ElecGraph-RAG: A Star-Shaped Graph Modeling and Efficient Hierarchical Retrieval Framework for Electronic Datasheets

Anonymous ACL submission

Abstract

Existing graph-based retrieval-augmented generation (RAG) methods often fail to capture the device-centric, star-shaped ontology of electronic datasheets, leading to topological mismatches and entity–attribute misattribution in technical question answering. We propose **ElecGraph-RAG**, an ontology-aligned hierarchical RAG framework explicitly designed for device-centric knowledge organization, which introduces ontology-aware semantic abstraction as an inductive bias for retrieval. The framework performs coarse-to-fine retrieval by first localizing relevant high-level semantic abstractions and then extracting fine-grained attribute-level evidence. We also introduce **ElecGraph-QA**, a benchmark of 200 complex factual and comparative queries derived from real-world industrial specifications. Experimental results show that ElecGraph-RAG achieves **94.28%** and **85.00%** accuracy on factual and comparative queries, respectively, while reducing token consumption by **76.3%** compared to state-of-the-art graph-based baselines. These results demonstrate the effectiveness of ontology-aligned hierarchical abstraction for efficient domain-specific RAG.

1 Introduction

Retrieval-Augmented Generation (RAG) enhances LLMs with external knowledge but struggles with the strict entity–attribute bindings in technical documentation (Es et al., 2024). Specifically, text-centric retrievers often fail to distinguish the operating voltage of distinct chips in the same datasheet corpus(Luan et al., 2021). This causes semantic ambiguity and hallucination—challenges that text-centric or generic graph-based retrieval methods fail to address(Cossio, 2025).

We identify a critical **topological mismatch** as a dominant failure mode. Datasheet knowledge is inherently asymmetric: each device serves as a semantic center for numerous attributes, forming a "star-shaped" ontology. Mainstream Graph RAG (Edge et al., 2024), however, relies on density-based community detection that ignores this semantic centering. By merging attributes from distinct devices into coarse clusters, these methods cause misattribution rather than simple information loss, where attributes are mapped to the wrong device.

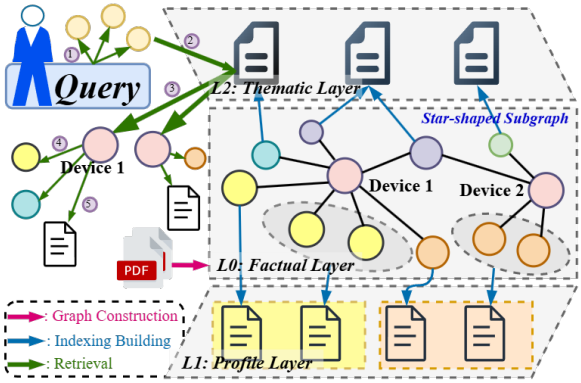


Figure 1: Workflow of ElecGraph-RAG. The framework employs a dual-path strategy (blue arrows) to construct independent local (L1) and global (L2) indices from the star-shaped graph (L0). During retrieval (green arrows), L2 serves as a cognitive map to prune the search space, guiding precise evidence extraction from targeted devices

We propose that faithful retrieval requires ontology-aligned abstraction. Architectures must identify semantic anchors (target devices) before parsing fine-grained details. To this end, we introduce **ElecGraph-RAG**. It employs a three-level hierarchical semantic index tailored to the domain's star-shaped topology, which includes atomic graphs (L0), device profiles (L1) and thematic summaries (L2). Unlike linear hierarchies, we implement a distinct dual-path abstraction strategy that simultaneously derives local device profiles and global thematic clusters from the atomic graph, ensuring that macro-level navigation does not obscure micro-level structural details.

Crucially, this architecture enables a coarse-to-

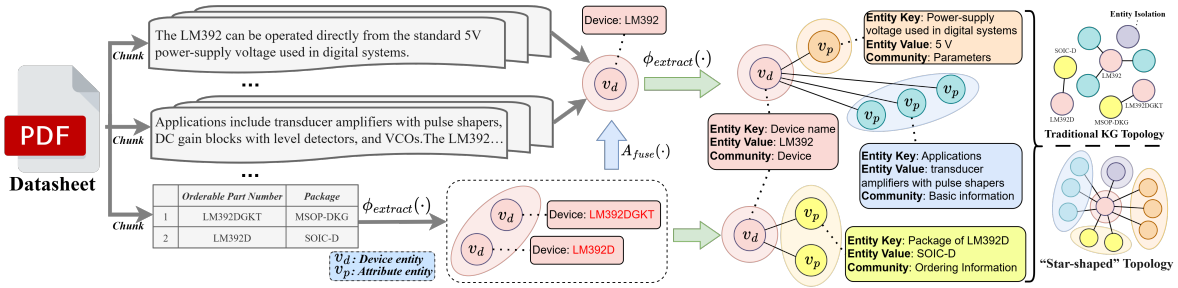


Figure 2: The device-centric knowledge graph modeling process in ElecGraph-RAG. The framework first extracts and aligns device entities from all text chunks to construct a normalized device layer; then, using these devices as anchors, it reprocesses the text to extract and align related attribute entities, ultimately forming a unified knowledge graph with a "star-shaped" topology.

fine hierarchical retrieval strategy with early-stage semantic pruning (Peng et al., 2025). Top-layer summaries act as a cognitive navigation map, mimicking human search by identifying relevant semantic regions before parsing details. This minimizes irrelevant processing while ensuring high recall for multi-hop queries.

Our contributions are as follows:

- We identify **topological mismatch** as a key cause of retrieval unfaithfulness in domain-specific RAG, and empirically demonstrate that aligning retrieval architectures with domain ontology significantly improves entity–attribute faithfulness in technical QA.
- We propose **ElecGraph-RAG**, a hierarchical RAG framework that integrates device-centric star-shaped graph modeling with a dual-path, ontology-aligned semantic abstraction mechanism, enabling coarse-to-fine focused retrieval that performs early-stage semantic pruning, precise device anchoring, and efficient extraction of fine-grained factual evidence.
- We introduce **ElecGraph-QA**, a domain-specific benchmark derived from real-world electronic datasheets, and show through comprehensive experiments that ontology-aligned hierarchical retrieval improves accuracy while reducing token consumption by up to 76.3%.

2 Related Work

Retrieval-Augmented Generation (RAG) mitigates LLM hallucinations via external knowledge (Es et al., 2024). Standard methods, or Naive RAG, rely on dense vector retrieval (Karpukhin et al., 2020). While effective for general QA, these text-centric pipelines implicitly assume that semantic similarity is sufficient for relevance matching. However, this assumption breaks down in technical domains such as electronic datasheets, where correct answers require preserving strict entity–attribute bindings (e.g., distinguishing parameters of closely related device models), resulting in fragmentation (Liu et al., 2023).

Recent works integrate knowledge graphs (KGs) to add structural context. Methods like GraphRAG (Edge et al., 2024) use community detection for global summarization. However, they suffer from a *topological mismatch* in electronics, as generic clustering fails to isolate specific device models based on semantic centrality. Subgraph methods like LightRAG (Guo et al., 2024) retrieve local paths but lack a unified macro-to-micro view. Recent bottom-up approaches (Zhang et al., 2025b) still rely on implicit topology rather than explicit schemata(Chepurova et al., 2024).

For efficiency, hierarchical approaches like ArchRAG (Wang et al., 2025) and HiRAG (Jiao et al., 2025) filter irrelevant data. Yet, they rely on vector clustering or general semantics, lacking domain alignment. In contrast, ElecGraph-RAG constructs a hierarchy based on the explicit "Device-Attribute" ontology. This provides an interpretable cognitive navigation map that ensures both precision and token efficiency.

3 The ElecGraph-RAG Architecture

To address the challenges of misattribution and topological mismatch, ElecGraph-RAG departs from generic graph clustering methods. Instead, we explicitly model device-centric star-shaped graphs and construct a hierarchical semantic index aligned with domain ontology. This design rationale enables a two-phase retrieval process: early-stage pruning at higher abstraction levels to lock onto

semantic anchors, followed by focused retrieval of fine-grained factual evidence. As illustrated in Figure 1, the workflow comprises three stages: device-centric graph construction, hierarchical semantic indexing, and cognitive-inspired retrieval.

3.1 Knowledge Graph Construction

The ElecGraph-RAG framework first leverages an LLM to transform unstructured electronic device datasheets into a domain-specific structured knowledge graph $G = (V, E)$. The entire construction process can be abstractly expressed as:

$$G = (V, E) = A_{\text{fuse}}\left(\bigcup_{t_i \in T} \phi_{\text{extract}}(t_i)\right) \quad (1)$$

where T is the set of text chunks parsed from the original PDF documents, and $\phi_{\text{extract}}(\cdot)$ and $A_{\text{fuse}}(\cdot)$ respectively represent the two-stage construction functions centered on device entities. The overall demonstration of this pipeline is shown in Figure 2, and the core logic is described as follows:

- The framework first feeds each text chunk t_i into the LLM and uses $\phi_{\text{extract}}(t_i)$ to identify and extract all potential device entities v_d. It then calls $A_{\text{fuse}}(\cdot)$ to align and merge all extracted v_d, combining different descriptions of the same real-world device into a normalized device entity set $\{v_d^{(1)}, v_d^{(2)}, \ldots, v_d^{(n)}\}$.

- Next, the framework treats the normalized device node set as anchors, reprocesses all text chunks with $\phi_{\text{extract}}(t_i)$ to extract all associated attribute entities v_p for each normalized device entity and assign predefined semantic community labels c to each attribute. Based on each community, $A_{\text{fuse}}(\cdot)$ is invoked again to align and aggregate attributes v_p associated with the same v_d and construct edges connecting v_d and v_p. The final output is a local entity set with community attributes, where each entity v is represented as a tuple $v = (p_{\text{key}}, p_{\text{value}}, c)$.

The resulting "star-shaped" knowledge graph G from ElecGraph-RAG faithfully mirrors the structure of knowledge in the electronics domain, with all attribute entities systematically organized under distinct "device" subgraphs. The detailed prompt template settings are provided in Appendix A.1.

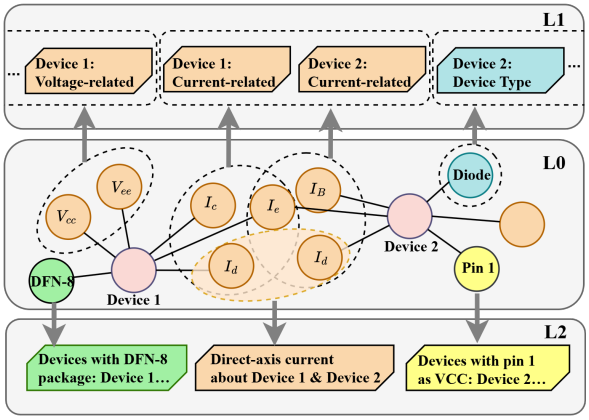


Figure 3: The Hierarchical Semantic Index Architecture of ElecGraph-RAG. The framework constructs device-specific profiles (L1: Local View) and cross-device thematic clusters (L2: Global View) in parallel from the atomic graph (L0), balancing intra-device detail with global navigability.

3.2 Hierarchical Semantic Indexing

To enable coarse-to-fine navigation, we construct a three-level index via a **dual-path abstraction strategy** (Figure 3). We derive both local (device-specific) and global (cross-device) views from the star-shaped graph to balance informational breadth and depth.

Graph factual layer L_0. This layer encodes atomic factual knowledge. We generate embeddings for nodes $E_{\text{node}}(v)$ and edges $E_{\text{edge}}(e)$. Additionally, to support semantic filtering, we construct dynamic community vectors $E_{\text{community}}(c)$ by fusing the community name with member embeddings, details are provided in Appendix B.1.

$$E_{\text{community}}(c) = \text{Fuse}\big(E_{\text{name}}(c), \{E_{\text{node}}(v_i) \mid v_i \in V_c\}\big) \quad (2)$$

Device profile layer L_1. To summarize the multifaceted properties of a single device, we perform intra-device clustering. For each device anchor v_d, the framework aggregates all associated attribute entities v_p and applies the density-based clustering algorithm HDBSCAN (Campello et al., 2013) to identify intrinsic functional themes (e.g., "Current Parameters"). Subsequently, attributes within each cluster C_{L1} are synthesized by an LLM into coherent natural language summaries S_{L1}. These summaries are then vectorized to form semantic embeddings $\{E_{L1}\}$, representing the major attribute dimensions of the device while reducing retrieval granularity.

Cross-device thematic summary layer L_2. To

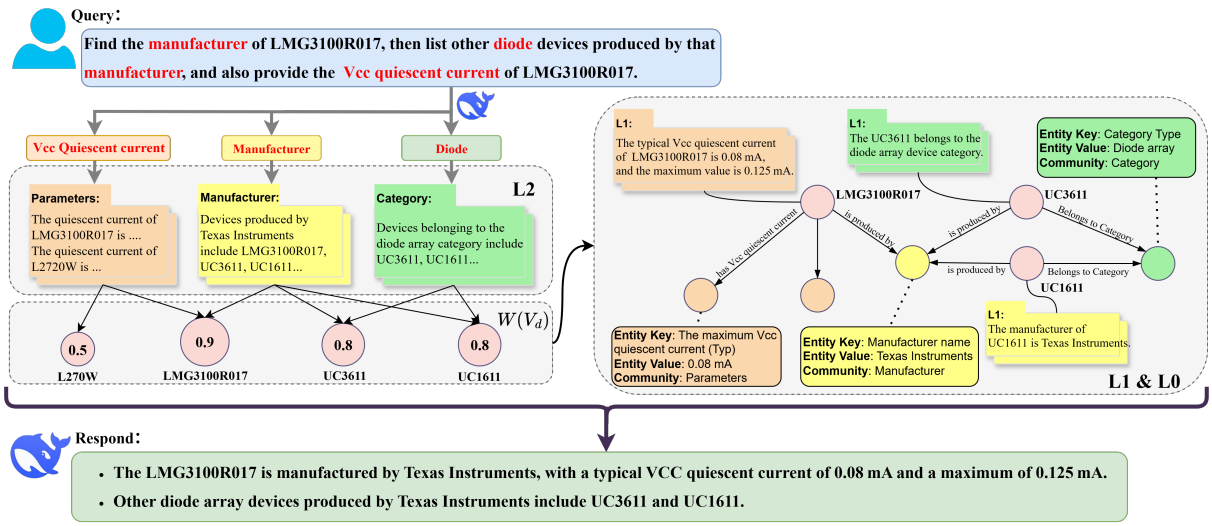


Figure 4: Illustration of the query retrieval process in the ElecGraph-RAG framework. The process first uses an LLM to decompose the query intent and extract keywords, which are matched to the corresponding communities. Retrieval is then performed within the matched communities at the L2 layer to obtain candidate device weights. The framework subsequently focuses on these candidate devices, retrieving L1 device profiles and L0 graph paths in parallel as evidence. Finally, these multi-granularity contexts are synthesized, and the LLM generates the final precise answer.

capture macro-level patterns while supporting continuous data ingestion, we employ a **hybrid incremental clustering strategy** directly on attribute entities across the graph.

We first define a composite distance metric $D(u, v)$ to calibrate semantic embeddings with lexical similarity(Wang et al., 2017), preventing the merging of functionally distinct but semantically close parameters:

$$D(u,v) = \alpha \cdot (1 - \cos(E_u, E_v)) + (1-\alpha) \cdot (1 - \text{Jaro}(S_u, S_v)) \quad (3)$$

where $\text{Jaro}(\cdot)$ denotes Jaro-Winkler string similarity.

For initialization, we apply HDBSCAN using this metric to discover high-density semantic regions. For incoming datasheets, we compute the distance between new attributes and existing L2 centroids. Matches trigger a localized summary update, fusing new evidence into the existing theme; non-matches beyond a threshold τ instantiate new thematic clusters. These clusters are organized under high-level semantic communities to facilitate pruned retrieval. Each L2 summary maintains an inverted index mapping the thematic cluster back to its contributing source devices $\{v_d\}$. This mapping enables the downstream retrieval module to identify and rank candidate devices based on their cumulative association with the matched query themes.

3.3 Hierarchical Retrieval

Retrieval in ElecGraph-RAG is formulated as a query-conditioned navigation problem over hierarchical semantic abstractions, inspired by the human cognitive process of selective attention. Rather than exhaustively traversing the entire knowledge graph, the framework progressively narrows the search space by leveraging higher-level semantic summaries to guide lower-level evidence extraction. Specifically, the L2 layer serves as a **cognitive navigation map** to identify relevant semantic regions, which enables early-stage pruning of irrelevant knowledge and focuses retrieval on a small set of candidate devices. Conditioned on this coarse localization, the system then performs parallel multi-granularity retrieval over device profiles (L1) and atomic factual paths (L0), ensuring both efficiency and precision.

As illustrated in Figure 4, the overall retrieval pipeline consists of four core stages. The hyperparameter settings used in this chapter are detailed in Appendix B.2.

Intent Understanding. Given a natural language query Q_{raw}, ElecGraph-RAG uses the LLM to extract a set of key concepts K_{goal}. Each key concept k is embedded into a vector E_k, resulting in the overall query intent vector E_q and a set of query anchor vectors $\{E_k\}$. Detailed prompts are provided in Appendix A.2.

Macro-Level Pruning (L2). Retrieval begins at the top layer. L2 summaries are organized under predefined semantic communities (e.g., *Voltage Parameters*, *Packaging*). To jointly consider the overall query intent and the focus of individual key concepts during retrieval, we construct a fused query vector for each key concept:

$$E_{\text{query}}(k) = \omega_q \cdot E_q + \omega_k \cdot E_k \tag{4}$$

where ω_q and ω_k are hyperparameters controlling the balance between global and local query semantics.

We compute the similarity between $E_{\text{query}}(k)$ and the community vectors $E_{\text{community}}(c)$ to identify the most relevant target communities c_k. This step performs semantic pruning by restricting the search scope to matched communities only. Within these active regions, we retrieve the top-K L2 thematic summaries $\{S_{L_2}^{(c_k)}\}$ using vector similarity.

Device Anchoring & Profiling. ElecGraph-RAG leverages the **inverted index** maintained by the retrieved L2 summaries to trace back to the set of linked **source device anchors**. We compute an importance weight $W(v_d)$ for each candidate device v_d to prioritize the most relevant hardware:

$$W(v_d) = \omega_{\text{freq}} \cdot F_{\text{norm}}(v_d) + \omega_{\text{qual}} \cdot S_{\text{avg}}(v_d) \tag{5}$$

where F_{norm} is the normalized frequency of the device appearing across retrieved L2 themes, and S_{avg} is the average relevance score. Based on $W(v_d)$, we select a candidate set V_{cand}. For these locked anchors, the system then retrieves their L1 Device Profiles in parallel to provide comprehensive device-level context.

Path Reasoning. For each candidate device entity $v_d \in V_{\text{cand}}$, ElecGraph-RAG evaluates all one-hop reasoning paths $P = (v_d, e, v_p)$. The relevance score S_{path} of each path is computed using a composite scoring function $Score$:

$$S_{\text{path}}(P, E_{\text{query}}(k)) = \max_{k \in K_{\text{goal}}} \text{Score}(\omega_d W(v_d), \omega_n S_n, \omega_e S_e, \omega_c S_c) \tag{6}$$

where $Score(P, E_{\text{query}}(k))$ integrates multiple semantic signals: importance weight $W(v_d)$ of the device v_d, the semantic similarity S_n between the attribute entity v_p and the query vector $E_{\text{query}}(k)$, the semantic similarity S_e between the edge e and the query vector $E_{\text{query}}(k)$, and the semantic similarity S_c between the community embedding $E_{\text{community}}(c_p)$ of entity v_p and the key concept vector E_k. Through this evaluation mechanism, ElecGraph-RAG produces, for each core candidate device v_d, a ranked subset of reasoning paths $P_{\text{graph}}(v_d)$ based on S_{path}, which serve as precise evidence sources for final answer generation.

3.4 Answer Generation

Finally, ElecGraph-RAG synthesizes the multi-granular evidence, global L2 context, L1 device profiles, and precise L0 paths. By explicitly providing the LLM with the source device identity for every piece of attribute evidence, the framework minimizes hallucination and ensures that comparative answers rigorously respect entity boundaries.

4 Experiments and Results Analysis

This section conducts a comprehensive evaluation of the effectiveness and efficiency of the ElecGraph-RAG framework in handling complex question-answering tasks in the electronics domain. We compare the overall performance of ElecGraph-RAG against existing baselines, analyze its adaptability and token consumption under different query types, and perform ablation studies to validate the contributions of its core modules.

4.1 Experimental Setup

Datasets. To simulate real-world scenarios of electronic design and device selection and to thoroughly evaluate the capabilities of RAG systems, we constructed a dedicated knowledge graph and question-answering dataset. Details of the dataset composition are provided in Appendix C.

ElecGraph-KG. We collected and processed 70 official datasheets of electronic devices from industry sources. To ensure diversity of data sources, 50 datasheets describe individual devices, while 20 datasheets cover multiple series of related models, spanning 20 categories of devices from 18 leading semiconductor manufacturers.

ElecGraph-QA. To evaluate complex QA capabilities, we constructed the **ElecGraph-QA** benchmark, a high-quality set of 200 questions. It covers four distinct categories: precise fact queries (70), comparative analysis across multiple devices (60), multi-hop reasoning (40), and macro-level summaries (30). The detailed generation process is described in Appendix A.3.

Baselines. To comprehensively evaluate the performance of ElecGraph-RAG, we selected several representative baseline methods for comparison:

- **LLM-only:** This baseline answers questions directly using a large language model without external retrieval, covering both Zero-Shot and Chain-of-Thought reasoning (Wei et al., 2022).

- **Retrieval-only:** This category represents standard text-based RAG methods, which retrieve from a vector index built on raw text chunks. Two implementations are included: Naive-RAG, which constructs a vector index on chunked datasheets and retrieves directly based on query embedding similarity; and HyDE (Gao et al., 2023), which uses an LLM to generate a hypothetical document from the original query and retrieves relevant chunks using this synthetic query, combining the retrieved chunks to produce an answer.

- **Graph-based RAG:** This category leverages graph data structures during retrieval. We include two advanced open-source frameworks: GraphRAG (Edge et al., 2024) and LightRAG (Guo et al., 2024). GraphRAG is evaluated with both global community summary retrieval (GraphRAG-Global) and local neighbor expansion retrieval (GraphRAG-Local). LightRAG is evaluated with three retrieval strategies: LightRAG-Local (low-level entity-focused retrieval), LightRAG-Global (high-level relation/theme-focused retrieval), and LightRAG-Hybrid (a hybrid strategy). Detailed graph indexing configurations are provided in Appendix B.3.

- **ElecGraph-RAG Ablation Models:** To validate the contributions of each innovative module of ElecGraph-RAG, we designed four ablation variants, detailed in Section 4.4.

Implementation Details. In all our experiments, ElecGraph-RAG and all baseline methods used the deepseek-chat model (Liu et al., 2024) for all LLM operations by default. We adopted Qwen-text-embedding-v4 (Zhang et al., 2025a) as the embedding model to vectorize all text. All relevant code and detailed hyperparameter settings are provided in Appendix B.4.

Evaluation Metrics and Methodology. We adopted both quantitative and qualitative evaluation. Accuracy was computed for precise fact and comparative analysis queries, where an answer was considered correct if it fully contained the core information of the reference answer (Wang et al., 2025). For complex reasoning and macro-level queries, we used head-to-head LLM-based evaluation (Guo et al., 2024), with deepseek-chat serving as a neutral judge for pairwise (Li et al., 2025), anonymous ranking. Although the same model family is used for both generation and evaluation, the judge is isolated from the retrieval process and only compares anonymized final answers, following established prior work. Evaluation focused on three aspects: (i) relevance and completeness, (ii) logical structure and clarity, and (iii) insightfulness and technical depth. Details of the evaluation setup and LLM-as-a-Judge prompts are provided in Appendix A.4.

Model	Precise Fact	Comparative Analysis
LLM-only		
Zero-shot	21.43%	8.33%
Chain-of-Thought	14.29%	15.00%
Retrieval-only		
Naive-RAG	82.86%	68.33%
HyDE	84.29%	70.00%
Graph-based RAG		
GraphRAG-Global	62.86%	41.67%
GraphRAG-Local	82.86%	78.33%
LightRAG-Global	41.13%	25.00%
LightRAG-Local	85.71%	66.67%
LightRAG-Hybrid	88.57%	76.67%
Our Method		
ElecGraph-RAG	**94.28%**	**85.00%**

Table 1: Accuracy (%) comparison on precise fact and comparative analysis.

4.2 Quantitative Evaluation

We first evaluated the accuracy of ElecGraph-RAG and all baseline methods on "Precise Fact" and "Comparative Analysis" queries (Table 1). Results show that LLM-only methods, lacking external retrieval, perform poorly, confirming the necessity of retrieval augmentation. ElecGraph-RAG achieved the best accuracy of 94.28% and 85.00% on the two tasks, respectively, surpassing both Retrieval-only and advanced Graph-based RAG baselines. Based on this, the following qualitative evaluation on "Multi-hop Reasoning" and "Macro-summary" focuses on comparisons among RAG frameworks.

4.3 Qualitative Evaluation

Table 2 shows that ElecGraph-RAG significantly outperforms text-based RAG baselines such as Naive-RAG and HyDE, which lack structured knowledge, achieving an overall win rate consistently exceeding 80% on multi-hop reasoning and

	Multi-hop Reasoning		Macro-summary	
	Naive-RAG	**ElecGraph-RAG**	**Naive-RAG**	**ElecGraph-RAG**
Relevance & Completeness	25.00%	75.00%	33.33%	66.67%
Logical Structure & Clarity	18.75%	81.25%	15.00%	85.00%
Insightfulness & Depth	17.50%	82.50%	28.33%	71.67%
Overall	**17.50%**	**82.50%**	**15.00%**	**85.00%**
	Hyde	**ElecGraph-RAG**	**Hyde**	**ElecGraph-RAG**
Relevance & Completeness	17.50%	82.50%	16.67%	83.33%
Logical Structure & Clarity	10.00%	90.00%	15.00%	85.00%
Insightfulness & Depth	8.75%	91.25%	8.33%	91.67%
Overall	**11.25%**	**88.75%**	**6.67%**	**93.33%**
	GraphRAG-Local	**ElecGraph-RAG**	**GraphRAG-Local**	**ElecGraph-RAG**
Relevance & Completeness	43.75%	56.25%	60.00%	40.00%
Logical Structure & Clarity	46.25%	53.75%	38.33%	61.67%
Insightfulness & Depth	38.75%	61.25%	70.00%	30.00%
Overall	**41.25%**	**58.75%**	**48.33%**	**51.67%**
	GraphRAG-Global	**ElecGraph-RAG**	**GraphRAG-Global**	**ElecGraph-RAG**
Relevance & Completeness	43.75%	56.25%	61.67%	38.33%
Logical Structure & Clarity	55.00%	45.00%	36.67%	63.33%
Insightfulness & Depth	43.75%	56.25%	65.00%	35.00%
Overall	**48.75%**	**51.25%**	**46.67%**	**53.33%**
	LightRAG-Local	**ElecGraph-RAG**	**LightRAG-Local**	**ElecGraph-RAG**
Relevance & Completeness	50.00%	50.00%	55.00%	45.00%
Logical Structure & Clarity	41.25%	58.75%	35.00%	65.00%
Insightfulness & Depth	48.75%	51.25%	53.33%	46.67%
Overall	**45.00%**	**55.00%**	**41.67%**	**58.33%**
	LightRAG-Global	**ElecGraph-RAG**	**LightRAG-Global**	**ElecGraph-RAG**
Relevance & Completeness	45.50%	57.50%	43.33%	56.67%
Logical Structure & Clarity	35.00%	65.00%	40.00%	60.00%
Insightfulness & Depth	43.75%	56.25%	40.00%	60.00%
Overall	**37.50%**	**62.50%**	**26.67%**	**73.33%**
	LightRAG-Hybrid	**ElecGraph-RAG**	**LightRAG-Hybrid**	**ElecGraph-RAG**
Relevance & Completeness	53.75%	46.25%	56.67%	43.33%
Logical Structure & Clarity	42.50%	57.50%	56.67%	43.33%
Insightfulness & Depth	55.00%	45.00%	48.33%	51.67%
Overall	**46.25%**	**53.75%**	**55.00%**	**45.00%**

Table 2: Win rate (%) of ElecGraph-RAG over baselines on multi-hop reasoning and macro-summary queries

macro-summary queries.

For multi-hop reasoning queries, ElecGraph-RAG achieves the highest overall win rate across all comparisons, with the most pronounced advantage observed in the "Logical Structure and Clarity" dimension. Through the synergy of the L1 device profile layer and the L0 factual graph layer, ElecGraph-RAG provides the LLM with logically coherent and highly relevant evidence chains, thereby generating clearer and more complete answers for tasks requiring deep reasoning.

In macro-summary query tasks, ElecGraph-RAG demonstrates strong competitiveness. Experimental results show that our framework significantly outperforms baselines such as GraphRAG-Global and LightRAG-Global, which adopt pure global retrieval strategies. This indicates that our dual-path abstraction strategy, combining device-centric L1 profiles and L2 thematic clusters, is more effective at aggregating and distilling macro-level information than generic global community summaries. Even when compared with top-performing hybrid baselines such as LightRAG-Hybrid, ElecGraph-RAG remains highly competitive, validating the effectiveness of our framework in handling global-level queries.

4.4 Ablation Analysis

To verify the effectiveness of each core module in the ElecGraph-RAG framework, we removed four specific modules and evaluated their impact on the performance of four query types, with the results shown in Figure 5, and the detailed experimental design is provided in Appendix B.5.

The experimental results clearly reveal the differentiated roles of the modules. Removing the

Model	Prompt	Completion	Total
Retrieval-only			
Naive-RAG	1,737,837	31,789	1,769,626
HyDE	2,389,431	113,363	2,502,794
Graph-based RAG			
GraphRAG-Local	2,099,273	133,536	2,232,809
GraphRAG-Global	44,733,550	603,552	45,337,102
LightRAG-Local	2,425,371	95,842	2,521,213
LightRAG-Global	1,584,920	82,568	1,667,488
LightRAG-Hybrid	3,426,133	100,569	3,526,702
Our Method			
ElecGraph-RAG	**783,775**	**53,661**	**837,436**

Table 3: Comparison of total token consumption on ElecGraph-QA.

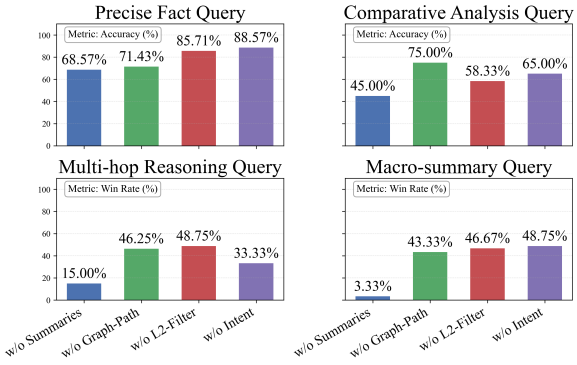


Figure 5: Ablation analysis of ElecGraph-RAG on four query types. For "Precise Fact" and "Comparative Analysis" queries, the metric is accuracy, while for "Multi-hop Reasoning" and "Macro-summary" queries, the metric is win rate.

summary layer (w/o Summaries) causes a severe performance drop on macro-level queries, with a win rate of only 3.33%, and also severely affects comparative analysis capability. This demonstrates the critical role of dual-path abstraction in global information aggregation and providing rich context. Removing graph path reasoning (w/o Graph-Path) has the largest impact on multi-hop reasoning queries, proving the importance of structured path evidence from the L0 layer in constructing rigorous logical chains. Removing the L2 global semantic pruning strategy (w/o L2-Filter) disrupts the cognitive navigation mechanism, failing to lock onto the correct focus of attention, and removing the intent understanding module (w/o Intent) both lead to performance degradation across all query types.

These ablation results strongly demonstrate that every module in ElecGraph-RAG's hierarchical indexing and focused retrieval architecture makes a key contribution to achieving its superior overall performance.

4.5 Cost Analysis

We evaluate the resource efficiency of ElecGraph-RAG by measuring total token consumption on the ElecGraph-QA benchmark (Table 3).

Our framework achieves a 52.7% reduction in total tokens compared to Naive-RAG. Crucially, when compared to the strongest quality baseline, LightRAG-Hybrid, ElecGraph-RAG consumes only 23.7% of its tokens. The efficiency gain is most pronounced against GraphRAG-Global, which incurs massive costs by traversing numerous communities.

ElecGraph-RAG leverages the macro-level L2 layer to efficiently focus on a small set of highly relevant candidate devices at the early stage of retrieval, avoiding the processing of large amounts of irrelevant context and significantly reducing both LLM API costs and generation latency, demonstrating the framework's advanced capability to balance answer quality and runtime cost.

5 Conclusion

This paper presents ElecGraph-RAG, the first retrieval-augmented generation framework explicitly designed to align hierarchical retrieval with the device-centric, star-shaped ontology of electronic datasheets. By introducing ontology-aligned semantic abstraction and coarse-to-fine retrieval, ElecGraph-RAG addresses a fundamental failure mode of existing graph-based RAG systems, entity-attribute misattribution in technical question answering. In addition to improved accuracy on both factual and comparative queries, our approach substantially reduces retrieval and generation cost, demonstrating that faithful retrieval and resource efficiency are not conflicting objectives but can be jointly achieved through principled structural design.

Comprehensive experiments validate the effectiveness of ElecGraph-RAG. It achieves 94.28% and 85.00% accuracy on "Precise Fact" and "Comparative Analysis" queries, and over 80% win rate on complex tasks compared with unstructured text RAG baselines. Moreover, its token usage is only 23.7% of the strongest baseline, demonstrating a balanced trade-off among accuracy, overall performance, and resource efficiency, and offering a new paradigm for building production-grade domain-specific RAG systems.

Limitations

While ElecGraph-RAG demonstrates superior performance in the electronics domain, it has several limitations: (1) **PDF Parsing:** The framework's performance is still partially bounded by the quality of initial PDF-to-text conversion, particularly for complex nested tables. (2) **Real-time Updates:** While we support incremental clustering in L2, a fully real-time graph update mechanism for rapidly changing data streams is yet to be implemented.

Ethics Statement

The data used in this study (ElecGraph-KG) consists of publicly available official datasheets from semiconductor manufacturers. No private or sensitive user data was involved. The proposed framework is intended for design assistance and does not generate harmful or biased content. However, users should verify critical parameters in industrial safety-sensitive applications.

References

Ricardo JGB Campello, Davoud Moulavi, and Jörg Sander. 2013. Density-based clustering based on hierarchical density estimates. In *Pacific-Asia conference on knowledge discovery and data mining*, pages 160–172. Springer.

Alla Chepurova, Yuri Kuratov, Bulatov, and Burtsev. 2024. Prompt me one more time: A two-step knowledge extraction pipeline with ontology-based verification. *Workshop on Graph-based Methods for Natural Language Processing*.

Manuel Cossio. 2025. A comprehensive taxonomy of hallucinations in large language models. *http://arxiv.org/abs/2508.01781*.

Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, Dasha Metropolitansky, Robert Osazuwa Ness, and Jonathan Larson. 2024. From local to global: A graph rag approach to query-focused summarization. *arXiv preprint arXiv:2404.16130*.

Shahul Es, Jithin James, Luis Espinosa Anke, and Steven Schockaert. 2024. Ragas: Automated evaluation of retrieval augmented generation. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 150–158.

Luyu Gao, Xueguang Ma, Jimmy Lin, and Jamie Callan. 2023. 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)*, pages 1762–1777.

Zirui Guo, Lianghao Xia, Yanhua Yu, Tu Ao, and Chao Huang. 2024. Lightrag: Simple and fast retrieval-augmented generation. *arXiv preprint arXiv:2410.05779*.

Yihan Jiao, Zhehao Tan, Dan Yang, Duolin Sun, Jie Feng, Yue Shen, Jian Wang, Peng Wei, and AntGroup Hangzhou Zhejiang China. 2025. Hirag: Hierarchical-thought instruction-tuning retrieval-augmented generation. *arXiv preprint arXiv:2507.05714*.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick SH Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *EMNLP (1)*, pages 6769–6781.

Dawei Li, Bohan Jiang, Liangjie Huang, Alimohammad Beigi, Chengshuai Zhao, Zhen Tan, Amrita Bhattacharjee, Yuxuan Jiang, Canyu Chen, Tianhao Wu, and 1 others. 2025. From generation to judgment: Opportunities and challenges of llm-as-a-judge. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 2757–2791.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, and 1 others. 2024. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*.

Mingjie Liu, Teodor-Dumitru Ene, Robert Kirby, Chris Cheng, Nathaniel Pinckney, Rongjian Liang, Jonah Alben, Himyanshu Anand, Sanmitra Banerjee, Ismet Bayraktaroglu, and 1 others. 2023. Chipnemo: Domain-adapted llms for chip design. *arXiv preprint arXiv:2311.00176*.

Yi Luan, Jacob Eisenstein, Kristina Toutanova, and Michael Collins. 2021. Sparse, dense, and attentional representations for text retrieval. *Transactions of the Association for Computational Linguistics*, 9:329–345.

Han Peng, Jinhao Jiang, Zican Dong, Wayne Zhao, and Lei Fang. 2025. Cafe: Retrieval head-based coarse-to-fine information seeking to enhance multi-document qa capability. *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*.

Shu Wang, Yixiang Fang, Yingli Zhou, Xilin Liu, and Yuchi Ma. 2025. Archrag: Attributed community-based hierarchical retrieval-augmented generation. *arXiv preprint arXiv:2502.09891*.

Yaoshu Wang, Jianbin Qin, and Wei Wang. 2017. Efficient approximate entity matching using jaro-winkler distance. In *International conference on web information systems engineering*, pages 231–239. Springer.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou,

and 1 others. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, and 1 others. 2025a. Qwen3 embedding: Advancing text embedding and reranking through foundation models. *arXiv preprint arXiv:2506.05176*.

Yaoze Zhang, Rong Wu, Pinlong Cai, Xiaoman Wang, Guohang Yan, Song Mao, Ding Wang, and Botian Shi. 2025b. Leanrag: Knowledge-graph-based generation with semantic aggregation and hierarchical retrieval. *arXiv preprint arXiv:2508.10391*.

Appendix

A Prompt Details

A.1 Knowledge Graph Construction Prompts

Our knowledge graph construction pipeline aims to systematically build a device-centric, well-structured star-shaped knowledge graph from unstructured PDF datasheets, addressing core challenges such as entity ambiguity, information dispersion, and attribute misassignment. The detailed process is as follows:

First, the raw PDF datasheets are parsed using OCR and segmented into text chunks. Subsequently, the framework applies the prompt for device model extraction (see Figure 6) to each text chunk to identify and extract all potential device model names. This process records the chunk ID where each model occurs and, in parallel, detects "common information chunks" that are not tied to specific models but instead apply to the entire document or product series.

Due to potential issues in the initial model extraction, such as the coexistence of aliases, series names, and specific models (e.g., LM392DGKT and LM392), or redundancies caused by formatting errors, we designed a crucial entity fusion and normalization step. We employ heuristic rules (e.g., subsequence relationships in device names) to automatically cluster potentially related models into groups. For each group, the framework invokes an LLM to construct context based on all models within the group and their associated text chunk IDs. Using built-in domain-specific rules for electronic components (e.g., distinguishing series names, packaging suffixes, orderable models, and datasheet subject models), the LLM intelligently determines whether to perform *merge* or *delete* operations. For instance, it merges information from the package model LM392DGKT into the base model LM392. This process ultimately outputs a clean, unique, and normalized device entity list, where each entity aggregates all its relevant text chunk IDs.

After obtaining the normalized device list, the framework performs attribute extraction for each device. First, it identifies all relevant text chunks for the device, including its dedicated chunks and all common information chunks. Each individual text chunk is then traversed, and using the *Attributes_prompt* (see Figure 8), with the device name as an anchor, the framework extracts structured parameters for the device. Finally, parameters extracted from all relevant text chunks are aggregated to form a complete attribute entity set for the device.

After aggregating all attribute entities for each device, the framework performs a fine-grained post-processing stage for conflict resolution. This process is handled by a tool-using LLM agent designed to merge, select, or consolidate conflicting values. To resolve ambiguity, the agent can autonomously request and analyze the original text context, ensuring the final attribute set is both consistent and accurate.

The final step in the pipeline is to assemble the normalized device entities and their cleaned attribute entities into a coherent knowledge graph. This process is governed by a set of rules designed to strictly enforce the device-centric, star-shaped topology:

- **Dynamic Edge Naming:** The relationship type for most attributes is dynamically generated from the attribute's base name. A key aspect of this process is normalization: for parameter attributes that include specific conditions (e.g., "Forward Voltage @Ta=25°C"), the framework strips these conditional suffixes to generate a generalized edge type (e.g., "has Forward Voltage"). This design choice allows multiple parameter nodes representing the same attribute under different conditions to share a single, canonical relationship type, which significantly improves the efficiency and robustness of retrieval for general queries.

- **Specific Relations:** Key relationships are explicitly defined, such as the "is produced by" edge connecting a "Device" to a "Manufacturer", and the "Belongs to Category" edge connecting a "Device" to its lowest-level "Category".

This rule-based assembly ensures that the final knowledge graph is consistent, logically structured, and optimized for the subsequent hierarchical retrieval tasks.

A.2 Intent Understanding Prompts

During retrieval, the framework employs the LLM to decompose the user query intent via chain-of-thought reasoning and extract a list of potential keywords contained in the query (see Figure **??**).

Device Extraction

---System Prompt---

You are an expert in analyzing electronic device datasheets, your task is to analyze the following Markdown text block extracted from a datasheet. This block may contain the names of multiple devices or consist solely of common information applicable to all models. Text for analysis: {Text chunk}

---User Prompt---

Please extract all specific model numbers appearing in the above electronic device datasheet paragraph and determine whether the paragraph contains common information.

1. "Devices_found" represents all specific electronic component models appearing within this text segment. All actual component devices must be output. Constructing non-existent component devices is not permitted. If no devices appear, an empty array [] is returned.
   - Extract only the "main electronic componentdevices" appearing in the segment, i.e., the component devices primarily described. Do not extract peripheral or supporting passive components.
   - Avoid extracting the following types of identifiers unless they are the core description of the datasheet segment:
   - Replacement devices: For example, "Replaceable with XXX device" is explicitly mentioned in the description.
   - Auxiliary components used in the application circuit: For example, other transistors, resistors, inductors, etc. used in the recommended circuit.
   - PCB layout recommendation numbers or names.
   - Component names required for PCB design.
   - Environmental, test standard, or certification numbers.

2. If the section appears to be general information (i.e., common to all devices or not specific to a particular model), please remove it. "Possible_common" is set to true. The following criteria can beused to determine whether a specification is common:
   - If the text mentions multiple sub-devices but uses a unified description or emphasizes that they share common features, it is also considered common information.
   - If the paragraph only describes common features, common parameters, or universal standards without mentioning any specific device;
   - If the title or content clearly indicates that the specification is common to the entire series or the applicable range;

3. If the above common information conditions are not met, set "Possible_common" to false.

---Output Example---

```
{
  "Devices_found": ["Device 1", "Device 2",  ... ]
  "Possible_common": true/false
}
```

Figure 6: Prompt for device model extraction and common information identification.

A.3 ElecGraph-QA Generation Process

The ElecGraph-QA benchmark was systematically constructed to ensure question quality, diversity, and relevance to real-world applications in the electronics domain. The process is detailed below.

Contextual Grounding. To ground the question generation process in realistic data, we employed a context-sampling strategy. In each generation round, the system randomly selected five datasheets from our corpus and extracted their full text content. This collection of texts served as the context provided to the Large Language Model (LLM), ensuring that the generated questions were based on a diverse but manageable set of information.

Persona-Based Question Generation. To guarantee the practical relevance and domain-specificity of the questions, we instructed the LLM to adopt one of four pre-defined expert personas for each generation task. This persona-based approach ensured that the generated questions covered a wide spectrum of user intentions encountered in the electronics industry (see Figure 9):

- **Hardware Engineer:** Responsible for power supply and signal conditioning module design, focusing on device voltage and current parameters, heat dissipation capabilities, and reliability, including power device and sensor selection.
- **Procurement Specialist:** Develops consumer electronics/IoT products, focusing on peripheral device interface compatibility, analog signal front-end configuration, and system integration efficiency.
- **Embedded Developer:** Responsible for product selection and price comparison, focusing on device packaging compatibility, substitution relationships between mainstream manufacturers, and model performance comparison.
- **Test and Validation Engineer:** Performs functional, extreme parameter, and logical behavior testing on samples in the laboratory, focusing on device response speed, extreme values, and behavioral consistency.

Quality Control and Curation. After multiple generation rounds, we curated a final set of 200 questions. To eliminate any potential for LLM-induced hallucinations or factual inaccuracies, every generated answer was manually verified against the source datasheets by human experts. Any answers found to be incorrect or ambiguous were either corrected or discarded, ensuring the high quality and reliability of the final benchmark. The

Attribute Extraction

---System Prompt---
You are an expert in electronic components, specializing in extracting structured information from datasheets.The following is the text content:{Text chunk}

---User Prompt---
Extract the structured parameters related to the device model `{Device_name}` from the above text and output the result in JSON format.
- Only extract content that is clearly related to or reasonably applicable to `{Device_name}`. Do not extract parameter information unrelated to `{Device_name}`.
- If the text contains difficult-to-understand symbols, convert them to a more understandable form.
- If a parameter is valid only under specific conditions (e.g., "Ta = 25°C"), indicate the condition in the parameter name, such as "Forward Voltage @Ta = 25°C".
- If the parameter is a minimum value (Min), typical value (Typ), or maximum value (Max), indicate it in the parameter name, such as "Capacitance (Typ)".
- Under the "Parameters" field, all device-related parameter information can be extracted, but only those parameters explicitly mentioned in the text and their corresponding values are required.

---Output Example---

```
{
 "Manufacturer": {
      "Name": "Manufacturer"
 },
 "Package type": {
      "Package": "Package type 1",
      "Package": "Package type 2",
      ...
 },
"Basic information": {
      "Description": "Description of the device",
      "Function": "Function description 1, Function description 2, ...",
      "Applications": "Applicable Application 1, Applicable Application 2, ...",
      ....
 },
 "Pin Configuration": {
      "Pin 1": "Pin Name 1 (Pin Function 1)",
      "Pin 2": "Pin Name 2 (Pin Function 2)",
       ...
 },
 "Parameters": {
      "Parameter Name 1": "Parameter Value 1",
      "Parameter Name 2": "Parameter Value 2",
       ...
 },
  ...
}
```

Figure 7: Prompt for structured extraction of device-centric attribute entities.

final distribution of question types is detailed in the main paper's Experimental Setup section.

A.4 Evaluation Setup and LLM-as-a-Judge Prompts

To ensure a comprehensive and fair assessment of our framework's performance, we employed two distinct evaluation methodologies, each tailored to the nature of the query types. An LLM was utilized as an impartial judge in both setups.

A.4.1 Accuracy Evaluation for Factual Queries

For "Precise Fact" and "Comparative Analysis" queries, which have definitive correct answers, we conducted a point-wise accuracy evaluation against a ground truth. The evaluation process for each question-answer pair involved providing the LLM judge with three pieces of information: the original **question**, the manually verified **ground truth answer**, and the **generated answer** from the model being evaluated.

Evaluation Criteria. The LLM was instructed to follow a strict "core fact matching" rule. A generated answer was deemed correct if it semantically contained all the key factual information present in the ground truth answer. The presence of additional, correct information in the generated answer was permissible and did not result in a penalty. An answer was marked as incorrect if it contradicted any core fact or omitted critical information from the ground truth. The detailed system prompt, which enforces a JSON output format specifying correctness and reasoning, is shown in Figure 10.

A.4.2 Head-to-Head (H2H) Evaluation for Complex Queries

For the more subjective "Multi-hop Reasoning" and "Macro-summary" queries, where a single ground truth answer is often insufficient, we adopted a pair-wise, head-to-head (H2H) evaluation methodology. In this setup, the LLM judge was presented with the original **question** and two anonymously provided answers ("Answer 1" and "Answer 2"), one from our ElecGraph-RAG frame-

Attribute Extraction

---System Prompt---
You are an expert in electronic components, specializing in extracting structured information from datasheets.The following is the text content: {Text chunk}

---User Prompt---
Extract the structured parameters related to the device model `{Device_name}` from the above text and output the result in JSON format.
- Only extract content that is clearly related to or reasonably applicable to `{Device_name}`. Do not extract parameter information unrelated to `{Device_name}`.
- If the text contains difficult-to-understand symbols, convert them to a more understandable form.
- If a parameter is valid only under specific conditions (e.g., "Ta = 25°C"), indicate the condition in the parameter name, such as "Forward Voltage @Ta = 25°C".
- If the parameter is a minimum value (Min), typical value (Typ), or maximum value (Max), indicate it in the parameter name, such as "Capacitance (Typ)".
- Under the "Parameters" field, all device-related parameter information can be extracted, but only those parameters explicitly mentioned in the text and their corresponding values are required.

---Output Example---

```
{
 "Manufacturer": {
      "Name": "Manufacturer"
 },
 "Package type": {
      "Package": "Package type 1",
      "Package": "Package type 2",
      ...
 },
 "Basic information": {
      "Description": "Description of the device",
      "Function": "Function description 1, Function description 2, ...",
      "Applications": "Applicable Application 1, Applicable Application 2, ...",
      ....
 },
 "Pin Configuration": {
      "Pin 1": "Pin Name 1 (Pin Function 1)",
      "Pin 2": "Pin Name 2 (Pin Function 2)",
       ...
 },
 "Parameters": {
      "Parameter Name 1": "Parameter Value 1",
      "Parameter Name 2": "Parameter Value 2",
       ...
 },
  ...
}
```

Figure 8: Prompt for structured extraction of device-centric attribute entities.

work and one from a baseline model.

Evaluation Criteria. The LLM, acting as a domain expert, was tasked with selecting the superior answer based on three specific dimensions: (i) **Relevance & Completeness**, (ii) **Logical Structure & Clarity**, and (iii) **Insightfulness & Technical Depth**. The full system prompt provided to the LLM judge is shown in Figure 11.

Bias Mitigation and Robust Win Condition. To mitigate potential positional bias from the LLM judge, each pair of answers was evaluated twice, with their order swapped in the second run (i.e., A vs. B, then B vs. A). A model was declared the winner for a given dimension only if the LLM's preference remained consistent across both evaluations. If the judgments were inconsistent between the two runs, the result for that dimension was considered a tie. This robust process ensures the reliability of the final win-rate calculations presented in the main paper.

B Implementation and Configuration Details

B.1 Configuration of Dynamic Community Vectors

As introduced in the Graph Factual Layer subsection, the dynamic community vector $E_{\text{community}}(c)$ is constructed via a $Fuse(\cdot)$ function, formally defined in Equation (2). The fusion is implemented as a weighted linear combination of two key components: the community's conceptual name vector and the average vector of its member entities.

Community Name Vector ($E_{\text{name}}(c)$). To create a rich semantic representation for each community's name, we do not simply embed the name itself (e.g., "Manufacturer"). Instead, each community is associated with a pre-defined set of semantically related keywords and descriptions, which are maintained within the framework's configuration. For instance, the "Parameters" community is associated with the string "Technical parameters, electrical characteristics, physical characteristics, performance metrics". The resulting embedding of this descriptive string serves as the community name vector, denoted as $E_{\text{name}}(c)$.

QA Generation

---System Prompt---
You are an expert in the field of electronic components. Your task is to play a designated role and generate high-quality question-answer (QA) pairs based on the provided original corpus (from the datasheet).

1.All QA questions must be derived from the provided datasheet corpus and avoid references to content outside the corpus.
2.Your output must strictly adhere to the following JSON format and should not include any additional explanations or non-JSON text.
3.The output should be a JSON list [] containing multiple QA objects. Each QA object must have the following four keys:
-"question": (string) A specific question posed based on the corpus and the persona's perspective.
-"answer": (string) An accurate and concise answer to the question based on the corpus content.
-"source_devices": (list of strings) A list of one or more core device models on which the answer is based. If the answer is a high-level summary, please list all relevant representative devices.
-"question_type": (string) The type of question, which must be one of the following four types.

4.Detailed definitions of the four question types:
-"Precise Fact Queries": A question about one or more specific attributes of a single device. The answer is typically a fact or value found directly in the corpus. Used to test the RAG system's ability to locate precise information.
-"Comparative Analysis ": Explicitly compares the similarities and differences between two or more devices in one or more attributes. Questions typically include terms such as "contrast," "compare," and "difference from..." This test tests the RAG system's ability to integrate and analyze information from multiple sources.
-"Multi-hop Reasoning": The answer cannot be directly derived from a single point of information and requires at least two or more steps of reasoning. This typically involves finding one piece of information (A) and using A as a clue to find another piece of information (B), ultimately integrating the chain of information. This test tests the RAG system's ability to perform logical reasoning and relationship traversal.
-"Macro-Summary Queries": Questions that summarize and generalize a specific category of devices or a specific topic. The answer requires integrating multiple devices or information points. This test tests the RAG system's ability to summarize global information and discover insights.

---Output Example---

```
[
  {
    "question": "What is the operating temperature range of the UC1611?",
    "answer": "The operating temperature range of this device is -55°C to 125°C.",
    "source_devices": ["UC1611"],
    "question_type": "Precise Fact Queries"
  },
  {
    "question": "Compare the package types of the UC1611 and LMG3010R017.",
    "answer": "The UC1611 is available in DIL(N) and CDIP(JG) packages, while the LMG3010R017 is available in VQFN-15 packages.",
    "source_devices": ["UC1611", "LMG3010R017"],
    "question_type": "Comparative Analysis"
  },
  {
    "question": "Design a 5V output DC-DC buck converter requiring a GaN power device to improve efficiency and a low on-resistance MOSFET as a synchronous rectifier switch. Please recommend suitable GaN and MOSFET models and explain their synergistic advantages."
    "answer": "GaN device: LMG3100R017 (high-efficiency main switch), MOSFET: STFI20NK50Z (low on-resistance synchronous rectifier), Synergistic advantage: GaN improves switching frequency and efficiency, while the MOSFET optimizes rectification loss, resulting in overall enhanced conversion efficiency."
    "source_devices": ["LMG3100R017", "STFI20NK50Z"],
    "question_type": "Multi-hop Reasoning "
  },
  {
    "question": "Find devices suitable for solar panel bypass applications and list their key features."
    "answer": "The MP6914 is an ideal diode specifically designed for solar panel bypass applications. Its key features include an integrated 30V, 5.3 mΩ power MOSFET, extremely low reverse leakage current and forward voltage drop, and an SOIC8-EP package."
    "source_devices": ["MP6914"],
    "question_type": "Macro-Summary Query"
  }
]
```

---User Prompt---
Original corpus: {datasheet_text}

Your role: {persona['role']}
Role description: {persona['description']}

Please act as the above role and, based on the provided raw datasheet corpus, generate 5 high-quality question-answer pairs that align with the role's focus areas.

Figure 9: Prompt for persona-based generation of QA pairs.

Aggregated Member Vector. The set of member entity vectors, $\{E_{\text{node}}(v_i) \mid v_i \in V_c\}$, is aggregated into a single representative vector by computing their element-wise mean. This results in the average member vector, which we can denote as $\bar{E}_{\text{members}}(c)$.

Fusion Implementation. The $Fuse(\cdot)$ function is thus implemented as the following weighted sum:

$$E_{\text{community}}(c) = w_{\text{name}} \cdot E_{\text{name}}(c) + w_{\text{members}} \cdot \bar{E}_{\text{members}}(c)$$

The weights for this fusion are set as follows:

- The weight for the community name vector, w_{name}, is set to **0.6**.
- The weight for the average member vector, w_{members}, is set to **0.4**.

Accuracy Evaluation

---System Prompt---
You are a rigorous and fair evaluation expert, and your task is to judge whether a "generated answer" accurately contains the core factual information in the "standard answer.

Evaluation Rules:
1. Core Fact Match: You must determine whether the generated answer contains key information from the standard answer. A verbatim match is not necessary; the key is semantic and factual correctness. The standard answer may sometimes be compared.
2. Allow Additional Information: The generated answer can contain more information than the standard answer. As long as this additional information is correct and does not contradict the standard answer, it should be considered correct.
3. Error: If the generated answer contradicts any core facts in the standard answer or completely omits key information from the standard answer, it should be considered incorrect.

---Output Example---
```
{
  "is_correct": true,
  "reasoning": "The generated answer completely and accurately covers all key information points in the standard answer and provides additional useful details."
}
or
{
  "is_correct": false,
  "reasoning": "The generated answer omits the resistor value information for the APD2220-203 model from the standard answer."
}
```

---User Prompt---
Question: {question}
Ground Truth Answer: {ground_truth_answer}
Generated Answer to Evaluate: {rag_answer}

Figure 10: Prompt for the point-wise accuracy evaluation.

H2H Evaluation

---System Prompt---
You are a senior electronic engineering expert and semiconductor data sheet analyst. Your task is to act as an absolutely fair and rigorous referee and conduct an indepth evaluation of two answers to the same technical question generated by AI assistants.

Core Evaluation Principles:
1. Fact-Based: All your judgments must be strictly based on the question and the content of the two answers themselves. Do not introduce external knowledge.
2. Relative Comparison: Your task is to judge which answer is "better" rather than assigning an absolute score to each answer.
3. Maintain Anonymity: You do not know which system "Answer 1" and "Answer 2" came from. Please judge based solely on the quality of their content.

Evaluation Dimensions:
1. Answer Relevance & Completeness: Does the answer closely address the question and fully address all user queries? Does it omit any key information?
2. Logical Structure & Clarity: Is the answer clearly organized and easy to understand? For complex questions, does the reasoning and information organization follow a logical progression?
3. Empowerment & Technical Depth: Does the answer go beyond a simple presentation of facts to effectively help professional readers make more informed technical or business decisions?

---Output Example---
```
{
  "relevance_completeness": {
      "winner": "Answer 1",
      "reasoning": "Answer 1 is more relevant to the question and fully answers all sub-questions."
  },
  "logical_clarity": {
      "winner": "Answer 2",
      "reasoning": "Answer 2 is more clearly structured and uses a bullet point format, making it easier to read."
  },
  "empowerment_depth": {
      "winner": "Answer 1",
      "reasoning": "Answer 1 not only provides the parameters but also explains their significance in the application, making it more insightful."
  }
}
```

---User Prompt---
Question: {question}
RAG Answer 1: {rag_answer1}
RAG Answer 2: {rag_answer2}

Figure 11: Prompt for the head-to-head (H2H) evaluation.

The choice of these weights is deliberate. By assigning a higher weight to the community name vector ($w_{\text{name}} = 0.6$), we ensure that the final community embedding $E_{\text{community}}(c)$ primarily reflects its abstract, predefined semantic concept (e.g., the general concept of "Parameters"). This makes the community vector a more stable and reliable target during the L2 Macro-summary Retrieval stage, where broad conceptual keywords extracted from the user's query need to be matched to the correct community. The non-zero weight for the average member vector ($w_{\text{members}} = 0.4$) serves as a data-driven refinement, grounding the abstract concept in the specific context of the entities actually

Hyperparameter	Symbol	Value
L2 Macro-summary Retrieval		
Query Fusion Weight	ω_q	0.4
Keyword Weight	ω_k	0.6
Community Match Threshold	–	0.3
L2 Summary Score Threshold	–	0.4
L1 Device Summary Retrieval		
Device Frequency Weight	ω_{freq}	0.4
Quality (Relevance) Weight	ω_{qual}	0.6
Total Summaries to Retrieve	K_{total}	10
Similarity Weight (Ranking)	–	0.7
Device Weight (Ranking)	–	0.3
L0 Path Reasoning		
Device Importance Weight	ω_d	0.1
Neighbor Node Similarity	ω_n	0.4
Edge Similarity Weight	ω_e	0.25
Community Similarity Weight	ω_c	0.5
Top Paths to Return	–	10

Table 4: Hyperparameter settings for the hierarchical retrieval process.

present in our knowledge graph.

Finally, the resulting vector $E_{\text{community}}(c)$ is L2-normalized to ensure it has a unit length. This weighted fusion approach allows the community vector to balance the community's core semantic definition with the actual semantics of its member entities, creating a robust and dynamic representation for retrieval.

B.2 Configuration of Hierarchical Retrieval

As described in the Hierarchical Retrieval section, our retrieval process consists of four core stages. This appendix provides the detailed hyperparameter configurations used in each stage. A summary of all key parameters is presented in Table 4.

Macro-summary Retrieval. This stage identifies relevant high-level themes from the L2 summary layer. The construction of the fused query vector, $E_{\text{query}}(k)$, as defined in Equation (3), is weighted by $\omega_q = 0.4$ and $\omega_k = 0.6$. To ensure relevance, a keyword is routed to a community only if the matching similarity exceeds a threshold of 0.3, and an L2 summary is retrieved only if its score is above 0.4.

Device Summary Retrieval. This stage calculates an importance weight, $W(v_d)$, for each candidate device based on the retrieved L2 summaries. As shown in Equation (4), this calculation is governed by the weights $\omega_{\text{freq}} = 0.4$ for normalized frequency and $\omega_{\text{qual}} = 0.6$ for average relevance score. Based on these weights, a total of $K_{\text{total}} = 10$ candidate devices are selected for the next step. The subsequent ranking of L1 summaries for these devices is performed using a separate scoring function that weighs semantic similarity at 0.7 and the device weight at 0.3.

Path Reasoning. This final retrieval stage scores one-hop reasoning paths from the candidate devices. The composite scoring function, detailed in Equation (5), integrates multiple semantic signals. The weights for this function are set as follows: device importance weight $\omega_d = 0.1$, neighbor node similarity $\omega_n = 0.4$, edge similarity $\omega_e = 0.25$, and community similarity $\omega_c = 0.5$. The top 10 paths are returned as the final graph-based evidence.

B.3 Baseline Model Configurations

To ensure a fair and reproducible comparison, this section details the key configurations for the LightRAG and GraphRAG baseline frameworks. For all experiments involving these baselines, key recall parameters (such as the number of retrieved entities or relationships) were consistently set to **10** to align with our method's settings.

B.3.1 LightRAG Configuration

For all LightRAG experiments, the set of extracted entity types was defined as "Device", "Package type", "Basic information", "Pin Configuration", "Parameters", "Manufacturer", and "Ordering information". These types were chosen to directly correspond to the predefined semantic communities used in our ElecGraph-RAG framework. The configurations for each retrieval mode are as follows:

Local Mode (LightRAG-Local). The local search retrieves relevant entities based on keyword similarity. The primary retrieval parameter, top k, was set to 10. The maximum token limits for entity context and the total query context were kept at their default values of 10,000 and 30,000, respectively.

Global Mode (LightRAG-Global). The global search focuses on retrieving relevant relationships. The top k for relationships was also set to 10. The cosine similarity threshold for vector search was the default value of 0.2, and token limits were kept at their default settings.

Hybrid Mode (LightRAG-Hybrid). The hybrid mode combines both local and global strategies.

For our experiments, the number of retrieved entities, relationships, and initial text chunks were all consistently set to 10.

B.3.2 GraphRAG Configuration

The initial knowledge graph for GraphRAG was built using the entity types "Device", "Manufacturer", "Package type", "Basic information", "Pin Configuration", "Parameters", and "Ordering Information". As with LightRAG, these entity types correspond to the semantic communities in our main framework.

Local Mode (GraphRAG-Local). Local search was configured to retrieve a balanced context consisting of the top 10 most relevant entities and the top 10 most relevant relationships. The proportional allocation of the context window for different information types was kept at the default setting.

Global Mode (GraphRAG-Global). Global search involves a map-reduce process over community summaries. We utilized the default configuration for our experiments, which sets the total context window size to 12,000 tokens and limits the search to a maximum depth of two levels within the community hierarchy.

B.4 Experimental Implementation Details.

In our experiments, all LLM operations for both ElecGraph-RAG and the baseline methods were implemented using the deepseek-chat model. We employed Qwen-text-embedding-v4 as the embedding model to encode all textual data.

To provide appropriately sized context for downstream tasks while preserving the document's semantic integrity, we designed and implemented a dynamic text chunking strategy. This process is applied to all datasheets and consists of the following steps:

1. **Structural Splitting:** First, we parse the Markdown document and split it into semantically coherent "structural blocks" based on its inherent structure (e.g., headings, paragraphs, tables). This initial step ensures that logically related content, such as an entire table, is not prematurely divided.

2. **Adaptive Chunk Size Calculation:** To avoid forcibly splitting large, important blocks (like a comprehensive parameter table), we dynamically calculate an adaptive target chunk size. This size is determined based on the length of the largest structural block found in the document, but is constrained within a predefined minimum and maximum range (e.g., 400 to 1000 tokens). This adaptability ensures that the chunk size is appropriate for the specific content of each datasheet.

3. **Greedy Block Combination:** Next, we greedily combine consecutive structural blocks into a set of initial, non-overlapping text chunks. This combination process continues until the inclusion of the next block would exceed the dynamically calculated target size. This step results in a series of chunks that are semantically complete.

4. **Structural Overlap Application:** Finally, to preserve contextual continuity between adjacent chunks, a structural overlap is applied. For each chunk (from the second chunk onwards), the framework prepends a portion of the preceding chunk. Critically, this overlap is not a fixed token count that could truncate a block mid-sentence. Instead, it is constructed by taking complete structural blocks from the end of the preceding chunk until the total size of the overlap reaches approximately 200 tokens. This method ensures that each chunk begins with a coherent context from the previous section, mitigating information loss at chunk boundaries.

B.5 Ablation Study Design

To quantitatively assess the contribution of each core module within the ElecGraph-RAG framework, we designed four distinct ablation variants. Each variant disables a specific component to isolate its impact on performance across the four query types.

B.5.1 w/o Summaries

This experiment aims to validate the value of the L1 Device Profile and L2 Cross-Device Thematic Summary layers in providing essential macro-level context and semantic generalization.

Methodology. In this mode, the L1 and L2 summary layers of the hierarchical index are completely disabled, forcing the entire retrieval process to operate directly on the L0 factual graph layer. The process is as follows: first, after query deconstruction, the overall query vector E_q is used to perform a global similarity search against all device entities

in the L_0 graph, identifying an initial set of top-10 candidate starting devices. Subsequently, for this candidate set, the framework conducts a one-hop path evaluation. The relevance of each path is scored using the fused query vector $E_{\text{query}}(k)$, as defined in Equation (3) of the main paper. Consequently, the final context provided to the LLM consists solely of L0 graph path evidence, without any summary information.

B.5.2 w/o Graph-Path

This experiment is designed to assess the contribution of the fine-grained, structured path evidence provided by the L0 factual graph layer in ensuring factual precision and logical interpretability.

Methodology. The retrieval pipeline in this mode—from intent deconstruction to L2 filtering and L1 drill-down—is identical to the full ElecGraph-RAG framework. The sole modification occurs in the final context construction stage, where the system is configured to ignore the results from the Path Reasoning step. As a result, the final context includes only the summary information from the L2 and L1 layers.

B.5.3 w/o L2-Filter

This experiment evaluates the core function of the L2 summary layer as an efficient candidate-set filter and focusing mechanism for improving both retrieval efficiency and precision.

Methodology. In this variant, the L2 summary layer is bypassed, forcing the initial retrieval to occur on the much larger and noisier L1 layer. The fused query vectors are used to perform a global top-10 vector search across an aggregated pool containing all L1 summaries from all devices. The results of this broad search are then used to reverse-engineer device relevance and compute their contextual weights, which in turn guide the final L0 path reasoning stage.

B.5.4 w/o Intent

This experiment tests the necessity of the LLM-based intent deconstruction module, particularly for handling multifaceted and complex queries.

Methodology. In this mode, the step of extracting multiple keywords via the LLM is disabled. Instead of using a cluster of keyword vectors, the entire retrieval process relies on a single vector derived from the original, un-decomposed user query.

Hyperparameters		Accuracy (%)	
ω_{qual}	ω_{freq}	**Precise Fact**	**Comp. Analysis**
0.1	0.9	67.14	51.67
0.3	0.7	72.86	60.00
0.5	0.5	70.00	63.33
0.6	**0.4**	**94.28**	**85.00**
0.7	0.3	71.43	63.33
0.9	0.1	71.43	60.00

Table 5: Sensitivity analysis of device anchoring weights on retrieval accuracy. The default configuration ($\omega_{\text{qual}} = 0.6, \omega_{\text{freq}} = 0.4$) yields the highest performance.

This single vector is used for all subsequent similarity calculations, including L2 community routing, L1 summary retrieval (top-10), and L0 path scoring (top-10). This forces the system to understand and retrieve information from a single, un-differentiated semantic perspective.

B.6 Hyperparameter Sensitivity Analysis

To determine the optimal configuration for the device anchoring mechanism described in Equation (4), we conducted a sensitivity analysis on the weighting parameters ω_{qual} (Quality/Relevance Weight) and ω_{freq} (Frequency Weight).

We evaluated the framework's performance on the "Precise Fact" and "Comparative Analysis" query sets under varying weight combinations, subject to the constraint $\omega_{\text{qual}} + \omega_{\text{freq}} = 1.0$. The results, summarized in Table 5, demonstrate a distinct performance peak.

Analysis of Results. The data reveals that the retrieval performance is highly sensitive to the balance between semantic relevance and structural centrality.

- **Dominance of Frequency ($\omega_{\text{freq}} > 0.5$):** When the frequency weight is too high (e.g., 0.9), the system tends to retrieve generic or "hub" devices that appear frequently in the dataset but may not be semantically specific to the user's query, leading to a significant drop in accuracy (e.g., 67.14% on factual queries).

- **Dominance of Quality ($\omega_{\text{qual}} > 0.6$):** Conversely, relying too heavily on semantic similarity alone (e.g., $\omega_{\text{qual}} = 0.9$) increases the risk of selecting devices that mention the query keywords purely by chance or in a tangential context, lacking the structural evidence of being a core topic.

- **Optimal Balance:** The configuration of $\omega_{\text{qual}} = 0.6$ and $\omega_{\text{freq}} = 0.4$ achieves a synergy where semantic relevance guides the search, while frequency acts as a structural anchor to confirm the device's importance. This setting provides a substantial accuracy gain ($+24.28\%$ on Precise Fact queries compared to the 0.5/0.5 baseline), validating its selection as the default for our framework.

C Dataset Composition

C.1 ElecGraph-KG: Knowledge Graph Source Data

Our knowledge base, **ElecGraph-KG**, was constructed to reflect real-world industry scenarios. The source corpus consists of 70 official electronic device datasheets collected from various industry-leading manufacturers. To ensure data source diversity, the corpus includes 50 datasheets for single devices and 20 for device series, spanning 20 distinct device categories from 18 leading semiconductor manufacturers. A representative sample of the datasheets is shown in Table 6.

Due to copyright restrictions associated with the official datasheets, the source PDF files for **ElecGraph-KG** cannot be publicly released.

C.2 ElecGraph-QA: Question Answering Benchmark

To address the lack of standardized benchmarks for complex question answering in the electronics domain, we constructed the **ElecGraph-QA** evaluation set.

The benchmark was generated using a Large Language Model (LLM) instructed to adopt four pre-defined domain expert personas: a Hardware Engineer, a Procurement Specialist, an Embedded Developer, and a Test & Validation Engineer. In each generation round, the LLM was provided with the text content from 5 randomly sampled datasheets as context. This process ensures the questions are both grounded in real data and representative of authentic application scenarios, with sufficient depth and breadth.

The final benchmark consists of 200 questions, which are categorized as follows:

- **70 Precise Fact Queries**, which ask about specific parameters for a single device.
- **60 Comparative Analysis Queries**, which require comparing specific performance aspects of two or more devices.
- **40 Multi-hop Reasoning Queries**, whose answers must be inferred through relationship chains in the knowledge graph.
- **30 Macro-summary Queries**, which target global questions about a device category or a specific topic.

The full **ElecGraph-QA** benchmark, including questions and manually reviewed ground-truth answers, will be made available as part of the supplementary material for this paper.

D Comparative Analysis of Framework Outputs

D.1 Comparison of Knowledge Base Structures

Based on the same corpus of 70 datasheets, we constructed knowledge bases using our proposed method as well as the two baseline frameworks. The following statistics highlight the significant differences in their resulting structure and scale.

ElecGraph-RAG. Using the device-centric construction methodology, our framework generated a highly structured and detailed knowledge base with the following statistics:

- **Entities:** 13,005
- **Edges:** 17,733
- **L1 Device Summaries:** 4,900
- **L2 Cross-Device Summaries:** 308

GraphRAG. The knowledge base constructed by the GraphRAG framework, using the entity types specified in Appendix B.3, consists of the following components:

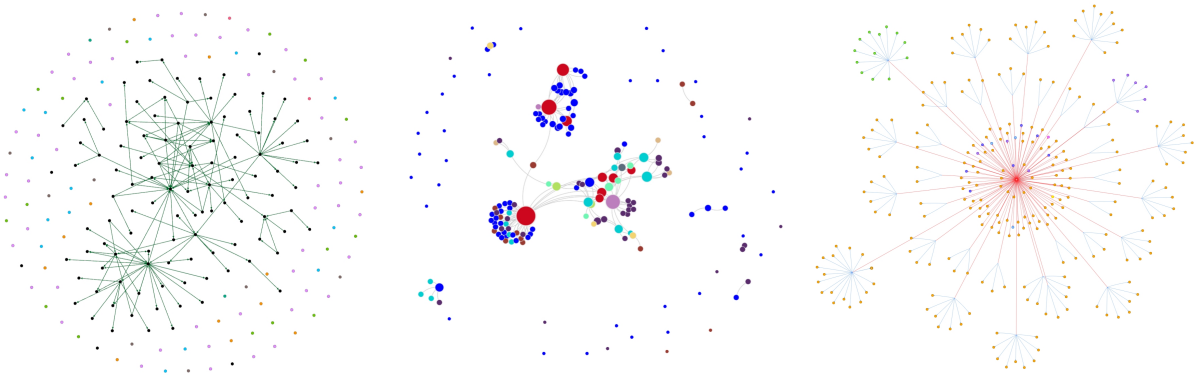
- **Entities:** 2,884
- **Relationships:** 3,705
- **Community Reports:** 427
- **Text Units:** 473

LightRAG. The knowledge base generated by the LightRAG framework, using the entity types specified in Appendix B.3, has the following statistics:

- **Entities:** 6,844
- **Edges:** 7,013
- **Text Chunks:** 473

Datasheet	Device Count	Manufacturer	Category
LDF	1	STMicroelectronics N.V.	LDO Regulator
SGT120R65AL	1	STMicroelectronics N.V.	GaN Device
STPS5L60SFY	1	STMicroelectronics N.V.	Schottky Diode
1N6642U	1	STMicroelectronics N.V.	Switching Diode
L2720W	1	STMicroelectronics N.V.	Op-Amp
STPSC8H065-Y	1	STMicroelectronics N.V.	Schottky Diode
UC3611M	2	Texas Instruments	Diode Array
TMAG6180-Q1	1	Texas Instruments	Angle Sensor
LMG3100R017	1	Texas Instruments	GaN Device
EZX84WC15V	4	Texas Instruments	Zener Diode
CSD75208W1015	1	Texas Instruments	MOSFET
UD1006FR	1	ON Semiconductor	Rectifier Diode
ON-J111	6	ON Semiconductor	JFET
NCV8614B	1	ON Semiconductor	LDO Regulator
NXP-X3T-OH048	4	NXP Semiconductors	Angle Sensor
BF904	2	NXP Semiconductors	MOSFET
PMEFJ112	3	NXP Semiconductors	FET
IRG4PH50S	1	Infineon	IGBT
TLV4976-2K	1	Infineon	Hall Effect Switch
IMT65R048M1H	1	Infineon	GaN Device
ADPD2140	1	Analog Devices, Inc.	Angle Sensor
LTC1555L	1	Analog Devices, Inc.	Latch
HMC843	1	Analog Devices, Inc.	SLogic Gate
ADI-MAX31827	1	Maxim Integrated	Temperature Sensor
APD Series	36	Isolink, Inc.	Switching Diode

Table 6: A Representative Sample of Datasheets in the ElecGraph-KG Corpus



(a) GraphRAG (b) LightRAG (c) ElecGraph-RAG

Figure 12: A visual comparison of knowledge graph structures generated for the LM392 datasheet. (a) The graph from GraphRAG shows a web-like structure. (b) The graph from LightRAG presents another relational structure. (c) Our ElecGraph-RAG framework generates a distinct, device-centric star-shaped topology.

D.2 Visual Comparison of Knowledge Graph Topologies

To visually illustrate the fundamental structural differences between the knowledge bases, Figure 12 presents the graph topologies generated by the three frameworks for the same LM392 datasheet.

GraphRAG. As shown in Figure 12(a), the topology generated by GraphRAG is, similar to LightRAG, a multi-centric and clustered structure. While several prominent hub nodes form local clusters, a significant number of entities remain completely disconnected as isolated nodes scattered throughout the graph space. This indicates that the generic extraction process fails to establish relationships for many entities. Consequently, retrieving a complete context for a single device is challenging, as information is fragmented across multiple, disconnected clusters and numerous isolated entities, leading to a high risk of information loss.

LightRAG. Similar to GraphRAG, the topology generated by LightRAG, shown in Figure 12(b), is also a multi-centric and clustered structure. The graph features several prominent hub nodes (e.g.,

Question	*Recommend a suitable IGBT module (600V, 50A, low thermal resistance) for an industrial robot servo driver, and explain its switching loss and packaging characteristics.*
GraphRAG-Local	Recommends the **IMT65R048M1H**, correctly identifying its voltage rating. However, it incorrectly states the continuous current is 15A and suggests a parallel design to meet the 50A requirement. The answer provides a very detailed analysis of the device's low switching loss (due to CoolSiC technology), its thermally optimized packaging, and is supplemented with extensive application suggestions [...].
ElecGraph-RAG	Also recommends the **IMT65R048M1H**. It correctly identifies the continuous current rating as 50A at 25°C, directly meeting the user's requirement. The answer is presented as a highly concise, structured list of key specifications covering the device's voltage/current capability, thermal resistance, packaging, and switching performance, concluding with a practical heat dissipation recommendation.
LLM Decision (Run 1: GraphRAG-Local vs. ElecGraph-RAG)	
	• **Relevance & Completeness: Winner = GraphRAG-Local**. Reasoning: The answer is more comprehensive, providing alternative solutions and application advice. • **Logical Structure & Clarity: Winner = ElecGraph-RAG**. Reasoning: The bulleted list structure is clearer and easier to read. • **Insightfulness & Depth: Winner = GraphRAG-Local**. Reasoning: The answer explains the meaning of the parameters in the application context and provides a life expectancy estimate.
LLM Decision (Run 2: ElecGraph-RAG vs. GraphRAG-Local)	
	• **Relevance & Completeness: Winner = ElecGraph-RAG**. Reasoning: The answer is closer to the question and completely addresses all user requests. • **Logical Structure & Clarity: Winner = ElecGraph-RAG**. Reasoning: The organization is clear and the bulleted list is easy to understand. • **Insightfulness & Depth: Winner = ElecGraph-RAG**. Reasoning: The answer provides practical advice such as heat dissipation recommendations and highlights the benefits of the SiC technology.

Table 7: An example of head-to-head evaluation for a multi-hop reasoning query, including the two generated answers and the full results from the bias-mitigation evaluation runs.

for the device model and manufacturer) that form local clusters. However, attributes are not anchored to a single, central device entity. Instead, information is distributed across these different clusters, and numerous entities exist as disconnected, isolated nodes. This fragmentation can lead to an incomplete context during retrieval for a specific device.

ElecGraph-RAG. In sharp contrast, our framework, shown in Figure 12(c), generates a distinctly hierarchical and device-centric star-shaped topology. The central node (red) represents the core device entity (LM392). The smaller star-like substructures visible in the diagram are a result of our edge generalization strategy. The centers of these small stars (e.g., parameter types, shown in orange) are not actual data nodes; they are visual convergence points for edges sharing the same generalized relationship type (e.g., "has Forward Voltage"). In the underlying data, all attribute nodes connect directly and exclusively to the single, central device entity. This architecture ensures that a simple one-hop query from the device node is sufficient to retrieve all its associated information, drastically simplifying query complexity and eliminating the risk of entity isolation that can occur in more complex graph structures.

D.3 Qualitative Comparison of Generated Answers

To illustrate the practical difference in output quality, Table 7 presents a concrete example of a head-to-head evaluation for a complex, multi-hop reasoning query. This comparison showcases how the underlying knowledge base structure impacts the relevance, clarity, and depth of the final generated answer.