

Ontology-Based Knowledge Graphs for Industrial Standards with Scope-Aware Logical Rule Structuring

Jiin Park^{†1} Hyuna Jeon^{†1} Yoonseo Lee² Misuk Kim^{*1,2}

¹ Department of Artificial Intelligence, Hanyang University, Seoul, Republic of Korea

² Department of Data Science, Hanyang University, Seoul, Republic of Korea
{jiinpark, hyunajeon, yoonseolee, misukkim}@hanyang.ac.kr

Abstract

Industrial standard documents combine hierarchical section structures, table-based rules, and complex conditional statements; as a result, the same expression can refer to different rules depending on where it appears in a document. However, existing text-based approaches and conventional knowledge graph (KG) construction methods do not jointly preserve these structures, making them brittle on complex rule-based queries. In this work, we propose an ontology-based KG construction method that formalizes section structure as an ontology and integrates it with rule-aware triplets extracted from text and tables. To evaluate our approach, we construct IndusSpec-QA, a QA dataset based on three industrial standard documents, and compare the proposed method against a range of baselines. Experimental results show that the proposed Ontology-based KG-RAG outperforms existing methods, with particularly large gains on table-based rule questions and multi-condition reasoning. These results demonstrate the importance of knowledge representations that jointly preserve section and rule structures for QA over industrial standard documents. The code is publicly available at: https://github.com/Ontology-KG/ontology_based_kg_paper.

1 Introduction

Industrial standard documents are widely used in various domains such as shipbuilding, construction, and manufacturing as the basis for design review, technical comparison, and contract specifications. Unlike general-domain text, these documents are organized such that section titles define semantic concepts, while subsections, conditions, tables, and footnotes progressively refine those concepts. As a result, identical terms may correspond to different rules depending on their section context, and

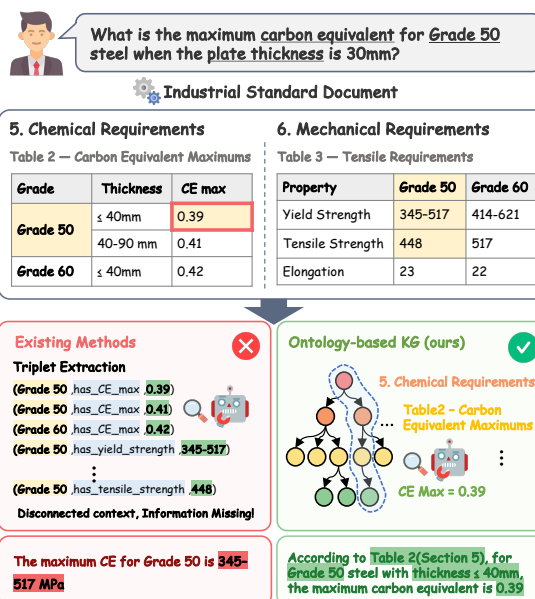


Figure 1: Comparison of Conventional KG and Ontology-based KG. The same term can denote different rules depending on section and table context. Conventional KG loses this context, while our ontology-based KG preserves section scope and condition–consequence structure for accurate inference.

a single rule often requires jointly interpreting distributed conditions and tabular constraints. Therefore, the meaning of a rule is determined not at the sentence level, but within the interplay between section-level conceptual structure and condition compositions (Kabzhan et al., 2025).

For example, Figure 1 shows how the same material grade, *Grade 50*, can be interpreted differently depending on context: under *Chemical Requirements*, it is associated with rules for maximum carbon equivalent conditioned on plate thickness, whereas under *Mechanical Requirements*, it defines yield strength and tensile strength specifications. This indicates that identical expressions can correspond to different attributes and rules depending on their associated section and table context.

[†]Equal contribution. ^{*}Corresponding author.

Recently, large language models (LLMs) and retrieval-augmented generation (RAG) have been actively applied to automate question answering and document understanding over industrial documents (Boecking et al., 2025). However, existing LLM-based and text-based RAG approaches typically process documents as linear text chunks, failing to explicitly model structural relationships such as section structure. Consequently, the contextual role of section titles in defining rule scope and inter-section dependencies is not sufficiently captured during retrieval and reasoning.

In addition, conventional triplet-based knowledge graph (KG) construction methods extract subject-relation-object triples independently at the sentence level (Zhang and Soh, 2024), making it difficult to preserve table row/column constraints and complex conditional rules as unified rule units. These limitations often lead to critical errors in downstream tasks, significantly reducing the reliability of RAG-based systems (Tufek, 2023).

To address these challenges, we model section titles as *section concepts* that define both semantic meaning and scope, and organize them into a *scope path* from the root section to the current section. We then propose an Ontology-based KG construction framework that connects extracted rules from text and tables to this structural representation.

We apply the proposed method to three real-world industrial standards (ASTM A578/A578M, API_2W, ASTM A6/A6M) to construct Ontology-based KGs. To evaluate its effectiveness, we build a QA dataset IndusSpec-QA covering Rule, Multi-hop, and Table reasoning types. Furthermore, we propose Ontology-based KG-RAG and compare it against multiple RAG-based and LLM baselines. Experimental results demonstrate that performance gains stem not from longer context or generic graph construction, but from jointly preserving section structure and condition-consequence relationships within the Ontology-based KG.

Our contributions are summarized as follows:

- We formalize the section structure of industrial standard documents as an ontology that defines semantic concepts and scope, and propose an Ontology-based KG construction framework.
- We introduce a rule-aware triplet extraction method using case and logic_group nodes to consistently represent complex conditional rules from text and tables.

- We construct an industrial QA dataset IndusSpec-QA and demonstrate the effectiveness of the proposed Ontology-based KG through comprehensive quantitative and qualitative evaluations.

2 Related Works

2.1 Ontology

Ontology is defined as a framework for structurally representing the core concepts of a specific domain and the relationships among them (Miller, 1995; Ashburner et al., 2000). Based on this view, ontology learning has developed around concept discovery and relation extraction, with recent work exploring LLM-based schema-free ontology construction and ontology expansion (Funk et al., 2023; Lo et al., 2024). Document ontologies such as DoCO and DITA-RDF provide vocabularies for representing document components, DITA structures, and metadata in RDF/OWL (Constantin et al., 2016; Maudry). Our work differs from these approaches not by introducing a new general-purpose document vocabulary, but by using document structure as an explicit semantic scope for rule applicability and retrieval. Specifically, we represent section titles as *section concepts* and their structural paths as *scope paths*, and connect rules extracted from text and tables to these scopes while preserving condition-consequence and logical structures. Similarly, existing ontology learning studies mainly focus on discovering concepts and semantic relations between entities, whereas our framework explicitly links document-level section structure with rule-level semantics.

2.2 Knowledge Graph

Knowledge graphs (KGs) represent entities and relations in the form of triplets, thereby supporting structured knowledge storage and reasoning. Early approaches mainly relied on rule-based and statistical methods (Etzioni et al., 2004; Yates et al., 2007), while later advances in neural relation extraction substantially improved extraction performance (Wei et al., 2020). Recently, LLM-based triplet extraction and prompt-based KG construction have also been actively applied across various domains (Huguet Cabot and Navigli, 2021; Zhang and Soh, 2024; Hu et al., 2024; Wang et al., 2025). Recent graph-based RAG methods, such as GraphRAG, LightRAG, and HippoRAG2, construct graph or memory structures from documents

to improve retrieval and reasoning (Edge et al., 2025; Guo et al., 2024; Gutiérrez et al., 2025). However, existing approaches mainly focus on relation representations for simple sentences, making it difficult to sufficiently represent complex rule structures in which multiple conditions and logical dependencies are combined in compound sentences. In contrast, our work jointly models ontology-based structure and condition–consequence relations, enabling the structural representation of complex rules in industrial standard documents.

3 Method

This section describes the overall procedure for converting industrial standard documents into an Ontology-based KG (See Figure 2). The core idea of the proposed method is to identify the conceptual scope defined by document section titles and connect rules extracted from text and tables to that structure. We first define the necessary notation and basic concepts in Section 3.1. We then describe the overall process in three stages: (i) section-grounded ontology modeling, (ii) rule-aware triplet extraction, and (iii) ontology-based KG construction.

3.1 Preliminaries

Industrial standard documents are organized into multiple levels of sections according to explicit numbering systems and title structures. In this study, we regard the concept defined by each section title as a *section concept*. We also define the path from the root section to the section containing the current content unit as the *scope path* p_i :

$$p_i = s_1 \rightarrow s_2 \rightarrow \cdots \rightarrow s_k, \quad (1)$$

where $s_1, \dots, s_k$ are the *section concepts* sequentially connected from the root section to the section containing the current content unit, and p_i indicates the section context in which the content should be interpreted. Each content unit of document D (text, table, footnote) is represented as a knowledge unit U_i together with its corresponding *scope path*:

$$D = \{U_1, U_2, \dots, U_N\}, \quad U_i = (p_i, x_i), \quad (2)$$

where p_i is the *scope path* of the i -th content unit, and x_i denotes the actual content of that unit. This representation keeps document content and its structural context from being separated.

In addition, we represent rules in text and tables as condition–consequence structures. Each rule is

represented as a case node, and condition combinations involving both AND and OR are preserved using a `logic_group` node. Let T_i denote the set of triplets extracted from each knowledge unit U_i ; the full triplet set of the document is defined as:

$$T(D) = \bigcup_{i=1}^N T_i \quad (3)$$

In the following steps, each T_i is connected to its corresponding p_i to construct the final Ontology-based KG.

3.2 Section-grounded Ontology Modeling

In this stage, we construct the *section concepts* and *scope paths* defined in Section 3.1 from the document. We identify *section concepts* using the numbering system and title patterns in the document, and generate *scope paths* along parent–child relations.

Section-as-Concept Parsing Industrial standard documents are organized into multiple levels of sections through explicit numbering systems and title structures. Just as a wiki document progressively refines sub-concepts under a higher-level concept, industrial standard documents are also structured such that lower-level sections are conceptually subordinate to higher-level sections. We model the concept defined by each section title as a *section concept*, and regard lower-level sections as sub-concepts that refine it.

For example, in Figure 2(a), the structure in which 7.4 *Alternative Toughness Tests* and 7.4.2 *Different CVN Values* are connected under 7 *Notch Toughness Testing* is interpreted not as a simple table of contents, but as a semantic structure in which a higher-level concept is progressively refined. We represent this path from the root section to the current section as the *scope path* p_i (Figure 2(b)).

To this end, we parse the section structure in a rule-based manner using the document numbering system and title patterns, and automatically identify each section as a *section concept* in the ontology. This process is designed to extend in the same way to deeper subsection structures.

Scope-grounded Knowledge Unit Construction

As defined in Section 3.1, each knowledge unit contains both a content unit and its corresponding *scope path*. In this stage, we combine the actual content x_i under each section (text, table, footnote) with its *scope path* p_i to construct a knowledge unit

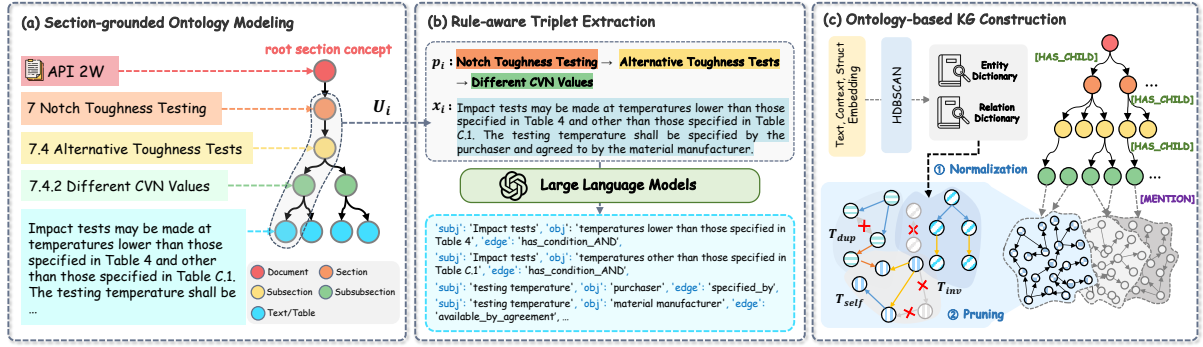


Figure 2: Ontology-based KG Pipeline. (a) Section-grounded Ontology Modeling organizes the document into a section-based ontology structure. (b) Rule-aware Triplet Extraction uses an LLM to extract condition–consequence-based triplets from text and tables. (c) Ontology-based KG Construction integrates the extracted triplets with the ontology structure and refines the resulting graph through synonym-dictionary-based normalization and pruning.

U_i . This preserves the content unit and its structural context together, preventing them from being separated within the document representation. For example, in Figure 2 (a), a sentence under 7.4.2 *Different CVN Values* is stored as a knowledge unit together with its *scope path*. As a result, even identical terms or numerical values are treated as different rule units when they belong to different *scope paths*, and this contextual scope is preserved through the subsequent triplet extraction stage.

3.3 Rule-aware Triplet Extraction

In this stage, we interpret the content x_i of each knowledge unit $U_i = (p_i, x_i)$ as a condition–consequence rule and convert it into rule-aware triplets.

Textual Condition–Consequence Modeling

Textual rules in industrial standard documents typically describe requirements that apply when specific conditions are satisfied. We represent each such rule as a case node and connect the corresponding conditions and consequences to it. When a condition contains both AND and OR structures, such as $A \wedge (B \vee C)$, we introduce a *logic_group* node to preserve the order of logical composition. In this sense, *logic_group* serves the structural role of parentheses in a logical expression. Algorithm 1 shows how the content x_i of a knowledge unit is decomposed into a condition structure C and a consequence structure R , and recursively connected to a case node. Figure 3 illustrates how the condition–consequence structure extracted through this process is represented in the ontology around case and *logic_group* nodes.

Algorithm 1 Triplet Extraction from Conditional Rules

```

1: Input: Content  $x_i$  from a knowledge unit  $U_i$ 
2: Output: Triplet set  $T$ 
3: Initialize an empty triplet set  $T$ 
4: Create a case node  $c$ 
5: Parse  $x_i$  into a condition structure  $C$  and a consequence structure  $R$ 
6:  $T \leftarrow T \cup MCT(c, \text{has\_condition}, C)$ 
7:  $T \leftarrow T \cup MCT(c, \text{has\_consequence}, R)$ 
8:  $\triangleright MCT$  denotes ModelConditionalTriplets
9: return  $T$ 

10: Procedure  $MCT(p, rel, S)$ 
11: Initialize an empty triplet set  $T_S$ 
12: if  $S$  contains only one condition or consequence then
13:    $T_S \leftarrow T_S \cup \langle p, rel, S \rangle$ 
14: else if all items in  $S$  are combined using the same logical operator then
15:    $op \leftarrow$  shared operator of  $S$ 
16:    $T_S \leftarrow T_S \cup \langle p, rel_{op}, S \rangle$ 
17: else
18:   Separate  $S$  into grouped parts and remaining parts
19:    $op \leftarrow$  logical operator connecting these parts
20:   for each grouped part  $G$  do
21:      $\triangleright$  Introduce a logic_group node for grouped substructures
22:     Create a logic_group node  $g$ 
23:      $T_S \leftarrow T_S \cup \langle p, rel_{op}, g \rangle$ 
24:      $T_S \leftarrow T_S \cup MCT(g, rel, G)$ 
25:   end for
26:   for each remaining part  $e$  do
27:      $T_S \leftarrow T_S \cup \langle p, rel_{op}, e \rangle$ 
28:   end for
29: end if
30: return  $T_S$ 

```

LLM-based Triplet Extraction with Rule Schema

We formulate the rule representation defined above as a prompt $\mathcal{P}$ and provide it to the LLM. The prompt includes the condition–consequence structure, rules for generating case and *logic_group* nodes, relation naming rules, unit normalization rules, and few-shot examples. Thus, the LLM interprets the text and tables in industrial standard documents as rule units and extracts consistent triplets. The prompt used in this study is provided in Appendix J.

For each knowledge unit $U_i = (p_i, x_i)$, we extract a triplet set $T_i = f_{\text{ext}}(\mathcal{P}, U_i)$ and obtain the full triplet set extracted from the document as in

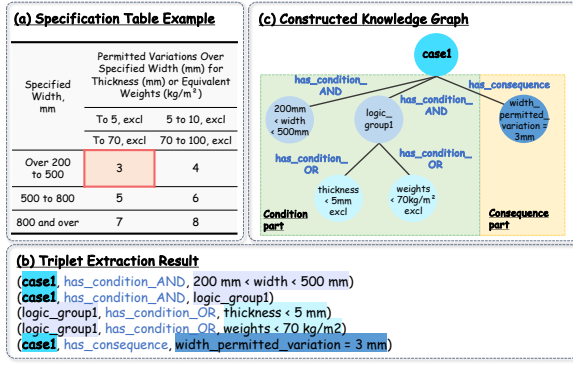


Figure 3: Table-to-rule transformation example. (a) In a table from an industrial standard document, a cell is interpreted as a rule that applies under the corresponding row and column conditions. (b) The selected cell is converted into condition–consequence triplets using case and logic_group nodes. (c) The extracted triplets are organized into a graph that preserves both the condition part and the consequence part.

Eq. 3.

This set is used in the next stage to construct the final Ontology-based KG by connecting each triplet to its corresponding *scope path*.

3.4 Ontology-based KG Construction

In this stage, we construct the final Ontology-based KG by connecting each knowledge unit’s *scope path* p_i with the triplet set T_i extracted from it.

Graph Integration Schema The final graph consists of three types of nodes: (i) *section concepts*, which represent the section structure, (ii) case and logic_group nodes, which represent condition–consequence structures, and (iii) entity/value nodes, which represent domain knowledge such as material names, test conditions, technical terms, and numerical values. Hierarchical relations between *section concepts* are connected by the [has_child] relation, while each rule and entity/value node is connected to the last section in the *scope path* of its corresponding knowledge unit through the [mention] relation. This allows identical terms or numerical values to be distinguished as different rules when they appear under different *section concepts*.

KG Normalization and Pruning After graph construction, we perform normalization and pruning to reduce representational redundancy and improve structural consistency. Normalization integrates semantically equivalent entities and relations into canonical forms based on a synonym dictio-

nary, considering string similarity, contextual information, graph structure, and the subject–object type distribution of relations. Pruning conservatively removes only low-information structures, such as self-loops, duplicate triplets, and clearly redundant reciprocal edges. This stage does not generate new rules; rather, it is a post-processing step for improving the representational consistency and structural stability of the constructed Ontology-based KG. Detailed procedures are provided in Appendix E.

4 Experiments

The goal of our experiments is to evaluate how effectively the proposed Ontology-based KG captures the section structure and rule-based knowledge of industrial standard documents. To this end, we construct Ontology-based KGs and the IndusSpec-QA dataset from three real-world industrial standard documents, and evaluate Ontology-based KG-RAG against various RAG and LLM baselines.

4.1 Experimental Setup

Datasets To evaluate the effectiveness of the proposed method across different document structures and rule types, we use three industrial standard documents. Each document is a real-world standard used in industrial practice and has distinct characteristics in terms of length, table structure, and conditional complexity.

- **ASTM A578/A578M¹** is a short and rule-centric document, containing many sentence-based rules, which makes it suitable for evaluating basic rule QA.
- **API 2W²** contains many table-based rules, making it challenging for table reasoning.
- **ASTM A6/A6M³** is the largest document and includes exception clauses, multi-hop conditions, and large tables, making it suitable for evaluating complex reasoning.

Based on these three documents, we apply the proposed method to construct Ontology-based KGs as shown in Table 1. To validate the effectiveness of the KG, we also construct a QA dataset, IndusSpec-QA (See Table 2). Each question is designed based on the knowledge unit $U_i = (p_i, x_i)$

¹<https://standards.globalspec.com/std/4055316/astm-a578-a578m-17>

²<https://standards.globalspec.com/std/921032/api-spec-2w>

³<https://store.astm.org/a0006-a0006m-14.html>

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-based, multi-hop, and table-based—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

defined in Section 3.1, and its evidence scope is determined by jointly considering the *scope path* p_i and content x_i . Question drafts are generated using an LLM and then finalized through domain expert validation. Details of the data generation and validation process are provided in Appendix A. The final dataset consists of three types: Rule refers to questions based on a single condition or sentence, Multi-hop refers to questions that require combining multiple sections or conditions, and Table refers to questions that require jointly interpreting row/column conditions and cell values.

Baselines The purpose of our baseline comparison is not to compete with the latest RAG methods in terms of raw performance, but to verify whether the proposed Ontology-based KG is effective as a knowledge representation for industrial standard documents. To this end, we organize the baselines according to the form of knowledge they use.

(1) LLM-only generates an answer using only the query, without using any external document or retrieval result. (2) LLM w/full doc provides the full document together with the query in the LLM context, allowing the model to directly refer to the document content without retrieval. (3) Text-only RAG is a text chunk-based retrieval method, for which we use BM25 (Robertson and Zaragoza, 2009) and Ada-002 (OpenAI, 2022). (4) Document-to-Graph RAG refers to graph-based RAG methods that convert documents into graph structures, for which we use GraphRAG (Edge et al., 2025), LightRAG (Guo et al., 2024), and HippoRAG2 (Gutiérrez et al., 2025). (5) KG-based RAG uses a general KG constructed without ontology, where the KG is built only with the pro-

posed rule-aware triplet extraction. For this setting, we use MindMap (Wen et al., 2024) and KG-Retriever (Chen et al., 2025). Finally, (6) Ontology-based KG-RAG uses the Ontology-based KG proposed in this study, and is implemented by extending MindMap and KG-Retriever to fit the ontology structure. In the main comparison, we use GPT-5-mini⁴ for ontology construction, KG construction, and response generation, and report additional backbone results as supplementary comparisons. Detailed implementations of each baseline are summarized in Appendix C.

Evaluation Metrics To evaluate the quality of model responses, we used F1, BERTScore, BLEU, and ROUGE. F1 measures token-level overlap between the predicted answer and the gold answer, BERTScore measures semantic similarity, while BLEU and ROUGE evaluate n-gram and phrase-level textual consistency. Detailed descriptions are provided in Appendix B.

4.2 Main Results

Table 3 presents the results comparing the proposed Ontology-based KG-RAG with various LLM and RAG baselines on three industrial standard documents. The main observations from this experiment are as follows.

Ontology-based KG-RAG achieves the highest performance on all datasets. On A578/A578M, Ontology-based KG-Retriever achieves the best performance with an F1 score of 0.722, while on API_2W and A6/A6M, Ontology-based MindMap achieves the highest performance with F1 scores of 0.564 and 0.543, respectively. This shows that the proposed method can be consistently applied not only to rule-centric documents, but also to documents with many tables and exception clauses.

LLM w/full doc shows limitations. The GPT-5-mini-based LLM w/full doc achieves an average F1 score of 0.340, which is substantially lower than Ontology-based KG-RAG (0.557). This indicates that long-context input alone is insufficient to reliably distinguish section-specific scopes and condition combinations. The performance degradation is particularly noticeable on A6/A6M, the longest document.

Document-to-Graph RAG shows that converting documents into graphs alone is insufficient.

⁴gpt-5-mini-2025-08-07.

Table 3: Performance comparison across datasets on three industrial standard documents. Scores are weighted averages over Rule, Table, 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-5-mini	0.000	0.796	0.000	0.003	0.000	0.794	0.000	0.001	0.000	0.789	0.000	0.000
	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-5-mini	0.372	0.880	0.183	0.438	0.376	0.884	0.159	0.494	0.273	0.866	0.123	0.362
	gemini-2.0-flash	0.372	0.880	0.183	0.439	0.215	0.840	0.105	0.240	0.095	0.808	0.044	0.108
	DeepSeek-v3.1	0.310	0.870	0.161	0.376	0.314	0.867	0.131	0.384	0.149	0.824	0.082	0.178
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-5-mini	0.348	0.875	0.174	0.411	0.290	0.860	0.128	0.352	0.170	0.832	0.075	0.198
	BM25+gemini-2.0-flash	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-5-mini	0.316	0.869	0.154	0.376	0.365	0.878	0.155	0.459	0.161	0.829	0.074	0.200
	Dense+gemini-2.0-flash	0.309	0.870	0.167	0.359	0.348	0.875	0.164	0.471	0.167	0.825	0.077	0.203
Document-to-Graph RAG	GraphRAG (Edge et al., 2025)	0.410	0.850	0.188	0.413	0.416	0.841	0.171	0.415	0.252	0.826	0.121	0.257
	LightRAG (Guo et al., 2024)	0.288	0.773	0.112	0.293	0.340	0.867	0.128	0.339	0.163	0.679	0.056	0.169
	HippoRAG2 (Gutiérrez et al., 2025)	0.280	0.814	0.115	0.421	0.203	0.789	0.077	0.359	0.195	0.788	0.054	0.390
KG-based RAG	MindMap (Wen et al., 2024)	0.601	0.924	0.362	0.607	0.413	0.892	0.278	0.447	0.339	0.873	0.182	0.352
	KG-Retriever (Chen et al., 2025)	0.706	0.945	0.449	0.711	0.435	0.891	0.290	0.449	0.271	0.848	0.171	0.275
Ontology-based KG-RAG	Ontology-based MindMap	0.719	0.951	0.460	0.725	0.564	0.923	0.345	0.602	0.543	0.911	0.370	0.575
	Ontology-based KG-Retriever	0.722	<u>0.948</u>	0.464	0.730	<u>0.472</u>	<u>0.905</u>	<u>0.325</u>	<u>0.512</u>	0.322	0.860	<u>0.205</u>	0.330

GraphRAG, LightRAG, and HippoRAG2 take raw documents as input and construct their own graph or memory structures. In our experiments, this category achieves an average F1 score of 0.283, which is slightly higher than Text-only RAG (0.257), but still lower than KG-based RAG (0.461) and Ontology-based KG-RAG (0.557). This is because simply converting documents into graph form cannot sufficiently preserve the section structure and condition-consequence-based rule units that are important in industrial standard documents. In particular, for table-based rules, the correspondence between row/column conditions and cell values must be maintained as a single rule unit, while Multi-hop questions require distinguishing scopes across different sections. However, Document-to-Graph RAG does not model these structures with explicit rule nodes and *scope paths*; therefore, even when it partially captures relevant evidence, it has limitations in consistently maintaining the scope of rule application and condition composition. This suggests that a knowledge representation incorporating both section structure and rule-aware triplets is more important than simple graph transformation.

Rule-aware KG leads to performance improvements. MindMap and KG-Retriever use a general KG constructed with the proposed rule-aware triplet extraction and achieve an average F1 score of 0.461. This is higher than Text-only RAG, Document-to-Graph RAG, and LLM w/full doc, suggesting that condition-consequence-based

triplet structures are effective for representing rules in industrial standard documents.

Ontology provides additional performance gains. Ontology-based KG-RAG uses an Ontology-based KG that integrates section structure into the rule-aware KG, and is implemented by extending MindMap and KG-Retriever in an ontology-aware manner. As a result, the average F1 improves from 0.461 for KG-based RAG to 0.557 for Ontology-based KG-RAG. This can be interpreted as the effect of the ontology-based KG construction method, which connects rules and entities to their corresponding *section concepts* through section structure and thereby better preserves the scope of rule application and condition structure.

4.3 Ablation Study

4.3.1 Consistent Gains Across QA Types

Figure 4 shows the F1 performance comparison by QA type. Using representative settings for each baseline category (LLM-only/LLM w/full doc: GPT-5-mini, Text-only RAG: Dense+GPT-5-mini, Document-to-Graph RAG: GraphRAG, KG-based RAG: MindMap, and Ontology-based KG-RAG: Ontology-based MindMap), Ontology-based KG-RAG achieves the highest performance across all Rule, Multi-hop, and Table types. In particular, compared with KG-based RAG, it improves by 11.1% on Rule (0.601 $\rightarrow$ 0.668), 11.3% on Multi-hop (0.515 $\rightarrow$ 0.573), and 120.2% on Table (0.287 $\rightarrow$ 0.632). These re-

Table 4: Case study on retrieval behavior for a scope-sensitive calibration-hole query. This example shows that even when similar thickness conditions are retrieved, preserving the correct section and table scope is necessary to avoid selecting a competing rule.

Question	What is the required hole diameter for ultrasonic examination of plates with a thickness greater than 225 mm and up to 300 mm?	
Gold	22 mm (Table S8.1)	
Approach	Retrieved Evidence	Output
Text-only RAG	Directly retrieved the matching Table S8.1 row for $225 \text{ mm} < \text{plate thickness} \leq 300 \text{ mm}$ and its corresponding hole diameter.	22 mm
Document-to-Graph RAG	Retrieved only a summary-level normalized thickness range, without preserving the exact table row and associated hole-diameter value.	200 to 250, excl; 250 to 300, excl
KG-based RAG	Retrieved the correct thickness condition, but mixed it with a competing ESR/VAR rule from Section S9.	10 mm
Ontology-based KG-RAG	Resolved the query to the flat-bottom-hole calibration context in Section S8 and matched the target condition with its corresponding consequence.	22 mm

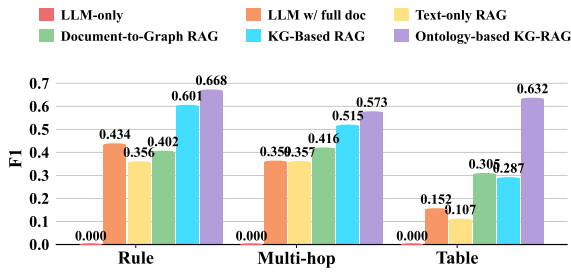


Figure 4: F1 performance comparison by QA data type. We compare the average F1 scores across Rule, Multi-hop, and Table question types on three industrial standard documents.

Table 5: Ablation study on API_2W rule data. + *Ontology* uses section structure only, + *KG* uses the rule-aware KG without section structure, and + *Ontology* + *KG* corresponds to the final Ontology-based KG.

	F1	BERTScore	BLEU	ROUGE
LLM w/full doc	0.447	0.883	0.218	0.510
+ <i>Ontology</i>	0.421	0.879	0.206	0.482
+ <i>KG</i>	0.591	0.926	0.403	0.612
+ <i>Ontology</i> + <i>KG</i>	0.612	0.933	0.434	0.638

sults suggest that the Ontology-based KG, which jointly preserves section structure and condition-consequence structure, is particularly effective for table-based condition interpretation and multi-hop reasoning. Detailed results are provided in Appendix Tables 8, 9, and 10.

4.3.2 Effect of Ontology and KG Integration

Table 5 analyzes the effects of the *section concept*-based ontology and the rule-aware KG on performance. The results show that + *KG* improves F1 from 0.447 to 0.591 compared with LLM w/full doc (+32.2%). This suggests that condition-consequence-based triplet structures are effective for representing rules in industrial standard documents.

In contrast, when using only + *Ontology*, the F1 score is 0.421, indicating that section structure alone cannot provide sufficient content knowledge. However, + *Ontology* + *KG* achieves the highest F1 score of 0.612, showing an additional improvement over + *KG* (+3.6%). This demonstrates that section structure further improves performance by complementing the rule-aware KG with the scope of rule application and structural context.

4.4 Case Study

Table 4 compares the retrieval results of different RAG settings for the same scope-sensitive calibration-hole query. Text-only RAG searches for relevant information over a relatively broad evidence space based on lexical similarity, without an explicit scope constraint. In this case, because the query has a strong surface-level match with the target row in *Table S8.1*, it retrieves the correct evidence and returns the correct answer, “22 mm,” although the retrieval process itself is not constrained by the corresponding section or table scope. Document-to-Graph RAG captures only summary-level representations of nearby thickness ranges and fails to recover the exact table row and its associated hole-diameter value. KG-based RAG retrieves the correct thickness condition, but also brings in a competing rule from Section S9, causing scope mixing and leading to the incorrect answer, “10 mm.” In contrast, Ontology-based KG-RAG first grounds the query in the flat-bottom-hole calibration context of Section S8, structurally narrowing the retrieval space, and then matches the target condition with its corresponding consequence within that scope to return the correct answer, “22 mm.” This case shows that lexical retrieval can still succeed when there is a strong surface-level match, whereas explicit scope-aware retrieval can

structurally narrow the evidence search space and reduce confusion among similar rules appearing in different document contexts.

5 Conclusion

This study proposes an Ontology-based KG construction method that combines ontology construction from section structure with rule-aware triplet extraction to effectively represent industrial standard documents containing complex conditional statements and table-based rules. The proposed method jointly preserves the section scope in which each rule should be interpreted and the condition–consequence structure within a single graph. Experimental results show that the proposed method outperforms existing approaches overall, with particularly large gains on table-based rule and multi-hop questions. These results demonstrate that an Ontology-based KG that jointly preserves section structure and condition–consequence structure is an effective knowledge representation for industrial standard document QA. This approach can serve as a foundation for future industrial-domain RAG and decision-support systems.

Limitations

Our approach is designed to stabilize LLM-based reasoning over documents that contain explicit structural information, such as section structures, conditional statements, and tables. Therefore, its direct extension to unstructured text environments with limited structural cues may be limited. However, the proposed framework is not restricted to industrial standard documents and can be applied to various rule-centric documents with clear structural units, such as technical manuals, legal and contractual documents, and policy documents. In addition, the proposed Textual Condition–Consequence Modeling provides a general way to decompose nested conditions and multiple exception rules into logical units, making it applicable to documents from diverse domains that contain complex rule structures. Meanwhile, our approach relies on LLM-based knowledge extraction and is therefore affected by model stability. In future work, we plan to reduce dependence on external LLMs and improve reliability and reproducibility by training domain-specific local models using the extracted ontology and rule patterns.

Ethical statement

This study proposes an automated system for question answering over industrial standard documents. To comply with research ethics, we considered the following aspects. First, regarding data use and copyright, the industrial standard documents used in our experiments were utilized only for academic research purposes, and we did not redistribute the original data or make commercial use of it. Second, we considered safety and potential misuse. Although the proposed model shows high accuracy, decision-making in industrial settings is directly related to safety; therefore, model outputs should not be blindly trusted without expert review. Our system is designed as a tool to assist engineers in decision-making, and we explicitly acknowledge that potential errors caused by hallucination cannot be completely ruled out.

Acknowledgments

We thank the researchers at The MiracleSoft for their support in data provision and preprocessing through our industry–academia collaboration. This work was supported by the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. RS-2020-II201373, Artificial Intelligence Graduate School Program (Hanyang University)), and by the Korea Institute of Energy Technology Evaluation and Planning (KETEP) and the Ministry of Climate, Energy & Environment (MCEE) of the Republic of Korea (No. RS-2026-25530097).

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.
- 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.
- Alexandru Constantin, Silvio Peroni, Steve Pettifer, David Shotton, and Fabio Vitali. 2016. **The document components ontology (doco)**. *Semantic Web*, 7(2):167–181.
- Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, Dasha Metropolitansky, Robert Osazuwa Ness, and Jonathan Larson. 2025. **From local to global: A graph rag approach to query-focused summarization**. *Preprint*, arXiv:2404.16130.
- 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.
- Zirui Guo, Lianghao Xia, Yanhua Yu, Tu Ao, and Chao Huang. 2024. **Lightrag: Simple and fast retrieval-augmented generation**.
- Bernal Jiménez Gutiérrez, Yiheng Shu, Weijian Qi, Sizhe Zhou, and Yu Su. 2025. **From RAG to memory: Non-parametric continual learning for large language models**. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 21497–21515. PMLR.
- 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. 2024. **End-to-end ontology learning with large language models**. In *Advances in Neural Information Processing Systems*, volume 37, pages 87184–87225. Curran Associates, Inc.
- Colin Maudry. The dita rdf project. <https://colinmaudry.github.io/dita-rdf/>. Accessed: 2026-05-26.
- 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.
- OpenAI. 2022. New and improved embedding model. <https://openai.com/index/new-and-improved-embedding-model/>.
- 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. 2024. [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.

A Dataset Details

This appendix describes the construction process, expert validation procedure, and provenance information of IndusSpec-QA. IndusSpec-QA is constructed from three industrial standard documents (ASTM A578/A578M, API_2W, and ASTM A6/A6M) and consists of three question types, Rule, Multi-hop, and Table, reflecting the section structure and rule characteristics of the source documents. Representative examples of each question type are shown in Table 6. In addition, each released QA instance includes provenance metadata that allows the corresponding evidence to be traced back to the original document; an example is provided in Figure 5.

A.1 IndusSpec-QA Construction

IndusSpec-QA is constructed based on the knowledge unit $U_i = (p_i, x_i)$ defined in Section 3.1. Each question is designed by jointly considering the *scope path* p_i and content x_i of a specific knowledge unit, thereby ensuring that the target rule under a particular *section concept* is clearly maintained. The question generation procedure differs by question type. The Rule type refers to questions that can be answered based on a single sentence or a single condition, and is mainly constructed so that the answer can be directly found within one knowledge unit. The Multi-hop type refers to questions that require step-by-step combination of multiple sections or multiple conditions, reflecting cases where rules distributed across the document must be connected. The Table type refers to questions that require jointly interpreting row/column conditions and cell values, and is constructed to explicitly evaluate table-based rules. Question drafts are generated using an LLM, with different generation strategies depending on the question type. For simple Rule questions, drafts are generated using a prompt template based on section-level content. In contrast, more challenging types, such as multi-hop questions or structural cases where the same term has different meanings depending on the section, are manually designed by domain experts.

For each finalized QA instance, we additionally retain provenance metadata that enables the corresponding answer to be traced back to its supporting evidence in the source document. Specifically, the released dataset provides `gold_doc_ids` identifying the source document, `page_no` indicating the page containing the evidence, `bbox` specifying

the location of the evidence within the page, and `context_text` containing the supporting text span. These fields allow each QA instance to be traced back to the corresponding evidence span in the original industrial standard document. Figure 5 shows an example dataset instance together with its provenance metadata.

A.2 Expert Validation

The final dataset was confirmed through a validation process conducted by domain experts. A total of eight experts participated in the QA dataset construction and validation, including one Ph.D. holder, one master’s degree holder, three master’s students, and three industry researchers working in the relevant industrial domain. Some participants also have more than ten years of practical experience in the relevant industrial field.

Expert validation was conducted according to the following criteria. (i) We checked whether each question was grounded in the actual document content. (ii) We examined whether the answer was determined by a single *scope path* or by a clear combination of rules. (iii) We verified whether numerical values, units, and technical terms exactly matched the expressions in the source document. (iv) When a question was ambiguous or allowed more than one possible answer, the corresponding example was revised or removed.

In particular, for the Multi-hop and Table types, we carefully checked whether answering the question required considering both section structure and condition composition, rather than relying on simple keyword matching alone. For the finalized instances, the released dataset also provides provenance metadata that allows the supporting evidence to be traced back to the original source document, as illustrated in Figure 5. The dataset is publicly available on Hugging Face⁵.

B Evaluation Details

In this study, we use F1 as the primary evaluation metric and additionally report BERTScore, BLEU, and ROUGE as complementary metrics. Industrial standard document QA often depends on the exact preservation of numerical values, units, inequality boundaries, technical terms, and applicability conditions; therefore, semantically similar responses may still be incorrect when these critical

⁵<https://huggingface.co/datasets/zzin/IndusSpec-QA>

```
{
  "question": "What type of welding is associated with A5.23/A5.23M?",
  "answers": ["submerged arc welding"],
  "type": "multi-hop",
  "gold_doc_ids": ["A6A6M_14"],
  "page_no": 2,
  "bbox": "[310.314, 560.671, 552.310, 580.802]",
  "context_text": "A5.23/A5.23M Low Alloy Steel Electrodes and Fluxes for Submerged Arc Welding"
}
```

Figure 5: Example of an IndusSpec-QA instance with provenance metadata. Each released instance includes the question and answer, QA type, source document identifier, page number, bounding box, and supporting context span. These fields allow each answer to be traced back to its supporting evidence in the original industrial standard document.

Table 6: Representative Question Examples from IndusSpec-QA. This table shows representative question examples extracted from the IndusSpec-QA dataset. Each 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 hole diameter required for ultrasonic examination of plates with a thickness greater than 300 mm and up to 500 mm?	hole_diameter = 29 mm

elements differ. We use token-level F1 as the primary metric because it directly reflects overlap in such answer-critical tokens between the predicted and gold answers, while BERTScore, BLEU, and ROUGE provide complementary measures of semantic similarity and textual consistency. F1 is computed after lowercasing, whitespace normalization, and whitespace-based tokenization. Accordingly, we interpret performance using these metrics jointly rather than relying on semantic similarity alone.

C Baselines & Implementation Details

This appendix provides a more detailed description of the implementation of the baselines summarized in Section 4.1.

LLM-only LLM-only is a setting in which the model generates an answer using only the query as input, without using external knowledge, retrieval results, or document context. This setting is used as a reference to examine whether an LLM can answer questions in specialized domains, such as industrial specification documents, relying solely on its internal knowledge. In the main comparison, we used GPT-5-mini, and in the additional backbone comparison, we also report results with gemini-2.0-flash and DeepSeek-v3.1. All queries were provided using the same prompt.

LLM w/full doc LLM w/full doc is a setting in which the entire document is provided as input together with the query. This setting is used as a comparative reference to evaluate performance when document information is directly provided without retrieval. In the main comparison, we used GPT-5-mini by directly inputting the full document, and in the additional backbone comparison, we used gemini-2.0-flash and DeepSeek-v3.1.

Text-only RAG Text-only RAG is a standard RAG setting in which the document is retrieved at the text chunk level, and the retrieved context is then provided to the LLM. In this study, we used both lexical retrieval and dense retrieval. For lexical retrieval, we used LangChain’s BM25Retriever, and the document was split into section-level text blocks based on section headers, which were used as retrieval units. For dense retrieval, we used text-embedding-ada-002 (OpenAI, 2022) embeddings, and the section-level documents were further split with chunk_size=500 and overlap=50 to compute semantic similarity. For each query, the top $k = 5$ sections or chunks were selected and inserted into the LLM input context to generate the answer. In the additional backbone comparison, we also used Qwen1.5-14B, Mistral-7B, and gemini-2.0-flash.

Table 7: Retrieval hyperparameters and implementation settings for all compared methods.

Method	LLM / Embedding	Retrieval Unit / Mode	Main Hyperparameters and Settings
<i>Non-Retrieval Baselines</i>			
LLM-only	gpt-5-mini	–	No retrieval
LLM w/ full doc	gpt-5-mini	Full document	Direct full-document input
<i>Standard RAG</i>			
BM25 RAG	gpt-5-mini	Section-level text blocks	Top- $k = 5$
Dense RAG	gpt-5-mini + text-embedding-ada-002	Chunked text	chunk_size=500, overlap=50, Top- $k = 5$
<i>Document-to-Graph RAG</i>			
GraphRAG	gpt-5-mini	chunk_size=1200, overlap=100, local search	max_workers=8; domain-adapted prompt settings for graph extraction and QA; preserve technical terms, numeric values, and units; concise answer formatting
LightRAG	gpt-5-mini + text-embedding-3-small	Raw text, mix retrieval	Hybrid retrieval using graph signal and vector similarity; no custom KG injection in the baseline setting
HippoRAG2	gpt-5-mini + facebook/contriever	Memory-oriented retrieval	retrieval_top_k=200, linking_top_k=5, qa_top_k=5, max_qa_steps=3, temperature = 0
<i>KG-based RAG</i>			
MindMap	gpt-5-mini + all-MiniLM-L6-v2	KG entity + triplet graph	Entity Top- $k = 5$, similarity threshold = 0.70, max 3-hop, up to 3 shortest paths, up to 30 neighbor triplets
KG-Retriever	gpt-5-mini + all-MiniLM-L6-v2	KG triplets	Similarity threshold = 0.30, max 30 triplets, 1-hop retrieval
<i>Ontology-based KG-RAG</i>			
Ontology-based MindMap	gpt-5-mini + all-MiniLM-L6-v2	Section concept paths + Ontology KG	Top-3 section paths, max 3-hop, final evidence up to 30 triplets, MMR $\lambda = 0.7$
Ontology-based KG-Retriever	gpt-5-mini + all-MiniLM-L6-v2	Ontology-level + global KG retrieval	Ontology KG-Level: 35 candidate triplets; Global KG-Level: 30 candidate triplets, including candidates prioritized by Guided Search; final evidence up to 30 triplets; max 2-hop; MMR $\lambda = 0.7$

Document-to-Graph RAG Document-to-Graph RAG baselines were implemented while preserving the original framework of each method, with only minimal domain adaptation applied to more reliably handle technical terms, numerical values, and units appearing in industrial specification documents. These adjustments were made without modifying the core graph construction and retrieval mechanisms of each method.

- GraphRAG (Edge et al., 2025) was implemented using the Microsoft GraphRAG framework. To more reliably preserve the technical context and numerical constraints of industrial specification documents, each document was split with a chunk_size=1200 tokens and an overlap=100 tokens. We used the local search setting to focus retrieval on specific clauses and numerical conditions, rather than on document-level summaries. In addition, to reflect the characteristics of industrial specification documents, the graph extraction and question-answering prompts were adjusted to prevent technical terms, numerical values, and units from being omitted.
- LightRAG (Guo et al., 2024) was implemented

using the official implementation by directly indexing raw text. For text embeddings, we used text-embedding-3-small with 1536 dimensions, and the retrieval mode was set to mix, which jointly considers graph signals and vector similarity. In addition, to reduce context misses caused by domain-specific technical terms in industrial specification documents, we conservatively adjusted the entity matching threshold while preserving the original retrieval mechanism.

- HippoRAG2 (Gutiérrez et al., 2025) was implemented as a memory-oriented graph retrieval method. We used facebook/contriever for dense retrieval and Personalized PageRank initialization, and gpt-5-mini for OpenIE and answer generation. The main retrieval hyperparameters were set to retrieval_top_k=200, linking_top_k=5, qa_top_k=5, and max_qa_steps=3. To ensure output stability, the generation temperature was fixed to 0.

KG-based RAG KG-based RAG retrieves evidence from a general KG constructed using the rule-aware triplet extraction process of this study.

Table 8: Performance comparison on A578/A578M across different question types. The results compare the performance of various QA models on table, rule, and multi-hop questions.

Type	Method	table				rule				multi-hop			
		F1	BERTScore	BLEU	ROUGE	F1	BERTScore	BLEU	ROUGE	F1	BERTScore	BLEU	ROUGE
LLM-only	GPT-5-mini	0.000	0.780	0.000	0.000	0.000	0.797	0.000	0.004	0.000	0.798	0.000	0.000
	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-5-mini	0.000	0.836	0.000	0.322	0.375	0.881	0.189	0.432	0.396	0.882	0.172	0.485
	gemini-2.0-flash	0.000	0.773	0.000	0.000	0.255	0.855	0.151	0.283	0.328	0.872	0.162	0.391
	DeepSeek-v3.1	0.000	0.783	0.000	0.134	0.307	0.869	0.164	0.365	0.358	0.883	0.161	0.455
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	0.103	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-5-mini	0.000	0.829	0.000	0.303	0.349	0.875	0.178	0.402	0.380	0.879	0.168	0.469
	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	0.106	0.310	0.302	0.882	0.104	0.386
	Dense+Mistral-7B	0.011	0.775	0.001	0.185	0.251	0.859	0.101	0.305	0.299	0.867	0.105	0.399
	Dense+GPT-5-mini	0.000	0.827	0.000	0.303	0.315	0.869	0.156	0.364	0.356	0.875	0.161	0.444
	Dense+gemini-2.0-flash	0.000	0.803	0.000	0.281	0.296	0.868	0.161	0.337	0.407	0.887	0.216	0.480
Document-to-Graph RAG	GraphRAG (Edge et al., 2025)	0.313	0.812	0.058	0.313	0.402	0.851	0.189	0.403	0.462	0.848	0.195	0.472
	LightRAG (Guo et al., 2024)	0.248	0.691	0.039	0.248	0.278	0.771	0.116	0.280	0.345	0.795	0.100	0.367
	HippoRAG2 (Gutiérrez et al., 2025)	0.068	0.746	0.000	0.249	0.286	0.814	0.122	0.413	0.269	0.821	0.089	0.481
KG-based RAG	MindMap (Wen et al., 2024)	0.544	0.899	0.399	0.594	0.593	0.922	0.371	0.599	0.647	0.939	0.310	0.653
	KG-Retriever (Chen et al., 2025)	0.077	0.841	0.011	0.188	0.724	0.948	0.476	0.727	0.674	0.937	0.355	0.679
Ontology-based	Ontology-based MindMap	0.928	0.985	0.896	0.931	0.722	0.951	0.470	0.727	0.679	0.946	0.365	0.691
KG-RAG	Ontology-based KG-Retriever	0.385	0.888	0.176	0.500	0.725	<u>0.949</u>	0.478	0.730	0.742	0.950	0.420	0.752

Table 9: API_2W Performance Comparison across different question types. The results compare the performance of various QA models on table, rule, and multi-hop questions.

Type	Method	table				rule				multi-hop			
		F1	BERTScore	BLEU	ROUGE	F1	BERTScore	BLEU	ROUGE	F1	BERTScore	BLEU	ROUGE
LLM-only	GPT-5-mini	0.000	0.800	0.000	0.000	0.000	0.790	0.000	0.001	0.000	0.795	0.000	0.003
	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-5-mini	0.294	0.890	0.082	<u>0.531</u>	0.454	0.884	0.218	0.514	0.330	0.876	0.146	0.402
	gemini-2.0-flash	0.048	0.811	0.010	0.079	0.322	0.858	0.164	0.342	0.224	0.842	0.113	0.250
	DeepSeek-v3.1	0.282	0.865	0.059	0.365	0.367	0.870	0.189	0.421	0.247	0.864	0.111	0.335
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-5-mini	0.084	0.833	0.018	0.179	0.409	0.874	0.192	0.451	0.328	0.870	0.147	0.385
	BM25+gemini-2.0-flash	0.110	0.830	0.026	0.218	0.396	0.873	0.191	0.428	0.322	0.870	0.178	0.369
	Dense+Qwen1.5-14B	0.334	<u>0.906</u>	0.061	<u>0.531</u>	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-5-mini	0.275	0.877	0.064	0.443	0.419	0.879	0.207	0.479	0.378	0.880	0.176	0.439
	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
Document-to-Graph RAG	GraphRAG (Edge et al., 2025)	0.446	0.846	0.151	0.445	0.401	0.840	0.180	0.403	0.407	0.838	0.178	0.399
	LightRAG (Guo et al., 2024)	0.417	0.886	<u>0.169</u>	0.417	0.323	0.856	0.120	0.322	0.268	0.864	0.086	0.266
	HippoRAG2 (Gutiérrez et al., 2025)	0.019	0.721	0.007	0.175	0.304	0.828	0.113	0.471	0.249	0.803	0.099	0.381
KG-based RAG	MindMap (Wen et al., 2024)	0.142	0.853	0.083	0.199	0.583	0.917	0.412	0.607	0.435	0.894	0.268	0.457
	KG-Retriever (Chen et al., 2025)	0.042	0.812	0.026	0.056	0.704	0.943	0.475	0.716	0.420	0.892	0.272	0.437
Ontology-based	Ontology-based MindMap	0.470	0.910	0.185	0.536	0.632	0.935	0.452	0.657	0.552	0.916	0.345	0.581
KG-RAG	Ontology-based KG-Retriever	0.108	0.847	0.058	0.201	0.715	0.946	0.508	0.728	<u>0.472</u>	<u>0.902</u>	<u>0.313</u>	<u>0.492</u>

relations, evidence candidates are restricted to document sections relevant to the query, effectively suppressing structural noise that may arise from global KG traversal. Then, local rule and entity relations are collected through Neighbor Search, and Path Search is performed to identify shortest paths between entity pairs. In this process, ontology relations representing the document structure are excluded from reasoning paths, so that only substantive rule 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 through `has_condition[_AND/_OR]` and `has_consequence[_AND/_OR]` relations.

In the **Evidence Graph Aggregation** stage, the retrieved triplets are integrated into a single reasoning graph through deduplication and structural normalization. Rather than simply listing triplets, evidence is aggregated at the condition-consequence unit, or case level, to prevent the logical meaning of each rule from being fragmented. The resulting reasoning graph provides evidence that jointly reflects the structural context of the document and its rule structure.

In the **LLM-integrated Reasoning** stage, the final reasoning graph is inserted into the LLM prompt to perform question answering. The LLM conducts reasoning along ontology-refined evidence paths, suppressing inferences that are irrelevant to the document structure and generating con-

Table 10: A6/A6M Performance Comparison across different question types. The results compare the performance of various QA models on table, rule, and multi-hop questions.

Type	Method	table				rule				multi-hop			
		F1	BERTScore	BLEU	ROUGE	F1	BERTScore	BLEU	ROUGE	F1	BERTScore	BLEU	ROUGE
LLM-only	GPT-5-mini	0.000	0.787	0.000	0.000	0.000	0.794	0.000	0.000	0.000	0.796	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	0.098	0.047	0.817	0.011	0.053
LLM w/full doc	GPT-5-mini	0.161	0.852	0.048	0.268	0.473	0.892	0.262	0.531	0.351	0.876	0.165	0.429
	gemini-2.0-flash	0.012	0.785	0.003	0.018	0.233	0.844	0.111	0.256	0.179	0.839	0.092	0.211
	DeepSeek-v3.1	0.026	0.792	0.008	0.039	0.370	0.879	0.213	0.423	0.229	0.853	0.136	0.282
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-5-mini	0.046	0.805	0.012	0.060	0.366	0.871	0.177	0.407	0.333	0.872	0.150	0.411
	BM25+gemini-2.0-flash	0.016	0.788	0.003	0.030	0.387	0.882	0.195	0.423	0.348	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-5-mini	0.047	0.804	0.012	0.079	0.334	0.864	0.172	0.372	0.338	0.874	0.161	0.422
	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
Document-to-Graph RAG	GraphRAG (Edge et al., 2025)	0.156	0.812	0.061	0.155	0.403	0.849	0.219	0.418	0.381	0.841	0.191	0.389
	LightRAG (Guo et al., 2024)	0.086	0.622	0.018	0.085	0.266	0.763	0.115	0.285	0.327	0.776	0.114	0.339
	HippoRAG2 (Gutiérrez et al., 2025)	0.088	0.754	0.000	0.311	0.388	0.848	0.148	0.529	0.263	0.816	0.099	0.455
KG-based RAG	MindMap (Wen et al., 2024)	0.176	0.840	0.050	0.190	0.627	0.931	0.419	0.631	0.464	0.903	0.264	0.497
	KG-Retriever (Chen et al., 2025)	0.030	0.792	0.006	0.026	0.697	0.943	0.476	0.709	0.451	0.906	0.251	0.479
Ontology-based	Ontology-based MindMap	0.499	0.895	0.334	0.547	0.649	0.938	0.471	0.651	0.487	0.922	0.280	0.511
KG-RAG	Ontology-based KG-Retriever	0.067	0.803	0.024	0.071	0.768	0.955	0.540	0.776	0.521	0.924	0.290	0.558

sistent answers. Through this process, Ontology-based MindMap implements a collaborative reasoning pipeline that combines the structural strengths of the KG with the linguistic reasoning capability of the LLM.

D.2 Ontology-based KG-Retriever

The method consists of two evidence extraction stages, namely the Ontology KG-Level and the Global KG-Level, followed by a final stage that integrates the extracted evidence and inserts it into the LLM prompt.

In the **Ontology KG-Level**, each knowledge unit $U_i = (p_i, x_i)$ in the document is used as the basic retrieval unit. Here, p_i denotes the *scope path* from the root section to the section containing the current content unit, and x_i denotes the content unit under the corresponding *section concept*, such as text, table, or footnote. First, x_i is encoded using Sentence-BERT (all-MiniLM-L6-v2) (Reimers and Gurevych, 2019), a sentence embedding function, to obtain the knowledge unit embedding $\mathbf{d}_i$. The query q is also transformed into a query embedding $\mathbf{q}$ in the same embedding space, and the cosine similarity $\cos(\mathbf{q}, \mathbf{d}_i)$ between the two embeddings is computed to measure the semantic relevance of each knowledge unit.

Based on this similarity-based ranking, the top three sections are selected and used to form a coarse-grained candidate set. Then, 1-hop traversal is first performed over the KG triplets contained in each section, and if sufficient evidence is not obtained, the traversal is sequentially expanded up to

2 hops to collect additional evidence. Specifically, 20, 10, and 5 triplets are collected from the selected sections according to their importance, forming a total of 35 fine-grained triplets. This allows the retriever to reflect both the local context closely related to the query and the hierarchical structure of the document.

