Do not directly connect conditions to consequences; always connect them through a `caseX` or `logic_groupX`.
6. Keep relation names consistent and in `snake_case`.
7. Do not generate vague or trivial relations.

Examples (Excerpt)

- Example 1 - Unit Conversion
  Title: Dimensions -> Thickness
  Input: The plates shall have a minimum thickness of 3/8 in. [10 mm].
  Output: <triplet> plates <subj> 10 mm <obj> has_minimum_thickness

- Example 2 - Table (Simple Condition)
  Title: Table -> Permitted Variation
  Input (excerpt):| Thickness Range | Thickness Difference | Minimum Number of Tension Tests |
  | Under 10 mm | <= 2 mm | Two tests per heat |

  Output: <triplet> case1 <subj> thickness < 10 mm <obj> has_condition_AND
  <triplet> case1 <subj> thickness_difference <= 2 mm <obj> has_condition_AND
  <triplet> case1 <subj> two_tests_per_heat <obj> has_consequence

Now extract triplets from the following section:
Title: `{title}`
Input: `{text}`
Output:

Text-based triplet Extraction Prompt

Instruction
You are an expert in constructing knowledge graphs. Your task is to extract knowledge triplets (*subject–relation–object*) from the given document section. The input content may include:

- Plain text (technical specifications, requirements, descriptions)
- Mathematical expressions (formulas, equations, definitions)
- Enumerated content (lists, measurement conversions)
- Tabular content (structured tables with conditions and permitted variations)

Output Format

- Use the following structure for each triplet:
  `<triplet> {subject} <subj> {object} <obj> {relation_name}`
- <subj> and <obj> are explicit boundary markers indicating the end of the subject and object spans, respectively. They are not natural language tokens but structural delimiters.

Guidelines (Excerpt)

1. Extract only semantically meaningful and domain-relevant relations.
2. For plain text:
   - Coverage or scope → (covers, applies_to)
   - Process or production → (produced_by, used_for)
   - Definitions → (stands_for, defined_as)
   - Specifications → (has_condition_AND, has_condition_OR, has_condition, has_consequence, has_consequence_OR, has_consequence_AND)
3. For mathematical formulas:
   - Variable definitions → (X <subj> Y <obj> stands_for)
   - Formula representation → (X <subj> formula <obj> calculated_by)
   - Component terms → (X <subj> Y <obj> includes_component)
   - Operations → (X <subj> Y <obj> divided_by / multiplied_by / added_to / subtracted_by)
4. For enumerations or units:
   - Capture requirement constraints (e.g., thickness ≥ 3 mm)
   - Always normalize numerical values into SI units (millimeters, Celsius, etc.) in the output triplets, even if the input uses mixed units.
5. When text starts with a numbering pattern (e.g., "1.", "2.1", "3.2.4"), do not extract that numeric label as a triplet. Only process the semantic content that follows.
6. Keep relation naming consistent (snake_case).
7. Use technical terms exactly as written in the input text.
8. Do not generate vague or trivial relations.
9. Do NOT output placeholder tokens such as "<relation>" or generic words like "relation". Always replace with an actual relation name (e.g., covers, produced_by, equivalent_to).

Examples (Excerpt)

- Example 1 - Unit Conversion
  Title: Dimensions -> Thickness
  Input: The plates shall have a minimum thickness of 3/8 in. [10 mm].
  Output: <triplet> plates <subj> 10 mm <obj> has_minimum_thickness

- Example 2 - Formula
  Title: Supplementary Requirements -> Maximum Carbon Equivalent for Weldability
  Input: CE = C + Mn/6 + (Cr + Mo + V) /5 + (Ni + Cu) / 51
  Output: <triplet> CE <subj> carbon equivalent <obj> stands_for
  <triplet> CE <subj> C + Mn/6 + (Cr + Mo + V) /5 + (Ni + Cu) / 51 <obj> calculated_by
  <triplet> carbon equivalent <subj> C <obj> includes_component
  <triplet> carbon equivalent <subj> Mn <obj> includes_component

- Example 3 - Conditional Requirement
  Title: Heat Treatment
  Input: If the plate thickness exceeds 2 in. [50 mm], heat treatment shall be required.
  Output: <triplet> case1 <subj> plate thickness ≥ 50 mm <obj> has_condition

<triplet> case1 <subj> heat treatment <obj> has_consequence

Now extract triplets from the following section:
Title: `{title}`
Input: `{text}`
Output: