PoweR4: A Domain Knowledge-Enhanced Rewrite-Retrieve-Rerank-Read Framework For Clause Localization In Power Equipment Supervision

Shuang Lin[1], Jiawei Chen[1], Yan Yang[1], Jian Qian[1], Wenxu Yao[1], and Yuxuan Hu[2*]

[1] *Electric Power Research Institute, State Grid Fujian Electric Power Co., Ltd., Fuzhou, China*
[2] *School of Informatics, Xiamen University, Xiamen, China*
[*] *Corresponding author. E-mail: huyuxuan@stu.xmu.edu.cn*

Received: May 24, 2026; Accepted: June 07, 2026

In the context of power equipment supervision, users often describe fault scenarios in natural language and expect accurate clause-level localization within regulatory manuals. While Large Language Models (LLMs) demonstrate strong generalization capabilities, they face challenges of hallucination and factual inconsistency in domain-specific tasks such as clause localization for power supervision. Although Retrieval-Augmented Generation (RAG) frameworks can alleviate this problem through external documents, existing approaches often focus on isolated enhancements—such as query rewriting or reranking—without addressing the interplay across multiple stages. Thus, we propose PoweR4, a four-stage domain knowledge-enhanced RAG framework tailored for clause localization in the power domain. It consists of: (1) query rewriting using hypothetical document generation, (2) clause retrieval via BM25-based indexing, (3) hierarchical reranking guided by document tree structures, and (4) response generation enriched by knowledge graph–based context construction. We conduct extensive experiments on three real-world datasets, demonstrating that PoweR4 consistently outperforms strong baselines in generation quality. Furthermore, we analyze the impact of query rewriting strategies, document structure depth in reranking, and the degree of knowledge injection, offering a novel paradigm for designing RAG systems in complex industrial domains.

Keywords: Retrieval-Augmented Generation; Power Equipment Supervision; Clause Localization; Knowledge Graph.

http://dx.doi.org/10.6180/jase.202611_34.012

1. Introduction

Large Language Models (LLMs) have demonstrated outstanding application potential in domains such as healthcare [1], finance [2], and law [3]. However, the issues of hallucination [4] and factual inaccuracies [5] in LLMs severely limit their reliability in these domain-specific applications. Retrieval-Augmented Generation [6] (RAG), as a knowledge augmentation technique, has emerged to mitigate these issues. By incorporating external knowledge, RAG enhances the reliability and usability of LLMs.

Despite the ability of RAG to address these problems by introducing external knowledge, generic knowledge enhancement modules or frameworks, such as Rewrite-Retrieve-Read [7] and IRCoT [8], still struggle to adapt to domain-specific downstream tasks [9]. The power industry, characterized by its highly specialized knowledge and complex business logic, presents significant challenges for research in knowledge enhancement technologies. Clause localization, a knowledge-intensive task deeply tied to power industry operations, requires precisely mapping user queries—typically fault descriptions—to specific regulatory clauses.

Existing knowledge enhancement techniques can be categorized into two main classes: (1) RAG-based Methods: A conventional Retrieval-Augmented Generation approach involves retrieving from an external knowledge base, such as a corpus of power system documents, to inject contex-

tual information into LLMs for answer generation. More advanced knowledge graph-augmented methods utilize a power-domain knowledge graph as the retrieval source, either by expanding query semantics through entity neighbor lookups or by optimizing retrieval paths via reasoning over graph relationships; (2) Fine-tuning-based Methods: This approach involves the continued training of a general-purpose, pre-trained LLM using specialized datasets from the electric power domain, thereby embedding domain knowledge directly into the model's parameters.

However, existing RAG-based methods primarily focus on isolated improvements, such as refining the retrieval mechanism or optimizing the generation model, but they often fail to systematically address the interdependencies between the retrieval, selection, and generation stages. Although fine-tuning can enable LLMs to internalize domain-specific patterns, this approach remains highly dependent on large-scale labeled data and exhibits limited transferability to unseen tasks. This highlights a critical challenge in knowledge enhancement for the power domain: isolated enhancements are insufficient, necessitating a joint optimization of the retrieval, selection, and generation processes.

Motivated by the above insights, we propose **PoweR4**—a four-stage joint enhancement framework for technical supervision in the electric power domain, structured as **R**ewrite-**R**etrieve-**R**erank-**R**ead. This framework is designed to jointly optimize the entire pipeline from query understanding to final answer generation, thereby addressing the shortcomings of isolated enhancement approaches. Specifically, our framework comprises the following modules: (1) **Rewrite & Retrieve**: An LLM-based query rewriting module that expands query semantics by generating hypothetical documents, improving the recall of relevant documents from the knowledge base; (2) **Rerank**: A hierarchical reranking module that parses clauses into a document tree and leverages multi-level contextual semantics to reorder the preliminary retrieval results, enhancing clause selection precision; (3) **Read**: A reading module that injects a lightweight, prebuilt power-domain knowledge graph during generation, supplying targeted entity definitions to fill domain knowledge gaps and boost clause localization accuracy. We further explore the following questions: Q1: How do different query rewriting strategies affect domain-specific retrieval performance? Q2: What is the impact of document tree abstraction levels on reranking effectiveness? Q3: To what extent does the granularity of domain knowledge improve generation quality?

Our key contributions are:

- We introduce PoweR4, a novel four-stage RAG framework that integrates query rewriting, hierarchical reranking, and targeted knowledge injection to achieve superior performance compared to existing baselines in power-domain clause localization.
- Extensive experiments on three real-world datasets demonstrate that, compared to standard RAG baselines, PoweR4 consistently improves both retrieval accuracy and answer fidelity.
- We perform a comprehensive analysis of various knowledge augmentation techniques at each stage, offering insights into their optimal strategies and limitations in specialized domains, thereby paving the way for future domain-specific RAG system design.

2. Related work

2.1. General-purpose Retrieval-Augmented Generation

RAG enhances the accuracy and factuality of LLMs by integrating external knowledge bases. Early RAG paradigms followed a straightforward "retrieve-then-read" process, but their performance was heavily constrained by retrieval quality. In practice, irrelevant or noisy context often degraded the quality of generated outputs.

To overcome these limitations, research has gradually evolved into Advanced RAG, the core idea of which is to add optimization stages before and after retrieval. Pre-retrieval optimization primarily focuses on query rewriting, with methods like Rewrite-Retrieve-Read [7] and HyDE [10] aiming to improve recall by generating better queries. Post-retrieval optimization, on the other hand, centers on reranking the retrieved results to filter for the most relevant and valuable context. Concurrently, the emergence of Modular RAG, exemplified by frameworks such as IRCoT [8], addresses more complex multi-hop problems through iterative retrieval and reasoning.

Despite the continuous evolution of these general-purpose RAG frameworks, they predominantly focus on optimizing a single stage, such as querying, retrieval, or reranking. This isolated optimization approach overlooks the synergistic effects among different stages, rendering it inadequate for tackling complex real-world challenges, especially in scenarios requiring deep domain knowledge.

2.2. Domain-specific Knowledge Augmentation

Adapting general pretrained language models to specialized domains is an active area of research, with two dominant approaches: knowledge graph-based augmentation and fine-tuning. In the power domain, these strategies have been applied to improve performance in expert tasks such as fault diagnosis and clause localization.

Knowledge Graph (KG)-based augmentation. This method retrieves structured factual knowledge from external knowledge graphs during inference, serving as non-parametric inputs to language models. Common techniques include serializing KG triples into prompt text (e.g., KG-CoT [11]) or constructing document-level graph structures for retrieval (e.g., GraphRAG [12]). TCE-KG [13] leverages a spatiotemporal event graph that integrates historical fault tickets and causal relationships within the distribution network, enabling structured reasoning for downstream tasks. SCER [14] adopts a KG-based evidence chain fusion strategy to improve the accuracy and reliability of answers in power maintenance QA tasks.

Fine-tuning-based approaches. These methods continue training pretrained LLMs on domain-specific data to internalize expert knowledge into model parameters. The general principle is established by Domain-Adaptive Pretraining [15], which performs unsupervised pretraining on domain-specific corpora followed by supervised task-specific fine-tuning—yielding consistent improvements in specialized tasks. In the power domain, this paradigm has been further extended. LLM4DistReconfig [16] fine-tunes an LLM on datasets that include power grid topologies and energy loss parameters, enabling direct prediction of optimal network reconfiguration strategies. P2FT [17] adopts a two-stage training process combining domain-adaptive pretraining with generation control strategies to produce well-structured, regulation-compliant maintenance reports.

In summary, existing knowledge augmentation methods typically focus on isolated optimization: general RAG frameworks target modular improvements, while domain-specific approaches struggle with scalability, knowledge freshness, and multi-stage coordination. This highlights the need for a unified solution that jointly optimizes retrieval, reranking, and generation, and integrates diverse forms of domain knowledge—a gap addressed by our proposed **PoweR4** framework.

3. The PoweR4 Framework

In this section, we present the proposed PoweR4 framework, which jointly optimizes three key stages within the retrieval-augmented generation pipeline: query understanding, document selection, and clause generation. As shown in Figure 1, PoweR4 comprises four core modules: Rewrite, Retrieve, Rerank, and Read.

- **Rewrite:** Generate structured hypothetical documents to clarify user intent and dynamically enrich query semantics.
- **Retrieve:** Perform targeted initial retrieval using the semantically enhanced query to recall highly relevant clause candidates.
- **Rerank:** Apply inherent document hierarchy to logically reorder and refine the retrieved clause candidates.
- **Read:** Inject structured knowledge from a domain knowledge graph to enhance the generation context and guide final clause output.

3.1. Query Rewriting

In the query rewriting stage, we draw on the HyDE (Hypothetical Document Embeddings) approach, using prompt engineering to guide LLMs to generate a structured pseudo-document that semantically augments the user's original query. The pseudo-document follows a two-paragraph format—"Key Risk" and "Recommended Action"—where the first paragraph summarizes the equipment risk or scenario implied by the query, and the second paragraph proposes corresponding standard remediation steps. This structured format substantially improves semantic coverage and retrieval friendliness.

To remove formatting artifacts, a regularization function κ filters out annotation tokens. To prevent dilution of the original query's term frequency, the pseudo-document is concatenated with the original query in a 5:1 ratio, yielding the enhanced query q_{enhanced} which is then passed to the retrieval module:

$$q_{\text{enhanced}} = \underbrace{q \parallel q \parallel \cdots \parallel q}_{5} \parallel \kappa(d_{\text{hyp}}) \tag{1}$$

where q denotes the original query, κ denotes the filtering function, and $\parallel$ denotes the concatenation operator.

3.2. Hierarchical Reranking

The retrieval module is built using the Pyserini framework, which indexes clause-level units from the regulatory corpus C. Given the enhanced query q_{enhanced}, the system first recalls a candidate set D_K using BM25 scoring:

$$D_K = \underset{d \in C,\ |D_K|=K}{\arg\text{topK}}\ S_{\text{BM25}}(q_{\text{enhanced}}, d) \tag{2}$$

where S_{BM25} denotes the BM25 relevance scoring function and K the fixed recall size. To address topic fragmentation within D_K, we introduce a hierarchical reranking mechanism based on the document tree T. For each candidate $d_i \in D_K$, we define a topic extraction function $\phi(d_i, L)$ that parses the clause's section ID and returns its level-L prefix:

$$t_i = \phi(d_i, L), \quad \mathcal{T} = \{t_i \mid d_i \in D_K\}, \quad L \in \{1, 2, 3\} \tag{3}$$

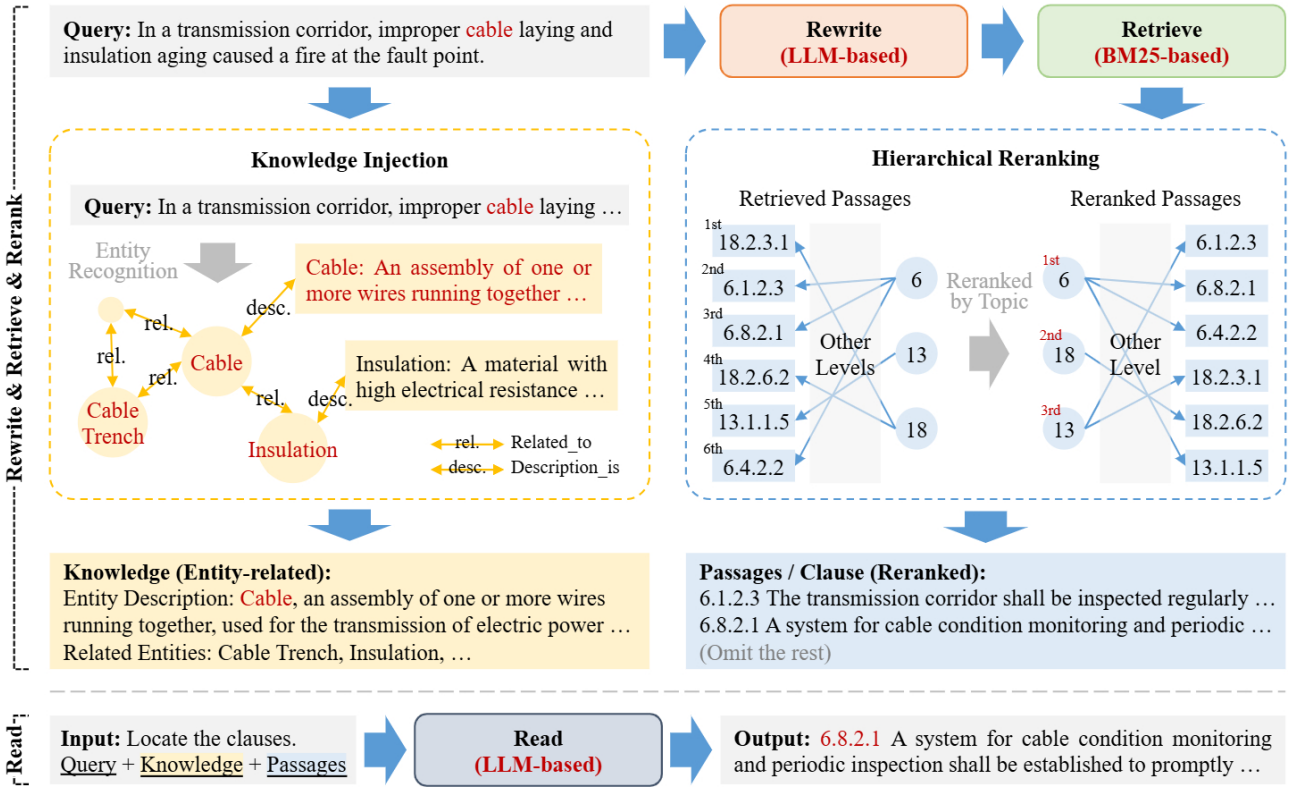

Fig. 1. The overall architecture of the PoweR4 framework.

For a clause with hierarchical index $s_i = a_1.a_2.a_3$, the level-L topic prefix t_i is computed by retaining the first L segments of s_i:

$$t_i = \phi(d_i, L) = a_1.a_2.\ldots.a_L \tag{4}$$

For example, given a clause indexed as $s_i = 6.1.2$, then $\phi(d_i, 1) = 6$ and $\phi(d_i, 2) = 6.1$. Empirical results suggest that setting $L = 1$ offers the best trade-off between semantic abstraction and ranking performance.

Next, an LLM $\mathcal{L}$ semantically ranks the candidate topics $\mathcal{T}$ to produce an optimal permutation π^* that maximizes relevance to the original query q:

$$\pi^* = \mathcal{L}(q, \{(t, T(t)) \mid t \in \mathcal{T}\}) \tag{5}$$

where $T(t)$ is the textual description of topic number t. The final result $\pi^* = (t_{(1)}, t_{(2)}, \ldots, t_{(|\mathcal{T}|)})$ is the sequence of topics arranged from high to low in terms of relevance. The original candidate set D_K is divided into multiple subsets G_t according to their topics $t \in T$ on level L. We then group the clauses by topic:

$$G_t = \{d \in D_K \mid \phi(d, L) = t\} \tag{6}$$

The final reranked list is:

$$D'_{\text{reranked}} = G_{t_{(1)}} \parallel G_{t_{(2)}} \parallel \cdots \parallel G_{t_{(\mathcal{T})}} \parallel G_{\text{unmatched}} \tag{7}$$

where $\parallel$ denotes concatenation and $G_{\text{unmatched}}$ contains clauses unmatched to any topic, preserving their original BM25 order.

This hierarchical strategy addresses the problem by separating the ranking process into two layers: a semantic ranking of topics at the inter-topic level, and a relevance ranking based on term frequency at the intra-topic level. This approach effectively alleviates the thematic incoherence of the search results.

3.3. Knowledge Injection

In the final generation stage, we integrate a lightweight power-domain knowledge graph $\mathcal{G} = (E, R)$, where E denotes the set of entities and $R \subseteq E \times E$ the set of relations. This lightweight knowledge graph is constructed by extracting structured entity relationship data from official domain documents. Using Neo4j for storage and indexing, the raw data undergoes strict preprocessing to eliminate empty values and self-loops. Ultimately, the graph comprises 8,463 entity nodes and 15,716 relations.

First, we extract the entity set E_q from the original query q using an NER(Named Entity Recognition) prompt template P_{NER} with the LLM $\mathcal{L}$. This template guides the model $\mathcal{L}$ to play the role of a domain expert and extract entities

according to preset power domain categories:

$$E_q = \mathcal{L}(P_{\text{NER}}(q)) \tag{8}$$

For each entity $e_i \in E_q$, we retrieve its description $\text{Desc}(e_i)$ and one-hot neighbor entities $\text{Neighbors}(e_i)$ from $\mathcal{G}$. These knowledge snippets are integrated into a structured supplementary context $C_{\mathcal{G}}$ by the formatting function $\mathcal{F}_{KG}$:

$$C_{\mathcal{G}} = \mathcal{F}_{\text{KG}}(\{(e_i, \text{Desc}(e_i), \text{Neighbors}(e_i)) \mid e_i \in E_q\}) \tag{9}$$

Finally, we concatenate the original query q, the KG context $C_{\mathcal{G}}$ and the reranked clause list $D'_{reranked}$ to form the final prompt:

$$P_{\text{final}} = q \parallel C_{\mathcal{G}} \parallel \text{Format}(D'_{\text{reranked}}) \tag{10}$$

The model then generates the precise clause localization answer:

$$a_{\text{final}} = \mathcal{L}(P_{\text{final}}) \tag{11}$$

4. Experiments

In this section, we evaluate our proposed PoweR4's effectiveness, presenting experimental setup, datasets, baselines, and results, followed by an ablation study and in-depth analysis of domain-specific knowledge augmentation strategies.

4.1. Dataset and Evaluation

Data Description. We evaluate the effectiveness of PoweR4 on three real-world datasets collected from a national power research institute, spanning technical supervision tasks recorded between 2021 and 2023. The datasets are as follows:

- **Supervision-2021**: 117 annotated query-clause pairs.
- **Supervision-2022**: 67 annotated query-clause pairs.
- **Supervision-2023**: 122 annotated query-clause pairs.

Each query is a natural language description of a fault or operational condition, and each clause is the corresponding regulation from official structured documents with section numbers and contents.

Evaluation. To evaluate the accuracy of clause localization, we adopt Exact Match (EM) as the primary metric. A prediction is considered correct if the generated clause ID exactly matches any of the gold-standard references. In cases where multiple valid answers exist, a match with any one of them is counted as a success. The EM score is defined as the proportion of correctly matched samples over the total number of test queries. Unlike general text generation tasks that rely on N-gram overlap metrics (such as BLEU or ROUGE), clause localization requires absolute strictness in regulatory compliance. Therefore, we exclusively adopt EM to evaluate whether the model outputs the exact regulatory source, prioritizing factual accuracy over textual fluency.

4.2. Baseline and Experimental Setting

To validate the effectiveness of the PoweR4 framework, we construct baseline systems based on six open-source LLMs with diverse parameter scales and architectures: Llama2-7B-chat [18], Llama3-8B-instruct, Llama2-13B-chat [18], Qwen2-7B-instruct [19], Qwen2.5-7B-instruct, and Qwen2.5-14B-instruct. On top of these models, we implement a Naive RAG baseline, which follows a basic retrieval-then-generate pipeline: given a raw user query, the system retrieves Top-K documents via BM25 and feeds the concatenated query and passages directly into the LLM without query rewriting, reranking, or knowledge injection.

For the PoweR4 framework, we use Qwen2.5-14B-instruct as the base model due to its strong performance in Chinese technical domains. Specifically, all experiments were conducted on a server equipped with four NVIDIA GeForce RTX 3090 GPUs (24GB VRAM each). The core software environment relies on PyTorch 2.6.0 and Pyserini 0.40.0. During the retrieval phase, we vary the number of retrieved documents ($K = 3, 5, 10$) to assess the impact of retrieval size on overall system performance.

4.3. Main Results

We compare the proposed PoweR4 framework against all Naive RAG baselines. The experimental results are presented in Table 1.

Results show that PoweR4 achieves the best or near-best performance across all datasets, with an overall EM of 67.0%, significantly outperforming all baselines. This confirms the effectiveness of the four-stage optimization strategy. Notably, Qwen models consistently outperformed Llama variants, likely due to better Chinese language and domain comprehension.

We further analyzed two key factors:

Top-K setting. At K=3, retrieval misses relevant context; at K=10, noise harms reasoning, especially for baselines lacking reranking. PoweR4 achieves optimal balance at K=5.

Model size. Larger models do not guarantee better performance. For example, Llama2-13B underperforms Qwen2-7B and Llama3-8B on Supervision-2021. Even Qwen2.5-14B is slightly outperformed by Qwen2.5-7B in certain cases. This suggests that structural guidance is more

Table 1. Exact Match Results on Clause Localization (Top-K Retrieval with $K = 3, 5, 10$)

Methods	Sup-2021			Sup-2022			Sup-2023			Overall		
	N = 3	N = 5	N = 10	N = 3	N = 5	N = 10	N = 3	N = 5	N = 10	N = 3	N = 5	N = 10
Llama2-7B-chat	0.4017	0.2991	0.0341	0.4098	0.3114	0.0000	0.3606	0.3442	0.0491	0.3867	0.3442	0.0333
Llama3-8B-instruct	0.5641	0.4700	0.4700	0.5081	0.4754	0.4590	0.4016	0.3688	0.3688	0.4867	0.4300	0.4267
Llama2-13B-chat	0.0598	0.1282	0.0854	0.1475	0.0655	0.1147	0.1065	0.1229	0.1147	0.0967	0.1133	0.1036
Qwen2-7B-instruct	0.6153	0.6410	0.6324	0.6393	0.6229	0.6885	0.5983	0.6147	0.5819	0.6133	0.6267	0.6233
Qwen2.5-7B-instruct	0.5982	0.6752	0.5641	0.5737	0.5409	0.5409	0.5409	0.5327	0.4836	0.5700	0.5733	0.5267
Qwen2.5-14B-instruct	0.6410	0.6667	0.6239	0.6393	0.6065	0.6229	0.5901	0.5819	0.5901	0.6200	0.6200	0.6100
Ours (PoweR4)	0.6324	0.7094	0.6495	0.6393	0.7049	0.7377	0.5983	0.6147	0.5819	0.6200	0.6700	0.6400

Table 2. Ablation study of our proposed PoweR4 framework. We report the Exact Match (EM) rate after removing each core module. The asterisk (*) denotes a statistically significant difference ($p < 0.05$) compared to the full model, as determined by McNemar's test.

Configuration	EM	Δ EM (vs. Full Model)	McNemar's χ^2 / p-value
PoweR4 (w/o Rewrite)	0.6557	-0.0492	8.76 / 0.0031*
PoweR4 (w/o Knowledge-graph)	0.6885	-0.0164	4.12 / 0.0425*
PoweR4 (w/o Tree-sort)	0.6721	-0.0328	6.03 / 0.0141*
PoweR4 (Full Model)	0.7049	-	-

critical than model scale alone.

4.4. Ablation Study

To quantify the independent contribution of each enhancement module in the PoweR4 framework, we conducted a series of ablation experiments on the overall dataset. Based on the complete PoweR4 framework, we sequentially removed the query rewriting (w/o Rewrite), hierarchical reranking (w/o Tree-sort), and knowledge graph injection (w/o Knowledge-graph) modules, and tested their performance under the setting of K=5. The results are presented in Table 2.

From Table 2, we can obtain the following key observations: (1) Query Rewriting (Rewrite): Removing this module led to a 4.9% absolute drop in EM (from 0.7049 to 0.6557), with McNemar's test yielding $p = 0.0031$, which suggests that hypothetical document generation plays a critical role in resolving ambiguous or underspecified queries. (2) Knowledge Graph Injection: Its removal caused a 1.6% decrease in EM, with $p = 0.0425$, confirming that disambiguating technical terms and leveraging entity associations significantly contribute to localization accuracy. (3) Hierarchical Reranking (Tree-sort): This removal resulted in a 3.3% drop in EM, with $p = 0.0141$, indicating that topic-level aggregation plays a significant role in effectively reducing cross-topic retrieval noise.

4.5. Analysis and Discussion

To delve into the practical effectiveness of diverse knowledge augmentation strategies within specialized domain-specific scenarios, we systematically structure our in-depth discussion around four pivotal central research questions.

4.5.1. *Q1: Effect of Query Rewriting Strategies in Domain-specific Retrieval*

To investigate the impact of different query rewriting strategies on domain-specific retrieval performance, we compared three primary methods across four backbone models: (1) Hypothetical Document Generation (HyDE); (2) Keyword Extraction and Expansion via LLMs; (3) Direct LLM-based Natural Language Rewriting. We evaluated these strategies across four base models from two perspectives: recall rate (Hit Rate @K) and final generation accuracy (EM). The experimental results are shown in Table 3 and Table 4.

Results indicate that retrieval is highly sensitive to query formulation. HyDE improves recall by enriching keyword coverage through structured context, but risks introducing semantic noise. In contrast, simpler keyword-based methods achieve more stable precision despite lower recall.

We conclude that no single query rewriting strategy is universally optimal across all scenarios. An effective approach must carefully balance recall-driven keyword coverage and semantic clarity, with pseudo-document generation emerging as a promising compromise tailored to specific task requirements.

4.5.2. *Q2: Effect of Document Tree Granularity in Domain-specific Reranking*

Document tree structure enables LLMs to rerank clauses by topic relevance. We evaluated different topic levels ($L = 1, 2, 3$) using Qwen2.5-14B on the Overall dataset. The experimental results are shown in table 5.

Performance peaked at $L = 1$, where top-level topics

Table 3. Comparison of different query rewriting strategies on retrieval recall (Hit Rate@K). The best result for each model is highlighted in bold.

Model	Rewrite-Method	Hit Rate@5	Hit Rate@10
Qwen2-7B	LLM-Rewrite	0.7333	0.8133
	Keyword-Rewrite	0.7666	0.8300
	HyDE	**0.8133**	**0.8566**
Qwen2.5-7B	LLM-Rewrite	0.7666	0.8266
	Keyword-Rewrite	0.7666	0.8333
	HyDE	**0.8000**	**0.8666**
Qwen2.5-14B	LLM-Rewrite	0.7700	0.8166
	Keyword-Rewrite	0.7433	0.7933
	HyDE	**0.8066**	**0.8566**
Llama3-8B	LLM-Rewrite	0.7566	0.8100
	Keyword-Rewrite	0.7200	0.7800
	HyDE	**0.8133**	**0.8500**

Table 4. Comparison of different query rewriting strategies on final generation accuracy (Exact Match). "None" indicates the baseline without any query rewriting. The best result for each model is highlighted in bold.

Model	Rewrite-Method	N=5	N=10
Qwen2-7B	None	**0.6267**	**0.6233**
	LLM-Rewrite	0.5633	0.5467
	Keyword-Rewrite	0.5800	0.6133
	HyDE	0.5967	0.6100
Qwen2.5-7B	None	0.5900	0.5267
	LLM-Rewrite	0.5466	0.5267
	Keyword-Rewrite	0.5567	0.5133
	HyDE	**0.6100**	**0.5500**
Qwen2.5-14B	None	0.6200	0.6100
	LLM-Rewrite	**0.6533**	0.6467
	Keyword-Rewrite	0.5967	0.6400
	HyDE	0.6467	**0.6567**
Llama3-8B	None	0.4300	0.4267
	LLM-Rewrite	**0.5000**	0.4867
	Keyword-Rewrite	0.4933	0.4767
	HyDE	0.4967	**0.4867**

offer clear semantic abstraction and distinct separability, enabling efficient alignment between user queries and high-level thematic categories (e.g., "Chapter 6: Transmission Line Safety"). In contrast, overly fine-grained headings at $L = 3$ often overlap with clause content and introduce semantic redundancy, which ultimately hinders ranking accuracy.

We conclude that moderate topic abstraction enhances reasoning, while over-segmentation introduces semantic noise. Using $L = 1$ is a practical and effective default for technical regulation corpora.

4.5.3. Q3: Effect of Knowledge Graph Injection Strategies in Domain-specific Generation

We evaluated five knowledge graph (KG) injection strategies involving various combinations of entities, descriptions, and neighbors.

Table 6 shows that the most effective setup included entity definitions and neighbor entities only. This configuration offers targeted context with minimal overhead: definitions resolve ambiguity, while neighbor names provide lightweight relational cues. In contrast, including neighbor descriptions increased content but diluted critical signals, reducing performance.

Thus, effective KG augmentation emphasizes conciseness and relevance. Optimal designs should provide minimal, well-targeted context around the target entity to support reasoning and accurate clause generation.

4.5.4. Q4: What are the primary boundaries and failure modes of PoweR4?

Despite the overall improvements, PoweR4 failed to precisely localize the clause in approximately 33% of the cases (Overall EM 0.6700). A qualitative error analysis of these failure cases reveals two primary causes. First, extreme ambiguity in user queries, such as the omission of specific equipment names or fault phenomena, makes retrieval highly challenging, even with hypothetical document rewriting. Second, the framework occasionally encounters out-of-vocabulary entities that are not yet covered by our lightweight, pre-constructed knowledge graph. Addressing these boundary cases remains a key direction for our future work.

Table 5. Impact of Different Document Tree Hierarchy Levels on Model Performance

N=5, Model	tree level = 1	tree level = 2	tree level = 3	Naïve
Qwen2.5-14B	0.6400	0.6333	0.6300	0.6200

Table 6. Comparison of different Knowledge Graph (KG) injection strategies on EM score. The headers denote the combination of injected information: 'E' (the entity itself), 'D' (its description), 'N' (the names of its neighbors), and 'ND' (the descriptions of its neighbors). 'All Components' includes all four types of information.

N=5, Model	E+D+N	N+ND	E+N+D	All Components	Naïve
Qwen2-7B	**0.6200**	0.6133	0.6167	0.6167	0.6267
Qwen2.5-7B	0.5533	0.5633	0.5300	0.5500	**0.5733**
Qwen2.5-14B	**0.6233**	0.6200	0.6200	0.6167	0.6200
Llama3-8B	**0.5400**	0.5300	0.5367	0.5367	0.4300

5. Conclusion

We propose PoweR4, a domain enhanced RAG framework optimizing three components of the RAG pipeline: query rewriting, hierarchical reranking based on document trees, and knowledge graph injection. Extensive experiments on real-world datasets demonstrate that PoweR4 significantly outperforms Naive RAG baselines, validating the effectiveness of our joint optimization strategy. Further analyses reveal the boundaries and best practices of knowledge augmentation techniques in domain-specific scenarios. Overall, this study not only provides a practical solution for intelligent clause localization in power supervision but also offers a paradigm for designing RAG systems in other specialized domains—particularly in balancing the breadth and precision of domain knowledge enhancement.

Acknowledgment

This work is supported by the Science and Technology Project of State Grid Fujian Electric Power Co., Ltd., specifically the project titled "Research and Application of Key Technologies for Equipment Technical Supervision Based on Retrieval Augmentation and Image-Text Feature Fusion (2025-2026)" (Project No. 521B0425000K) from State Grid Fujian Electric Power Research Institute. This work is solely funded by the above-mentioned project, with no other competing financial interests.

References

[1] A. J. Thirunavukarasu, D. S. J. Ting, K. Elangovan, L. Gutierrez, T. F. Tan, and D. S. W. Ting, (2023) *"Large language models in medicine"* **Nature medicine 29**(8): 1930–1940. DOI: 10.1038/s41591-023-02448-8.

[2] Y. Li, S. Wang, H. Ding, and H. Chen. "Large language models in finance: A survey". In: *Proceedings of the fourth ACM international conference on AI in finance*. 2023, 374–382. DOI: 10.1145/3604237.3626869.

[3] J. Lai, W. Gan, J. Wu, Z. Qi, and P. S. Yu, (2024) *"Large language models in law: A survey"* **AI Open**: DOI: 10.1016/j.aiopen.2024.09.002.

[4] L. Huang, W. Yu, W. Ma, W. Zhong, Z. Feng, H. Wang, Q. Chen, W. Peng, X. Feng, B. Qin, et al., (2025) *"A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions"* **ACM Transactions on Information Systems *43***(2): 1–55. DOI: 10.1145/3703155.

[5] C. Wang, X. Liu, Y. Yue, X. Tang, T. Zhang, C. Jiayang, Y. Yao, W. Gao, X. Hu, Z. Qi, et al., (2023) *"Survey on factuality in large language models: Knowledge, retrieval and domain-specificity"* **arXiv preprint arXiv:2310.07521**: DOI: 10.48550/arXiv.2310.07521.

[6] Y. Gao, Y. Xiong, X. Gao, K. Jia, J. Pan, Y. Bi, Y. Dai, J. Sun, H. Wang, and H. Wang, (2023) *"Retrieval-augmented generation for large language models: A survey"* **arXiv preprint arXiv:2312.10997 2**(1): DOI: 10.48550/arXiv.2312.10997.

[7] X. Ma, Y. Gong, P. He, H. Zhao, and N. Duan. "Query Rewriting in Retrieval-Augmented Large Language Models". In: *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. Ed. by H. Bouamor, J. Pino, and K. Bali. Singapore: Association for Computational Linguistics, 2023, 5303–5315. DOI: 10.18653/v1/2023.emnlp-main.322.

[8] H. Trivedi, N. Balasubramanian, T. Khot, and A. Sabharwal, (2022) *"Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step*

questions" **arXiv preprint arXiv:2212.10509**: DOI: 10.48550/arXiv.2212.10509.

[9] N. Kandpal, H. Deng, A. Roberts, E. Wallace, and C. Raffel. "Large language models struggle to learn long-tail knowledge". In: *International Conference on Machine Learning*. PMLR. 2023, 15696–15707. DOI: 10.48550/arXiv.2211.08411.

[10] L. Gao, X. Ma, J. Lin, and J. Callan. "Precise zero-shot dense retrieval without relevance labels". In: *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 2023, 1762–1777. DOI: 10.18653/v1/2023.acl-long.99.

[11] R. Zhao, F. Zhao, L. Wang, X. Wang, and G. Xu. "Kg-cot: Chain-of-thought prompting of large language models over knowledge graphs for knowledge-aware question answering". In: *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence (IJCAI-24)*. International Joint Conferences on Artificial Intelligence. 2024, 6642–6650. DOI: 10.24963/ijcai.2024/734.

[12] D. Edge, H. Trinh, N. Cheng, J. Bradley, A. Chao, A. Mody, S. Truitt, D. Metropolitansky, R. O. Ness, and J. Larson, (2024) *"From local to global: A graph rag approach to query-focused summarization"* **arXiv preprint arXiv:2404.16130**: DOI: 10.48550/arXiv.2404.16130.

[13] F. Liao, J. Huang, Q. Liu, X. Peng, B. Liu, X. Wu, and J. Qian. "TCEKG: A Temporal and Causal Event Knowledge Graph for Power Distribution Network Fault Diagnosis". In: *International Conference on Intelligent Computing*. Springer. 2024, 505–517. DOI: 10.1007/978-981-97-5615-5_41.

[14] J. Zhao, Z. Ma, H. Zhao, X. Zhang, Q. Liu, and C. Zhang. "Self-consistency, Extract and Rectify: Knowledge Graph Enhance Large Language Model for Electric Power Question Answering". In: *International Conference on Intelligent Computing*. Springer. 2024, 493–504. DOI: 10.1007/978-981-97-5615-5_40.

[15] S. Gururangan, A. Marasović, S. Swayamdipta, K. Lo, I. Beltagy, D. Downey, and N. A. Smith, (2020) *"Don't stop pretraining: Adapt language models to domains and tasks"* **arXiv preprint arXiv:2004.10964**: DOI: 10.18653/v1/2020.acl-main.740.

[16] P. Christou, M. Z. Islam, Y. Lin, and J. Xiong, (2025) *"LLM4DistReconfig: A Fine-tuned Large Language Model for Power Distribution Network Reconfiguration"* **arXiv preprint arXiv:2501.14960**: DOI: 10.18653/v1/2025.naacl-long.208.

[17] Y. Yang, C. Li, B. Zhu, W. Zheng, F. Zhang, and Z. Li, (2024) *"Enhancing domain-specific text generation for power grid maintenance with P2FT"* **Scientific Reports *14***(1): 26771. DOI: 10.1038/s41598-024-78078-y.

[18] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale, et al., (2023) *"Llama 2: Open foundation and fine-tuned chat models"* **arXiv preprint arXiv:2307.09288**: DOI: 10.48550/arXiv.2307.09288.

[19] Q. Team, (2024) *"Qwen2 technical report"* **arXiv preprint arXiv:2412.15115**: DOI: 10.48550/arXiv.2412.15115.