In the **Global KG-Level**, global traversal is performed over the entire knowledge graph without section-level constraints. Using the query embedding $\mathbf{q}$, both 1-hop and 2-hop neighbor traversal are conducted to collect global structural clues beyond local constraints. In addition, a **Guided Search module** is applied within this global retrieval stage to compute the direct cosine similarity between the query embedding and the embeddings of the textual representations of triplets, including table-based representations. Based on this similarity, semantically relevant triplets are prioritized and selected as part of the global candidate set. As a result, a total of 30 triplets are collected at the Global KG-Level. Guided Search operates as an independent module that complements global structure-based traversal from the perspective of semantic similarity.

Finally, the evidence collected from the Ontology KG-Level and the Global KG-Level is integrated to construct a candidate set of 65 triplets. The candidate triplets are then re-ranked based on their semantic similarity to the query, and the final 30 triplets are selected and inserted into the prompt of the LLM (GPT-5-mini) for response generation. Through this multi-stage evidence extraction and re-ranking process, Ontology-based

KG-Retriever jointly reflects the semantic context of the query and the structural information of the KG, thereby supporting more accurate question answering. Table 7 summarizes the main retrieval settings of Ontology-based MindMap and Ontology-based KG-Retriever.

D.3 Performance under Different k -Hop Neighborhoods

Figure 7 presents the experimental results according to the number of hops for the two proposed methods that incorporate the ontology structure. Ontology-based MindMap achieved the best performance at 3-hop, whereas Ontology-based KG-Retriever achieved the best performance at 2-hop. This result appears to be attributable to the different structural characteristics of the retrieval strategies used by the two methods. Specifically, Ontology-based KG-Retriever explores related nodes in a multi-dimensional manner at both the ontology and KG levels, whereas Ontology-based MindMap follows a relatively linear traversal strategy. These structural differences can explain why the optimal number of hops differs between the two methods. Therefore, in our experiments, we applied the optimal hop setting for each method.

D.4 Component Attribution under Fixed Traversal

To isolate the contributions of the ontology representation and the ontology-aware traversal mechanism, we conduct a fixed-traversal graph-swap ablation. In the *Graph Swap* setting, the original MindMap traversal, query encoder, hop limit, evidence budget, and answer generator are kept unchanged, while only the Rule-aware KG is replaced with the Ontology-based KG. Figure 8 reports the incremental F1 gains obtained from each component. Introducing the ontology representation alone improves performance across all QA types, yielding gains of 0.168, 0.027, and 0.065 F1 on Table, Rule, and Multi-hop questions, respectively, and a weighted F1 gain of 0.079. Applying ontology-aware traversal on top of the same Ontology-based KG provides further gains across all QA types, including an additional 0.072 in weighted F1. These results show that the performance improvement is not solely attributable to the modified traversal procedure: the ontology representation itself provides a measurable benefit under fixed traversal, while ontology-aware traversal contributes an additional complementary gain.

E KG Normalization and Pruning Details

This appendix supplements the KG normalization and pruning procedures briefly described in Section 3.4. The purpose of this stage is not to generate new rules, but to improve the representational consistency of the Ontology-based KG by reducing notational variations of the same concept and structural redundancy.

E.1 Synonym Dictionary Construction

To mitigate the issue where identical or similar concepts are redundantly represented with different surface forms in the constructed KG, and to ensure overall representational consistency, we construct synonym dictionaries for entities and relations.

First, to construct the entity dictionary, we define the final embedding for each entity a by integrating textual semantics, local context, and global structural information, as shown in Eq. 4.

$$\mathbf{e}_{\text{final}}(a) = [\mathbf{e}_{\text{text}}(a) ; \mathbf{e}_{\text{ctx}}(a) ; \mathbf{e}_{\text{pos}}(a)], \quad (4)$$

where the textual semantic embedding $\mathbf{e}_{\text{text}}(a)$ reflects the semantic similarity between entities using Sentence-BERT (all-MiniLM-L6-v2). To capture how each entity is actually used in the KG, we compute the local context embedding $\mathbf{e}_{\text{ctx}}(a)$ using Personalized PageRank (PPR) (Gasteiger et al., 2022), which reflects the contextual relevance and importance between a given entity and its neighboring nodes. In addition, we derive the global structural embedding $\mathbf{e}_{\text{pos}}(a)$ using Spectral Embedding (Belkin and Niyogi, 2001) based on the eigenvectors of the graph Laplacian, thereby capturing the overall structural position and role of the entity. Combining these two methods enables the embedding representation to jointly consider 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 semantically and structurally similar entities. Entity and relation expressions in the KG exhibit uneven domain-specific frequencies and distributions, and the sizes of synonym sets are difficult to assume in advance. We adopt HDBSCAN because it can automatically identify semantically dense groups of representations without explicitly specifying the number of clusters, while also separating noisy expressions from the clusters.

For the relation dictionary, we explicitly incorporate the functional characteristics of relations

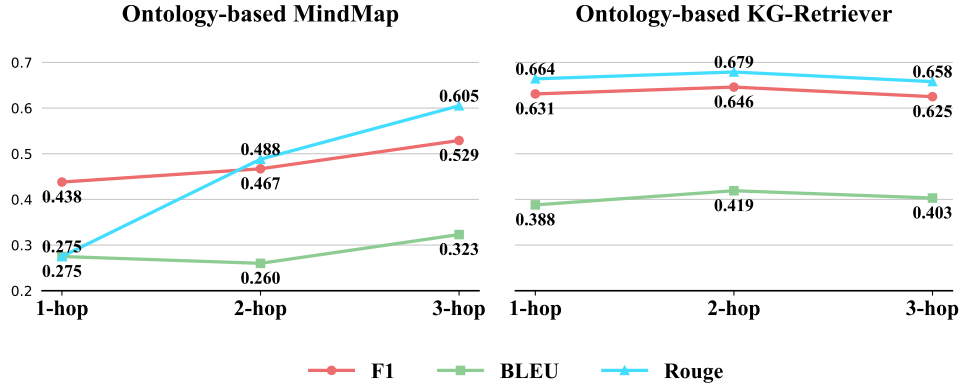


Figure 7: Performance across different k -hop settings using API_2W rule data. Ontology-based MindMap achieved the highest performance at 3-hop across all evaluation metrics, while Ontology-based KG-Retriever achieved the highest performance at 2-hop.

Table 11: Ablation study on contextual and graph-structural information for synonym dictionary construction.

Setting	Method	F1	BERTScore	BLEU	ROUGE
Text-only	Ontology-based MindMap	0.408	0.891	0.242	0.468
	Ontology-based KG-Retriever	0.626	0.931	0.438	0.646
Graph-only	Ontology-based MindMap	0.396	0.894	0.234	0.459
	Ontology-based KG-Retriever	0.604	0.925	0.424	0.623
Text + Graph (Ours)	Ontology-based MindMap	0.529	0.917	0.323	0.605
	Ontology-based KG-Retriever	0.646	0.933	0.419	0.679

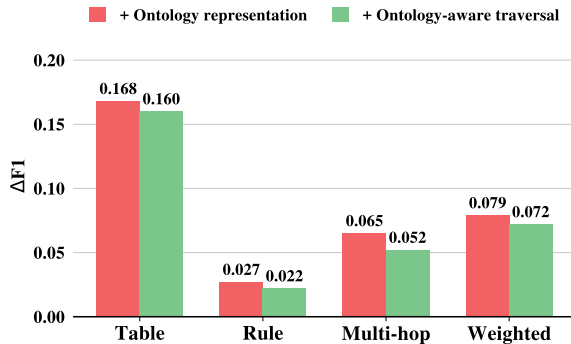


Figure 8: Incremental F1 gains from ontology representation and ontology-aware traversal. Both components improve performance across all QA types, showing complementary contributions.

in addition to an embedding configuration similar to that used for entities. Specifically, for each relation r , in addition to textual semantics, contextual information, and structural information, we further introduce a Selectional Preference embedding $\mathbf{e}_{\text{prefer}}(r)$, which reflects the distribution of subject-object entity types that the relation primarily connects:

$$\mathbf{e}_{\text{final}}(r) = [\mathbf{e}_{\text{text}}(r); \mathbf{e}_{\text{ctx}}(r); \mathbf{e}_{\text{pos}}(r); \mathbf{e}_{\text{prefer}}(r)]. \quad (5)$$

Table 12: Statistics of pruned triplets across documents.

Document	Pruned	Total	Ratio
API_2W	8	5,128	0.156%
A578/A578M	5	1,986	0.252%
A6/A6M	386	53,508	0.721%

where $\mathbf{e}_{\text{prefer}}(r)$ is designed to capture not only semantic similarity but also functional similarity of relations by modeling the type distribution of entity pairs (u, v) in which relation r frequently appears as a probability distribution $p(t | u, v)$. Finally, the relation embeddings are also clustered using HDBSCAN, and within each cluster, the relation closest to the cluster center is selected as the canonical predicate to standardize relation expressions. The resulting entity-relation synonym dictionary is then used in the subsequent KG normalization and refinement process, contributing to reducing redundant expressions and improving the structural consistency of the graph.

E.2 KG Refinement and Pruning

Ontology-based KG may include semantically equivalent expressions, repeatedly extracted triplets, and low-information structures while pre-

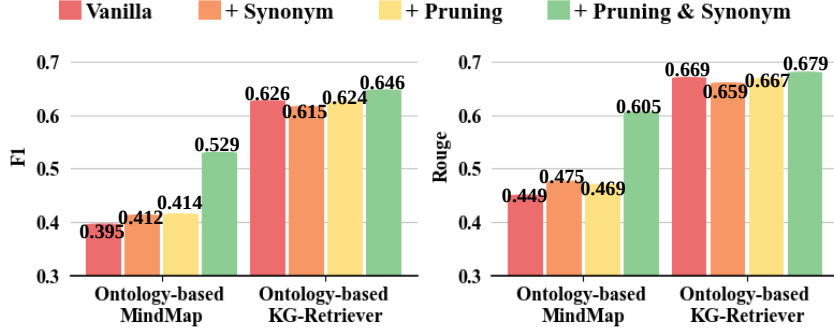


Figure 9: Effect of the synonym dictionary and pruning modules on QA performance (F1 and ROUGE) on API_2W rule data. Ontology-based MindMap shows progressive gains as the refinement modules are sequentially applied, whereas Ontology-based KG-Retriever performs best when synonym dictionary and pruning are applied together.

serving document structure and rule information. Therefore, a refinement and pruning process is required to improve the consistency of the graph and the efficiency of retrieval. To this end, we first perform conceptual normalization using the constructed synonym dictionary, which merges entities and relations into their canonical forms. This reduces conceptual redundancy and prevents nodes with the same meaning from being dispersed across the graph. Next, to improve structural efficiency, we apply a multi-stage structural pruning procedure that removes low-information or redundant elements. Here, nodes are represented as a set V containing subject and object concepts, and relations representing semantic connections between nodes are represented as a set R . Each knowledge triplet is expressed as (u, r, v) , where $u \in V$ is the subject node, $r \in R$ is the relation, and $v \in V$ is the object node. The pruning process consists of the following three steps:

- (i) **Self-loop Pruning:** Self-loop relations that point to themselves, defined as $T_{\text{self}} = \{(u, r, v) \in T \mid u = v\}$, are removed because they do not provide meaningful information.
- (ii) **Reciprocal Edge Pruning:** When reciprocal edges coexist, the edge with the lower occurrence frequency is removed, as shown in Eq. 6.

$$T_{\text{inv}} = \left\{ \arg \min_{t \in \{(u,v), (v,u)\}} w(t) \mid (u,v), (v,u) \in T \right\}. \quad (6)$$

If the weights w are identical, both reciprocal edges are retained because there is no basis for determining a preferred direction, thereby preventing arbitrary information loss.

- (iii) **Duplicate Triplet Pruning:** If completely identical triplets are extracted multiple times from the

same document, only one instance is retained:

$$T_{\text{dup}} = \{(u, r, v)_1, \dots, (u, r, v)_k \mid (u, r, v)_i \in T, k > 1\}. \quad (7)$$

Afterward, isolated nodes are removed, yielding the final triplet set in Eq. 8:

$$T_{\text{final}} = T \setminus (T_{\text{self}} \cup T_{\text{inv}} \cup T_{\text{dup}}). \quad (8)$$

The resulting T_{final} is then used to construct the KG.

E.3 Text and Graph Signal Ablation

Section E.1 constructs a synonym dictionary to integrate synonymous entity and relation expressions and ensure conceptual consistency. In this process, we jointly use string-based and context-based similarities between expressions, as well as graph-structural information. To examine the contribution of each input component, we conduct an additional ablation study. We divide the input configuration into Text-only, Graph-only, and Text+Graph, and compare the performance of Ontology-based MindMap and Ontology-based KG-Retriever under each setting.

As shown in Table 11, both the Text-only and Graph-only settings achieve a certain level of performance, indicating that textual semantics and graph structure each provide useful information independently. However, the Text+Graph setting achieves the best overall performance. In particular, Ontology-based KG-Retriever records the highest scores in F1, BERTScore, and ROUGE, showing that context-based semantic information and graph-based structural information complement each other. In other words, using both text and graph signals is the most stable setting for constructing the synonym dictionary.

E.4 Pruning for Structural Consistency

In this study, we applied a pruning step to ensure the structural stability and consistency of the extracted triplet set. This step serves as a post-processing procedure to maintain a more refined graph structure by mitigating structural redundancy, cyclic relations, and semantically redundant connections that may occur within the triplet graph. In particular, as the document structure becomes more complex, this refinement process can play a meaningful role in inspecting the integrity of the graph and reducing potential noise. Table 12 presents the document-level pruning statistics, showing the actual scale of pruning applied to each document.

The number of triplets removed through pruning ranges from 5 to 386 across documents, corresponding to 0.16%–0.72% of the total triplets. Although the removal ratio is relatively small, this indicates that the pruning step is designed to refine structural redundancy and potential noise only within the necessary scope, without excessive information removal.

Figure 9 shows the performance changes when the synonym dictionary and pruning are applied individually or jointly. For Ontology-based MindMap, F1 and ROUGE gradually improve as the two modules are applied step by step. In contrast, for Ontology-based KG-Retriever, some metrics slightly decrease when either the synonym dictionary or pruning is applied alone, whereas the best performance is achieved when both modules are applied together. This suggests that each refinement step does not always provide an independent gain, but the representational consistency and structural stability of the graph are best ensured when synonym normalization and pruning operate together. Therefore, pruning can be interpreted not as a standalone module, but as a conservative post-processing step that is most effective when combined with the synonym dictionary.

F Rule-aware Triplet Extraction Details

This appendix supplements the details of rule-aware triplet extraction introduced in Section 3.3. In particular, we illustrate how table-based rules, which represent one of the most complex forms in industrial specification documents, are converted into condition–consequence structures.

F.1 Table-to-Rule Transformation Example

Figure 3 in the main text illustrates how a selected cell in a table from an industrial specification document is converted into a rule with a condition–consequence structure. In this example, the selected cell is interpreted as a rule that applies only when both the row condition and the column condition are satisfied. Specifically, the row condition is $200 \text{ mm} < \text{width} < 500 \text{ mm}$, and the column condition is $\text{thickness} < 5 \text{ mm}$ or $\text{weight} < 70 \text{ kg/m}^2$. The consequence is $\text{permitted variation} = 3 \text{ mm}$. Therefore, this cell is represented as the following rule instance:

$$C_1 \wedge (C_2 \vee C_3) \rightarrow C_4. \quad (9)$$

In this study, such a rule is represented as a single case node, and when different logical operators appear together, a `logic_group` node is used to preserve the compositional structure. Accordingly, this example is converted into the following set of triplets:

- (case1, has_condition_AND, $200 \text{ mm} < \text{width} < 500 \text{ mm}$)
- (case1, has_condition_AND, logic_group1)
- (logic_group1, has_condition_OR, $\text{thickness} < 5 \text{ mm}$)
- (logic_group1, has_condition_OR, $\text{weight} < 70 \text{ kg/m}^2$)
- (case1, has_consequence, $\text{permitted variation} = 3 \text{ mm}$)

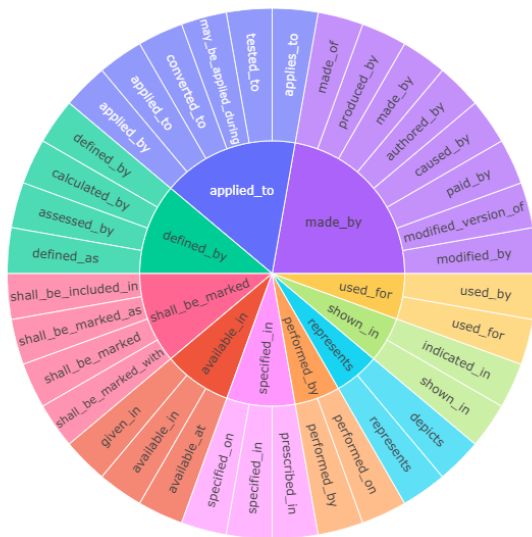
This example shows that a single table cell is transformed not merely into a simple value, but into a rule unit that jointly includes conditions and a consequence. In other words, the proposed table-to-rule transformation uses row/column headers together with the cell value, enabling table-based rules to be represented within the same rule schema as textual conditional statements.

F.2 Prompt Template

In this study, we used two types of prompts to extract both text and tables into condition–consequence structures. The first is a table-specific triplet extraction prompt, which is designed to convert each non-empty cell into a case node, extract row and column headers as conditions, and extract the cell value as the consequence. The second is a text-specific triplet extraction prompt, which is

designed to extract only semantically meaningful relations from general sentences, equations, and conditional statements. The prompt templates actually used in this study are provided in Appendix J.

G.1 Synonym Dictionary Visualization



G.2 Ontology-based KG Visualization

H Qualitative Applicability Beyond the Benchmark Documents

H.1 Generalization to Different Hierarchy Depths

H.2 Cross-domain Applicability

⁶[https://mdcpp.com/doc/standard/ISO10993-1-2018\(E\).pdf](https://mdcpp.com/doc/standard/ISO10993-1-2018(E).pdf)

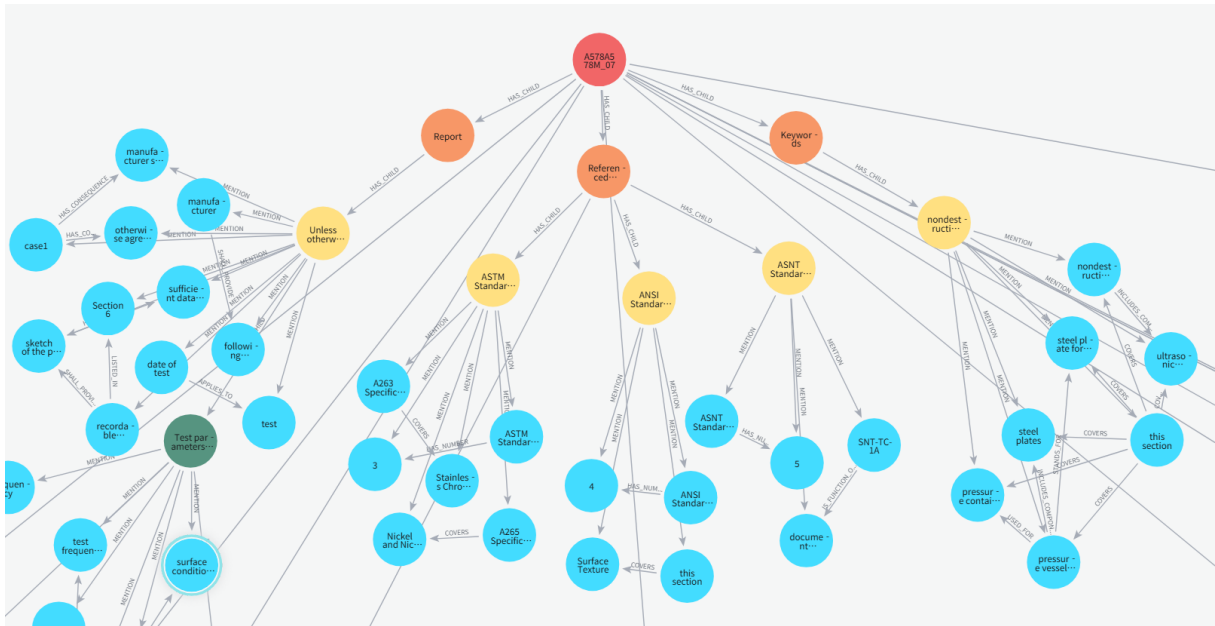


Figure 11: Visualization of an Ontology-based KG using Neo4j.

Table 13: Cross-domain document characteristics used to illustrate the generalizability of the proposed framework.

Document	Organization	Domain	Hierarchy depth	Conditional Rule
A578/A578M	ASTM	Steel	3	Yes
API 2W	API	Steel	3	Yes
A6/A6M	ASTM	Steel	3	Yes
ISO 10993-1	ISO	Medical device	4	Yes
DIN EN 10027-1	DIN	Steel	2	Yes

These qualitative cases provide initial evidence that the proposed representation can be applied beyond the three benchmark documents to technical standards with different organizational structures and section structures. A broader quantitative evaluation across additional domains and document types will be investigated in future work.

I Additional Analyses

This section provides qualitative examples that demonstrate the importance of preserving *scope paths* based on *section concepts* for industrial standard document QA, as well as the importance of preserving logical structures in complex tables.

I.1 Extended Component Ablation

Section 4.3.2 reports a representative subset of the component ablation results, while Table 15 presents the complete evaluation across all three benchmark documents. As shown in the table, the + *Ontology* + *KG* setting consistently achieves the best overall performance across datasets. In contrast, ontology alone does not consistently improve over the document-only setting, indicating that section

scope is most effective when combined with the condition–consequence structures preserved by the rule-aware KG.

I.2 Scope Collision Case Study

In Figure 12, Table S8.1 and Table S9.1 in ASTM A578/A578M both define the calibration hole diameter according to plate thickness, and their table structures and key terms are highly similar. However, because the two tables are located under different supplementary requirements, even the same thickness expression may correspond to different rules. Thus, the key issue in this case is not simple lexical matching, but correctly identifying which section scope and table scope the query refers to.

For example, Table S8.1 defines the calibration hole diameter for *Ultrasonic Examination Using Flat Bottom Hole Calibration*, whereas Table S9.1 defines a separate calibration criterion for *Electroslag Remelted (ESR) and Vacuum-Arc Remelted (VAR) Plates*. Therefore, even when superficially similar thickness ranges appear, the correct answer may differ depending on which supplementary requirement the rule belongs to.

Table 14: Cross-domain case studies demonstrating the applicability of the proposed ontology construction methodology

Document	Example Content	Constructed Triplets
ISO 10993-1	p_i (scope path): <i>Categorization of medical devices</i> → <i>General</i>	(Categorization_of_medical_devices, has_child, General) (General, mention, case1) (case1, has_condition_AND, medical_devices_include_implanted_component)
	x_i : Medical devices including both an implanted component and a delivery system used only during surgery should be assessed separately.	(case1, has_condition_AND, medical_devices_include_delivery_system) (case1, has_condition_AND, delivery_system_used_during_surgical_procedure) (case1, has_consequence, implant_assessed_separately_from_delivery_system)
DIN EN 10027-1	p_i (scope path): <i>Principles</i> → <i>Allocation of steel names</i>	(Principles, has_child, Allocation_of_steel_names) (Allocation_of_steel_names, mention, case1) (case1, has_condition_OR, steels_specified_in_European_Standards_EN)
	x_i : Steel names for steels specified in EN, TS, and TR shall be allocated by the ECISS Technical Committee concerned.	(case1, has_condition_OR, steels_specified_in_Technical_Specifications_TS) (case1, has_condition_OR, steels_specified_in_Technical_Reports_TR) (case1, has_consequence, steel_names_allocated_by_the_ECISS_Technical_Committee_concerned)

TABLE S8.1 Calibration Hole Diameter as a Function of Plate Thickness (S8)				
Plate Thickness, in. [mm]	4–6 [100–150]	>6–9 [>150–225]	>9–12 [>225–300]	>12–20 [>300–500]
Hole Diameter, in. [mm]	5/16 [16]	3/4 [19]	7/8 [22]	1 1/2 [29]

TABLE S9.1 Calibration Hole Diameter as a Function of Plate Thickness (S9)				
Plate Thickness, in. [mm]	1–4 [25–100]	>4–8 [>100–200]	>8–12 [>200–300]	>12–16 [>300–400]
Hole Diameter, in. [mm]	1/8 [3]	1/4 [6]	3/8 [10]	1/2 [13]

Figure 12: Calibration hole diameter rules from ASTM A578/A578M that differ by section scope.

Consider the following question: “What is the required hole diameter for ultrasonic examination of plates with a thickness greater than 225 mm and up to 300 mm?” Because this question does not specify ESR/VAR plates, it should be interpreted according to the rule in Table S8.1, which covers flat-bottom-hole calibration, rather than Section S9. Therefore, the correct answer is 22 mm.

Table 16 shows how the retrieval results differ across retrieval methods for this question. In this case, Text-only RAG directly retrieved the relevant row from Table S8.1 and returned the correct answer, 22 mm. This is because the A578/A578M document is relatively short, and the surface expression of the query strongly matches the thickness interval in the corresponding row. By contrast, Document-to-Graph RAG returned only a summarized range expression for the relevant interval, failing to recover the exact hole diameter value. KG-based RAG retrieved evidence related to $225 \text{ mm} < \text{plate thickness} \leq 300 \text{ mm}$; however, because the section/table scope was not sufficiently separated, it also retrieved a similar rule from S9,

namely $200 \text{ mm} < \text{plate thickness} \leq 300 \text{ mm} \rightarrow 10 \text{ mm}$, leading to an incorrect answer. In contrast, Ontology-based KG-RAG first identifies the flat-bottom-hole calibration context of Section S8 and then connects the condition $225 \text{ mm} < \text{plate thickness} \leq 300 \text{ mm}$ with the result *hole diameter* = 22 mm within the corresponding table scope, thereby deriving the correct answer more reliably.

This case shows that scope collision does not always appear as a failure of lexical retrieval. When the query expression strongly matches a specific row, Text-only RAG can also retrieve the correct answer. However, without explicitly preserving section scope and table scope, the model can easily confuse the target rule with rules from other supplementary requirements that have similar table structures. Therefore, the key issue is to preserve not only the value itself, but also the section scope under which the value is valid.

Table 15: Complete component ablation results across all three benchmark datasets, averaged over three runs. The best-performing setting for each dataset is shown in bold.

Dataset	Setting	F1	BERTScore	BLEU	ROUGE
A578/A578M	Document	0.372	0.881	0.182	0.446
	+ KG	0.601	0.924	0.362	0.607
	+ Ontology	0.315	0.869	0.156	0.379
	+ Ontology + KG	0.719	0.951	0.460	0.725
API 2W	Document	0.377	0.884	0.159	0.495
	+ KG	0.410	0.891	0.272	0.451
	+ Ontology	0.369	0.879	0.157	0.462
	+ Ontology + KG	0.561	0.922	0.341	0.599
A6/A6M	Document	0.272	0.866	0.121	0.364
	+ KG	0.325	0.863	0.160	0.354
	+ Ontology	0.166	0.830	0.078	0.208
	+ Ontology + KG	0.546	0.907	0.509	0.578

I.3 Row-level Ambiguity Within a Large Table

The case in Table 17 is based on a table on page 45 of the A6/A6M document. This case illustrates row-level ambiguity within a single large table, rather than a section scope issue. The query asks for the area value corresponding to the specific metric designation $W920 \times 588$. However, the same table contains many highly similar designations, such as $W920 \times 586$, $W920 \times 652$, and $W920 \times 970$, making it essential to identify the correct row identity.

In this case, Text-only RAG retrieves the overall table context but fails to isolate the exact row corresponding to the target designation, and therefore cannot generate the correct answer. Document-to-Graph RAG also fails to preserve the row-level designation accurately due to summary-based retrieval, incorrectly determining that the corresponding entry does not exist. In contrast, KG-based RAG retrieves evidence containing *designation_metric* = $W920 \times 588$ and the corresponding row, *case68*, thereby deriving the correct answer, $75,000 \text{ mm}^2$. However, during this process, neighboring designations such as $W920 \times 586$, $W920 \times 652$, and $W920 \times 970$ are also retrieved, indicating that entity-centric traversal can be vulnerable to row-level noise.

By contrast, Ontology-based KG-RAG first grounds the query to the table section A2. *Dimensions of Standard Shape Profiles*, and then links the exact designation $W920 \times 588$ to the corresponding row-level result, *area* = $75,000 \text{ mm}^2$. That is, in this setting, not only the section structure but also the row identity and attribute–value relations within the table are preserved, reducing confusion with

adjacent similar rows. This case demonstrates that, for QA over industrial specification documents, a section-grounded ontology alone is not sufficient; it is also necessary to preserve the internal conditional structure of tables.

Table 16: Case study with condensed retrieval traces. The query requires resolving the correct table scope between Table S8.1 and Table S9.1 in ASTM A578/A578M.

Query: What is the required hole diameter for ultrasonic examination of plates with a thickness greater than 225 mm and up to 300 mm? Gold: 22 mm (Table S8.1)		
Approach	Condensed Retrieval Trace	Prediction
Text-only RAG (gpt-5-mini ada-002)	Query Type: Table Lookup Retrieved clue: matched the table entry for <i>Plate Thickness, in. [mm]</i> > 9–12 [<i>> 225–300</i>] and <i>Hole Diameter, in. [mm]</i> 7/8 [22]. Reasoning: directly retrieved the relevant calibration-hole table and returned the corresponding metric hole diameter.	22 mm
Document-to-Graph RAG (GraphRAG)	Related Query Type: Graph-derived Summary Retrieval Retrieved clue: returned a normalized interval expression, <i>200 to 250, excl, 250 to 300, excl</i> . Reasoning: captured nearby thickness ranges through summary-level graph retrieval, but failed to recover the exact table interval and its associated hole-diameter value.	200 to 250, excl, 250 to 300, excl
KG-based RAG (MindMap)	Query Type: Entity-centered Graph Search Matched entities: <i>case3</i> , <i>225 mm < plate_thickness ≤ 300 mm</i> , and related calibration-hole entities. Key evidence: retrieved the relevant condition <i>225 mm < plate_thickness ≤ 300 mm</i> , but also retrieved competing cases from the ESR/VAR table, such as <i>200 mm < Plate Thickness ≤ 300 mm → Hole Diameter = 10 mm</i> . Reasoning: although the correct condition appeared in the retrieved triples, the graph search did not sufficiently preserve the section/table scope, and the model followed the S9 rule instead.	10 mm
Ontology-based KG-RAG (Ours)	Query Type: Section-aware Rule Matching Matched entities: <i>Hole Diameter = 22 mm</i> , <i>225 mm < plate_thickness ≤ 300 mm</i> , and <i>Ultrasonic Examination Using Flat Bottom Hole Calibration</i> . Key evidence: first resolved the query to the flat-bottom-hole calibration context in Section S8, then matched the condition <i>225 mm < plate_thickness ≤ 300 mm</i> with the consequence <i>hole_diameter = 22 mm</i> . Reasoning: preserved the table scope and condition–consequence structure, avoiding confusion with the ESR/VAR-specific rule in Section S9.	22 mm

Table 17: Case study with condensed retrieval traces. The query requires resolving the correct row identity within the large *Dimensions of Standard Shape Profiles* table in ASTM A6/A6M.

Query: What is the area in mm^2 for the designation metric $W920 \times 588$ in the dimensions of standard shape profiles? Gold: 75,000 mm^2		
Approach	Condensed Retrieval Trace	Prediction
Text-only RAG (gpt-5-mini ada-002)	Query Type: Table Lookup Retrieved clue: matched general phrases such as <i>area in mm^2</i> , <i>designation metric</i> , and <i>dimensions of standard shape profiles</i> . Reasoning: retrieved the broad table context but failed to isolate the exact row for $W920 \times 588$, likely because visually and lexically similar rows such as $W920 \times 586$, $W920 \times 652$, and $W920 \times 970$ appeared nearby.	I don't know
Document-to-Graph RAG (GraphRAG)	Related Query Type: Graph-derived Summary Retrieval Retrieved clue: reported that no $W920 \times 588$ entry was found in the provided table summary. Reasoning: the graph-level summary failed to retain the exact row-level designation, causing the model to treat the answer as unavailable despite the entry existing in the table.	I don't know
KG-based RAG (MindMap)	Query Type: Entity-centered Graph Search Matched entities: <i>designation_metric = $W920 \times 588$</i> , <i>A2. Dimensions of Standard Shape Profiles</i> , <i>case68</i> , and nearby entities such as $W920 \times 586$, $W920 \times 652$, and $W920 \times 970$. Key evidence: retrieved the correct row entity <i>case68</i> and the neighbor fact <i>designation_metric = $W920 \times 588 \rightarrow \text{area} = 75\,000\,\text{mm}^2$</i> . Reasoning: despite retrieving several visually similar designations, the graph search preserved the target designation entity and selected the associated area value from the correct row.	75,000 mm^2
Ontology-based KG-RAG (Ours)	Query Type: Row-aware Table Rule Matching Matched entities: <i>A2. Dimensions of Standard Shape Profiles</i> , <i>designation_metric = $W920 \times 588$</i> , <i>case68</i> , and <i>area = 75 000 mm^2</i> . Key evidence: first resolved the query to the standard-shape-profile table, then matched the exact metric designation $W920 \times 588$ to its row-level consequence <i>area = 75 000 mm^2</i> . Reasoning: preserved the table section, row identity, and attribute–value relation, avoiding confusion with nearby designations such as $W920 \times 586$, $W920 \times 652$, and $W920 \times 970$.	75,000 mm^2

J 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: $CE = C + Mn/6 + (Cr + Mo + V) / 5 + (Ni + Cu) / 15$
Output: <triplet> CE <subj> carbon equivalent <obj> stands_for
 <triplet> CE <subj> $C + Mn/6 + (Cr + Mo + V) / 5 + (Ni + Cu) / 15$ <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: