HyperSU: Corpus-Driven Semantic-Unit Hypergraph for Retrieval-Augmented Generation

Jiate Liu[1*] **Liuyi Chen**[2*] **Zhengyi Yang**[3†] **Chuan He**[1]
Mingchen Ju[4] **Bocheng Han**[5] **Ruyi Liu**[1] **Xu Zhou**[2]
[1]The University of New South Wales, Sydney, Australia [2]Hunan University, Changsha, China
[3]The University of Sydney, Sydney, Australia [4]Euler AI [5]Vecton AI
{jiate.liu, chuan.he3, ruyi.liu2}@unsw.edu.au {liuyi.chen, zhxu}@hnu.edu.cn
zhengyi.yang@sydney.edu.au mingchen.ju@eulerai.au bocheng.han@vectonai.com

Abstract

Recent Hypergraph-based retrieval-augmented generation (HyperRAG) methods use hyperedges to connect multiple entities simultaneously, enabling more efficient multi-entity evidence organization than pairwise graph structures. However, existing HyperRAG methods often rely on LLM-generated summaries to construct hyperedges, which can introduce hallucinations while also incurring high indexing costs. In addition, during retrieval, existing methods typically rely on either one-hop neighbor expansion or PageRank diffusion. The former may miss useful multi-hop evidence, while the latter can suffer from uncontrolled propagation over excessive hub nodes, leading to semantic drift and noisy reasoning chains. To address these challenges, we propose **HyperSU**, a novel hypergraph-based RAG framework featuring semantic-unit hyperedges and clue-guided bidirectional retrieval. During construction, **HyperSU** formulates hyperedge construction as an entity-aware minimum-description-length (MDL) optimization problem, inducing source-grounded semantic-unit hyperedges that balance sentence-level semantic coherence and entity compactness. It then constructs a hypergraph by modeling each semantic unit as a hyperedge over its co-mentioned entities. During retrieval, HyperSU performs clue-guided bidirectional expansion over the semantic-unit hypergraph, enabling both multi-hop evidence discovery and evidence-anchored noise reduction. Experiments show that **HyperSU** consistently improves answer accuracy over standard, graph-based, and hypergraph-based RAG baselines, raising average accuracy on GraphRAG-Bench by 5.5 points over the strongest baseline (68.7 vs. 63.2), with larger gains on reasoning-intensive tasks, where the relative improvement reaches 14.1%. Our code is available at https://github.com/RAIDSLab/HyperSU.

[*] Equal contribution.
[†] Corresponding author.

1 Introduction

Retrieval-augmented generation (RAG) enhances large language models by grounding generation on retrieved external knowledge. (Lewis et al., 2020; Gao et al., 2024). However, traditional retrieval methods often struggle to capture complex multi-hop relationships across dispersed evidence. To address this limitation, recent graph-based RAG methods (Edge et al., 2024a; Guo et al., 2025; Gutiérrez et al., 2024) organize corpus knowledge as graphs to support multi-hop retrieval and reasoning. Since each edge connects only two entities, graph-based methods are limited to binary relations and struggle to model common multi-entity relations in real-world knowledge. Hypergraph-based RAG methods address this limitation by using hyperedges to connect multiple related entities simultaneously.

As illustrated in Figure 1(a)-(b), consider the query: *"Which company developed the AI that defeated the world champion of Go?"*

RAG retrieves independent chunks without bridging evidence, introducing noisy context (*Ke Jie* and *Go*) while missing critical connecting evidence such as *DeepMind*. GraphRAG improves retrieval by organizing knowledge as relational graphs. However, each edge connects only two entities, a multi-entity relation must be decomposed into multiple binary triples linked through shared hub entities such as *Seoul*, fragmenting a coherent n-ary relation into multiple pairwise connections.

Existing HyperRAG methods (Luo et al., 2025a; Feng et al., 2026), however, commonly rely on LLM-generated hyperedges and often retrieve through one-hop neighboring expansion or PageRank (Page et al., 1999) diffusion. This design still has two limitations. First, generated hyperedges may contain unsupported or incomplete relations. For example, in Figure 1(c), hyperedge H_2 incorrectly includes *Shane Legg* and *Mustafa Suleyman*,

Findings of the Association for Computational Linguistics: EMNLP 2026, pages 22399–22424
October 24-29, 2026 ©2026 Association for Computational Linguistics

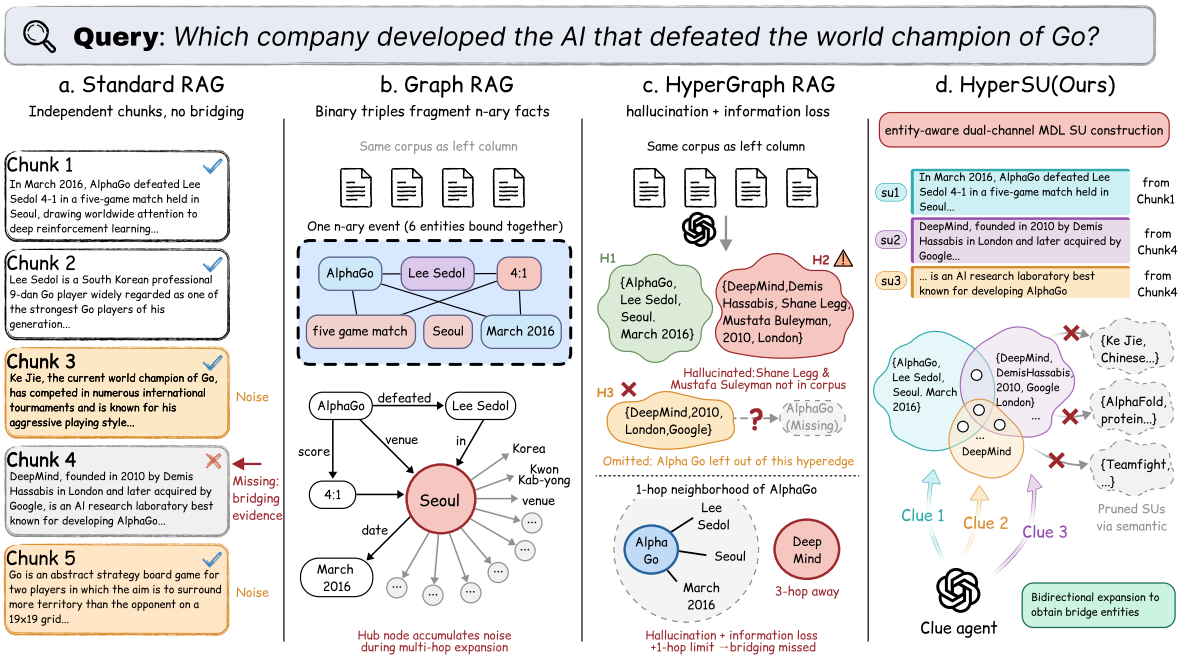

Figure 1: Comparison of RAG paradigms on a multi-hop QA example. Unlike Standard RAG, GraphRAG, and existing hypergraph-based RAG, HyperSU builds source-grounded semantic-unit hyperedges and retrieves bridging evidence through clue-guided bidirectional expansion.

while H_3 omits the key entity *AlphaGo*. Second, one-hop expansion can miss distant but necessary bridge evidence, while unrestricted PageRank diffusion may introduce excessive noisy evidence and semantic drift. In this example, *DeepMind* remains outside the one-hop neighborhood of *AlphaGo*, leaving the reasoning chain incomplete.

These limitations motivate us to develop a framework that constructs hyperedges that **reduce dependence on LLM-generated summaries**, while supporting **effective retrieval** for more complete and reliable answers.

To achieve this goal, we propose a novel framework, **HyperSU**. During construction, HyperSU formulates semantic-unit construction as a minimum description length (MDL) optimization problem (Rissanen, 1978; Grünwald, 2007) that jointly optimizes sentence-level semantic coherence and entity compactness. During retrieval, HyperSU decomposes the query into multiple clues and performs clue-guided bidirectional expansion over the semantic-unit hypergraph. Forward expansion discovers distant bridge entities to improve multi-hop recall, while backward anchoring verifies forward-reached evidence against dense-retrieved candidates to suppress semantic drift. The activated SUs are finally mapped back to their source passages for ranking and answer generation.

HyperSU constructs a semantically coherent and entity-compact hypergraph that reduces redundant and noisy evidence. Meanwhile, its retrieval strategy improves evidence coverage while controlling irrelevant propagation. As shown in Figure 1(d), each semantic unit (SU) captures a coherent event or factual context. For example, the hyperedge marked in blue, {*AlphaGo*, *Lee Sedol*, *Seoul*, *March 2016*}, describes the *AlphaGo–Lee Sedol* match held in *Seoul* in *March 2016*. Shared entities marked in white naturally connect related SUs, forming an entity-compact hypergraph with strong semantic associations across dispersed evidence. During retrieval, query-based clues guide bidirectional exploration over the hypergraph, while irrelevant SUs such as *Ke Jie*, *AlphaFold* and *Teamfight* are effectively pruned to suppress noisy propagation.

Our main contributions are as follows:

- We propose **HyperSU**, a corpus-driven HyperRAG framework built on semantic units. HyperSU constructs a semantically coherent hypergraph with compact entity connections, where hyperedges are grounded in complete and faithful corpus evidence.
- We design a clue-guided bidirectional hypergraph retrieval algorithm that combines fault-tolerant multi-clue SU activation, recall-oriented forward expansion, precision-oriented backward anchoring, and convergence verification, enabling reli-

able bridge-evidence discovery while suppressing semantic drift in multi-hop reasoning.

- Experiments on GraphRAG-Bench, HotpotQA, 2WikiMultiHopQA, and MuSiQue show that HyperSU consistently improves over standard, graph-based, and hypergraph-based RAG baselines, raising average GraphRAG-Bench accuracy by 5.5 points over the strongest baseline (68.7 vs. 63.2), with a maximum relative gain of 14.1% on a single setting, and obtaining the best GPT-Accuracy across three multi-hop QA datasets. HyperSU further constructs hyperedges with zero offline LLM indexing cost, requiring 0 indexing tokens and $0 API cost.[1]

2 Related Work

Graph-based RAG Graph-based RAG methods augment retrieval by organizing corpus evidence into structured representations. Existing systems differ in how they build and traverse such structures: community-summary methods construct graph communities and summaries (Edge et al., 2024a,b); PageRank- or traversal-based methods expand from query-linked entities (Guo et al., 2025; Circlemind AI, 2024; Gutiérrez et al., 2024, 2025); learned, prompted, or agentic retrievers use neural or LLM-guided mechanisms to select relevant subgraphs (Luo et al., 2025b; Wang et al., 2024; He et al., 2024; Liu et al., 2026); and hierarchical, relation-free, or hybrid methods organize evidence beyond standard binary KG retrieval (Sarthi et al., 2024; Zhuang et al., 2026; Zhao et al., 2025). While these methods improve structured retrieval, binary or summary-based representations may fragment multi-entity evidence or require costly LLM-based indexing. A detailed comparison is provided in Appendix A.

Hypergraph-based RAG Hypergraphs are widely used to represent higher-order relations in graph learning and analytics (Luo et al., 2024; Li et al., 2026; Pan et al., 2025). Hypergraph-based RAG methods model higher-order evidence with hyperedges rather than pairwise graph edges. HyperGraphRAG (Luo et al., 2025a) constructs local n-ary relational hyperedges from pre-segmented chunks, while Hyper-RAG (Feng et al., 2026) models higher-order correlations through hypergraph-driven retrieval. However, these hyperedges or correlations are typically generated from chunk-level text by LLMs, which can introduce unsupported entities, omit source-critical bridge entities, and increase offline indexing cost. HyperSU instead induces source-grounded semantic-unit hyperedges from corpus spans and composes them at query time through clue-guided bidirectional expansion. Because each hyperedge is a source span, retrieved evidence remains traceable to the original text, a property recent work argues data systems should provide by design (Yang et al., 2026).

[1]Throughout the paper, zero LLM indexing cost means zero offline generative-LLM API calls and tokens. Indexing still runs GLiNER, an encoder-based NER model, locally, and its wall-clock cost is reported in §5.5; an LLM is used online by the Clue Agent, whose per-query cost is also reported there.

3 Preliminaries

Definition 1: Retrieval-Augmented Generation. Given a question q and a passage corpus $\mathcal{C}$, retrieval-augmented generation (RAG) retrieves a subset of passages $\mathcal{P}^* \subseteq \mathcal{C}$ and generates an answer y conditioned on $(q, \mathcal{P}^*)$:

$$P(y \mid q) = \sum_{\mathcal{P}^* \subseteq \mathcal{C}} P(y \mid q, \mathcal{P}^*)\, P(\mathcal{P}^* \mid q, \mathcal{C}).$$

Definition 2: Hypergraph. A hypergraph $G_H = (V, E_H)$ generalizes a graph by allowing each hyperedge $e_H \in E_H$ to connect a set of entities $V_{e_H} = \{v_1, v_2, \ldots, v_n\}$, where $n \geq 2$. It can be represented by an incidence matrix $H \in \{0,1\}^{|V| \times |E_H|}$, where $H[v, e_H] = 1$ iff $v \in V_{e_H}$. Through star expansion (Agarwal et al., 2006; Zhou et al., 2006; Sun et al., 2008), G_H can be equivalently encoded as an entity–hyperedge bipartite graph whose biadjacency matrix is exactly H; we detail this correspondence in Appendix B.2.

We use this formalism to represent evidence units that involve more than two co-mentioned entities. Unlike the pairwise entity–entity projection, a hyperedge retains the joint membership of all entities associated with the same source-grounded evidence unit.

Proposition 1 *For a source-grounded multi-entity evidence unit e with entity set V_e, representing e as a hyperedge preserves the unit-level membership and provenance of its co-mentioned entities. In contrast, projecting e onto pairwise entity–entity edges may lose the original group identity unless auxiliary event or provenance nodes or labels are added.*

We provide the formal proof in Appendix B.1.

4 Method

HyperSU consists of two stages: offline semantic-unit hypergraph construction and online clue-

guided retrieval. As shown in Figure 2, the offline stage first extracts sentence-level entity mentions, induces compact semantic units (SUs) through entity-aware MDL segmentation, and assembles the semantic-unit hypergraph $G_H = (V, E_H, H)$. The online stage composes these local evidence units into query-specific reasoning chains: it maps query entities to the hypergraph, derives parallel clues for SU activation, performs bidirectional frontier expansion, and verifies convergent evidence before projecting activated SUs back to passages for ranking and answer generation. We describe offline construction in Section 4.1 and online retrieval in Section 4.2.

4.1 Offline Indexing: Semantic-Unit Hypergraph Construction

Given a corpus $\mathcal{C}$, HyperSU constructs the hypergraph through sentence-level entity extraction, entity-aware SU induction, and assembly.

Entity Extraction. For each passage $p = (s_1, \ldots, s_n)$, HyperSU first performs sentence-level entity extraction using GLiNER (Zaratiana et al., 2024), a generalist named entity recognition model. The extracted mentions are normalized into canonical entity names and used to construct the entity-count channel for SU induction.

Definition 3: Semantic Unit. Given a passage $p = (s_1, s_2, \ldots, s_n)$, a *semantic unit* (SU) is a contiguous, semantically coherent, self-contained evidence span $u = (s_i, \ldots, s_j)$. HyperSU represents each SU as a source-grounded local evidence unit and induces a hyperedge $e_u \in E_H$ over its co-mentioned entities, with $V_u = \{v \in V : v \text{ appears in } u\}$. SU hyperedges are reused and composed during retrieval to form query-specific reasoning chains, rather than pre-materialized as predicate-level n-ary facts. The construction is formalized as an MDL segmentation problem in §4.1.

Entity-Aware SU Induction. Let $(\mathbf{x}_1, \ldots, \mathbf{x}_n)$ denote L2-normalized sentence embeddings. For each passage p, HyperSU partitions sentences into contiguous, non-overlapping semantic units $\mathcal{U}_p = u_1, \ldots, u_k$. We formulate this segmentation as an entity-aware dual-channel MDL problem: the semantic channel encourages coherent sentence grouping, while the entity channel favors compact spans whose entity mentions form focused evidence units.

For a candidate segment $u = (s_a, \ldots, s_b)$, the semantic channel uses the embedding resultant length $R_u = \left\|\sum_{i=a}^{b} \mathbf{x}_i\right\|$. Let $c_u(v)$ be the count of normalized entity name v in segment u, $N_u = \sum_v c_u(v)$, $U_u = \{v : c_u(v) > 0\}$, and $\pi_u(v) = c_u(v)/N_u$. The entity channel penalizes spans that mix many weakly related entity mentions:

$$\Phi(u) = N_u \mathcal{H}(\pi_u) + \frac{|U_u| - 1}{2} \log N_u, \quad (1)$$

where the entity penalty is set to zero when $N_u = 0$. The segment reward is

$$r(u) = \kappa R_u - \Phi(u) - \frac{d_{\text{eff}} - 1}{2} \log n. \quad (2)$$

The objective is

$$\mathcal{Q}_{\text{MDL}}(\mathcal{U}_p) = \sum_{u \in \mathcal{U}_p} r(u), \quad (3)$$

subject to $w_{\min} \leq \text{words}(u) \leq w_{\max}$. Here κ controls the semantic reward scale and d_{eff} is the effective semantic dimension used in the MDL complexity term. The derivation of this entity-aware dual-channel MDL objective is in Appendix B.3.

This objective keeps semantically coherent sentences together while discouraging segments whose entity mentions are diffuse across unrelated topics. The semantic term rewards groups of sentences with consistent embedding directions, while the entity term can be viewed as the coding cost of the empirical entity-mention distribution within a segment. The MDL complexity term penalizes excessive segmentation. The resulting SUs are compact text spans suitable for retrieval that induce useful hyperedges over co-occurring entities.

Dynamic Programming The optimal partition is found by one-dimensional dynamic programming. Let $\text{dp}[i]$ be the best score for prefix $(s_1, \ldots, s_i)$:

$$\text{dp}[i] = \max_{j \in \mathcal{J}(i)} \left\{\text{dp}[j] + r(s_{j+1}, \ldots, s_i)\right\},$$

where $\mathcal{J}(i)$ contains feasible boundaries satisfying the word-count constraints. Sentence embedding prefix sums and sentence-level entity counts allow each candidate segment to be scored from its boundary statistics, and the optimal partition is recovered by backtracking through parent pointers.

Hypergraph Assembly. After SU induction, HyperSU canonicalizes entity mentions into entity nodes using exact matching over normalized names and entity labels, optionally followed by a high-threshold embedding merge for near-identical mentions. For each semantic unit u, let V_u be the set

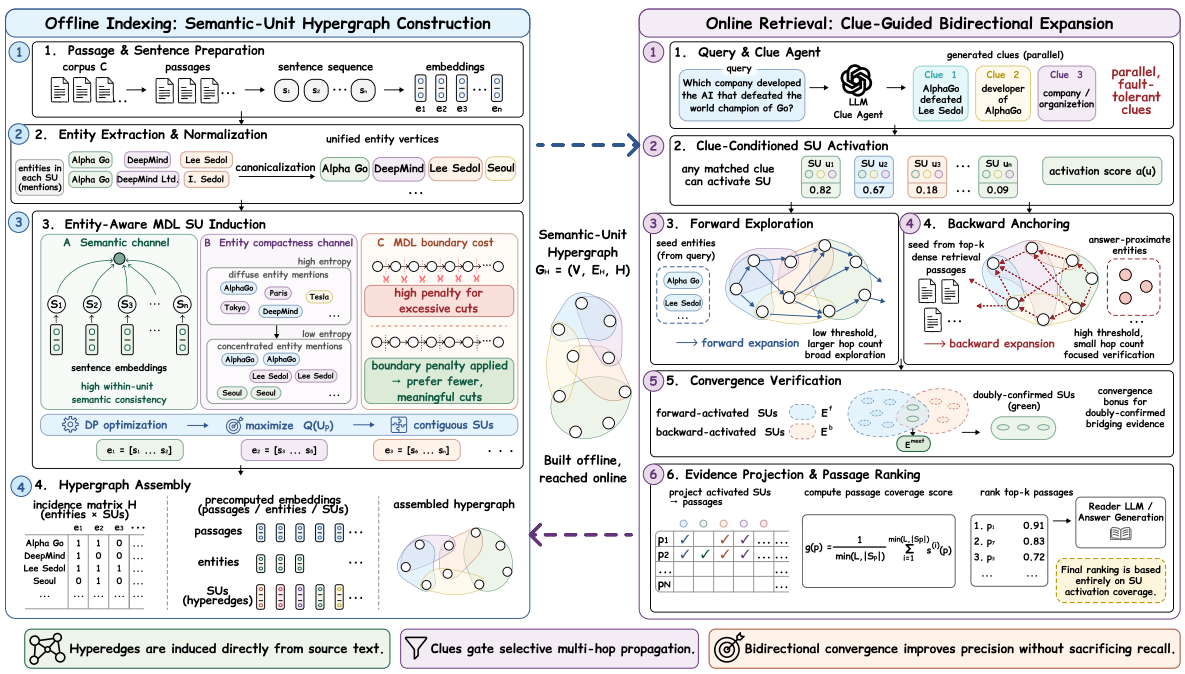

Figure 2: Overview of HyperSU. Offline indexing builds a source-grounded entity–SU hypergraph, while online retrieval activates, expands, and ranks SUs through clue-guided bidirectional retrieval.

of entities it contains. HyperSU constructs the hypergraph $G_H = (V, E_H, H)$ by treating each SU as a hyperedge $e_u \in E_H$ over V_u, with incidence matrix $H[v, u] = 1$ iff entity v appears in u.

4.2 Online Retrieval: Clue-Guided Bidirectional Expansion

At retrieval time, HyperSU extracts query entities with the same GLiNER-based extractor and maps them to indexed canonical entities by exact name matching or embedding similarity. It then retrieves cross-passage evidence through clue-conditioned SU activation, bidirectional frontier expansion, and SU-based passage scoring. This design enables forward expansion to reach distant bridge entities while using clue gating and backward anchoring to suppress noisy propagation.

Clue Agent and SU Activation. A Clue Agent decomposes the query into parallel clues $\mathcal{C}_q = \{c_1, \ldots, c_M\}$. If no planner is used, the original query embedding acts as the single clue. For each SU u, HyperSU computes

$$\alpha(u) = \mathrm{clamp}\left(\frac{\max_i \mathrm{sim}(c_i, u) - \delta}{1 - \delta}, 0, 1\right)^{\gamma},$$

where $\mathrm{sim}(c_i, u)$ is cosine similarity, δ is the SU activation threshold, and γ controls sharpening. The max over clues activates an SU when any clue matches it well. This provides a fault-tolerant activation mechanism: retrieval does not require all generated clues, or a complete multi-hop plan, to be correct. Even if some clues are noisy, a single well-matched clue can still activate relevant SUs and trigger expansion, which is especially useful for complex multi-hop questions that require several pieces of evidence.

Bidirectional Frontier Expansion. Both directions use the same propagation rule. At hop t, an unactivated entity $v \notin A_{t-1}$ receives

$$c_t(v) = \max_{u:\, v \in V_u} \left[\alpha(u) \cdot \max_{z \in F_t \cap V_u} a(z)\right],$$

where F_t is the current frontier, A_{t-1} is the activated entity set, and $a(z)$ is the entity activation score. HyperSU keeps the top-K new entities and applies hop decay, $a(v) = c_t(v)\beta^t$.

For each direction, HyperSU records the best SU bridge score across hops:

$$b(u) = \max_t \left[\alpha(u) \cdot \max_{z \in F_t \cap V_u} a(z)\right].$$

We compute b^{fwd} and b^{bwd} using the forward and backward frontiers, respectively.

Forward expansion starts from query-linked entities and explores multiple hops to improve recall. Backward anchoring starts from entities in top dense-retrieved passages and verifies forward-reached evidence against those candidates. SUs activated by both directions form $\mathcal{U}_{\mathrm{meet}} =$

$\mathrm{supp}(b^{\mathrm{fwd}}) \cap \mathrm{supp}(b^{\mathrm{bwd}})$ and receive a convergence bonus:

$$s(u) = \begin{cases} b^{\mathrm{fwd}}(u) \cdot \mu, & u \in \mathcal{U}_{\mathrm{meet}}, \\ b^{\mathrm{fwd}}(u), & u \in \mathrm{supp}(b^{\mathrm{fwd}}) \setminus \mathcal{U}_{\mathrm{meet}}. \end{cases}$$

Backward-only SUs are not directly ranked; they are used only to verify and boost forward-reached evidence. This prevents dense-retrieved seeds from dominating the final ranking while still using them for candidate-evidence anchoring.

Evidence Projection and Passage Ranking. Activated SU evidence is projected back to passages. For passage p, let $S_p = \{u : u \subseteq p,\ s(u) > 0\}$ be its activated SUs. HyperSU scores p by the capped mean of its top-L SU scores:

$$g(p) = \frac{1}{\min(L, |S_p|)} \sum_{i=1}^{\min(L,|S_p|)} s^{(i)}(p),$$

where $s^{(i)}(p)$ is the i-th highest SU score in p. If S_p is empty, we set $g(p) = 0$. The top-ranked passages are passed to the reader LLM for answer generation. Dense retrieval is used only to initialize backward anchoring.

5 Experiments

We evaluate HyperSU on GraphRAG-Bench and standard multi-hop QA benchmarks, together with component ablations and efficiency analyses. These experiments test whether source-grounded SU hyperedges and clue-guided bidirectional expansion improve generation accuracy and retrieval quality while keeping indexing and retrieval practical.

5.1 Experimental Setup

Datasets We evaluate on two groups of benchmarks; their statistics are shown in Table 1.

GraphRAG-Bench. GraphRAG-Bench (Xiang et al., 2026) evaluates end-to-end RAG across two domains, *Novel* and *Medical*, and four task types: Fact Retrieval, Complex Reasoning, Contextual Summarization, and Creative Generation.

Multi-hop QA. Following Zhuang et al. (2026), we use 1,000 questions each from HotpotQA (Yang et al., 2018), 2WikiMultiHopQA (Ho et al., 2020), and MuSiQue (Trivedi et al., 2022), which together cover reasoning chains ranging from two to four hops.

Baselines We compare HyperSU with three groups of methods. *Standard RAG*: dense retrieval of Top-K passages. *Graph-based RAG*: MS-GraphRAG (Edge et al., 2024a), HippoRAG (Gutiérrez et al., 2024), HippoRAG 2 (Gutiérrez et al., 2025), LightRAG (Guo et al., 2025), RAPTOR (Sarthi et al., 2024), LinearRAG (Zhuang et al., 2026), KGP (Wang et al., 2024), G-retriever (He et al., 2024), E^{2}GraphRAG (Zhao et al., 2025), and GFM-RAG (Luo et al., 2025b). *Hypergraph-based RAG*: HyperGraphRAG (Luo et al., 2025a) and Hyper-RAG (Feng et al., 2026).

Table 1: Statistics of evaluation datasets. GraphRAG-Bench spans two domains and four task types; multi-hop QA datasets contain 1,000 questions each. Corpus size is measured in characters.

Subset	Task Type	#Q	Corpus
GraphRAG-Bench (Xiang et al., 2026)			
Novel	Fact Retrieval	971	4.82 M
	Complex Reasoning	610	
	Contextual Summ.	362	
	Creative Generation	67	
Medical	Fact Retrieval	1098	1.05 M
	Complex Reasoning	509	
	Contextual Summ.	289	
	Creative Generation	166	
Multi-hop QA (Zhuang et al., 2026)			
HotpotQA	Bridge	811	6.10 M
	Comparison	189	
2WikiMH	Compositional	413	2.92 M
	Comparison	244	
	Bridge Comp.	235	
	Inference	108	
MuSiQue	2-hop	518	6.36 M
	3-hop	316	
	4-hop	166	

Evaluation Metrics For GraphRAG-Bench, we follow the official task-specific protocol: Accuracy and ROUGE-L for Fact Retrieval and Complex Reasoning, Accuracy and Coverage Score for Contextual Summarization, Accuracy, Faithfulness Score, and Coverage Score for Creative Generation, and Context Recall and Context Relevance for retrieval. For multi-hop QA, we report Contain Accuracy and GPT-evaluated Accuracy following Wang et al. (2026). To mitigate potential dependence on LLM-as-judge evaluation, Appendix G.1 further reports deterministic metrics, including answer token-level F1 and evidence Recall@5. We additionally report mean $\pm$ standard deviation over 9 independent runs and statistical significance tests in Appendix G.4.

Implementation Details We rerun all baselines in our environment using the released implementations when available, under the same benchmark

Model	Novel Dataset									Medical Dataset								
	Fact Retr.		Cmplx Reas.		Ctx. Summ.		Creative Gen.			Fact Retr.		Cmplx Reas.		Ctx. Summ.		Creative Gen.		
	ACC	R-L	ACC	R-L	ACC	Cov	ACC	FS	Cov	ACC	R-L	ACC	R-L	ACC	Cov	ACC	FS	Cov
RAG	58.95	37.67	41.69	15.22	50.05	82.84	41.37	47.19	37.86	63.26	29.22	58.16	13.95	63.50	78.07	58.48	35.98	57.50
Graph-based RAG																		
MS-GraphRAG	49.25	26.22	51.09	24.24	64.35	75.00	38.80	55.43	35.49	38.72	26.89	46.68	22.10	42.23	22.97	53.32	32.38	39.70
HippoRAG	52.74	26.53	38.50	11.10	48.38	**85.55**	38.99	71.31	38.90	56.32	21.12	55.46	13.53	60.36	62.78	63.85	69.53	65.89
HippoRAG 2	60.25	31.34	53.16	**33.73**	64.35	70.67	48.39	49.51	31.23	66.12	36.46	61.72	36.87	62.96	45.87	67.65	59.09	51.42
LightRAG	58.09	35.27	48.85	24.08	48.87	63.47	23.66	57.35	25.02	63.57	37.44	61.49	25.06	62.87	50.91	67.67	**78.10**	52.16
RAPTOR	49.18	23.82	38.81	11.67	47.13	82.13	37.67	70.79	35.75	54.42	17.86	53.72	11.84	59.02	**78.47**	61.81	60.22	53.56
GFM-RAG	59.23	32.81	45.91	25.57	61.18	71.17	49.13	**71.45**	43.98	63.46	33.83	65.78	25.04	67.52	64.92	65.20	51.09	57.91
LinearRAG	62.17	33.32	56.68	32.91	65.73	66.09	49.12	52.18	47.15	67.21	35.94	64.29	35.23	67.36	47.82	67.05	63.80	65.97
Hypergraph-based RAG																		
HyperGraphRAG	63.13	27.38	57.41	31.02	69.13	61.88	53.29	44.97	52.79	62.73	21.98	65.40	31.57	67.92	75.43	62.14	48.12	46.41
Hyper-RAG	65.29	37.72	56.52	29.18	71.83	69.41	49.92	54.84	**55.71**	61.42	25.25	60.12	25.98	72.53	71.85	67.92	56.88	50.05
HyperSU (Ours)	**67.42**	**42.19**	**61.79**	33.36	**75.04**	82.49	**56.94**	63.93	48.29	**68.49**	**42.06**	**75.03**	**38.74**	**75.01**	76.72	**69.92**	64.19	**67.21**

Table 2: Generation quality on GraphRAG-Bench. All results are obtained from our local reproduction under the unified evaluation protocol.

Model	Novel Dataset								Medical Dataset							
	Fact Retr.		Cmplx Reas.		Ctx. Summ.		Creative Gen.		Fact Retr.		Cmplx Reas.		Ctx. Summ.		Creative Gen.	
	Rec	Rel	Rec	Rel	Rec	Rel	Rec	Rel	Rec	Rel	Rec	Rel	Rec	Rel	Rec	Rel
RAG	61.38	74.78	59.37	80.33	69.59	79.56	32.56	**83.02**	85.48	63.89	84.88	**83.71**	84.00	89.75	44.44	58.77
Graph-based RAG																
MS-GraphRAG	61.11	27.29	72.71	38.70	81.91	43.35	53.40	34.75	38.25	5.67	61.88	4.22	60.17	5.22	67.18	2.75
HippoRAG	80.78	56.30	87.33	59.03	91.40	59.60	65.47	46.87	86.59	52.62	84.25	41.95	83.25	49.17	82.44	44.77
HippoRAG 2	70.94	**80.01**	70.44	**86.02**	83.31	**88.31**	41.96	78.87	79.30	**87.88**	76.70	81.44	77.32	86.43	61.12	**79.18**
LightRAG	74.35	33.34	85.77	37.48	86.93	37.81	71.55	38.09	80.40	41.09	82.44	43.13	85.23	42.87	80.62	45.28
RAPTOR	62.02	54.30	68.39	60.88	75.21	63.22	58.28	58.76	86.15	68.84	89.44	53.50	89.73	59.26	72.91	52.49
GFM-RAG	75.37	47.93	78.77	47.05	87.82	52.91	72.30	47.59	86.37	57.37	83.60	35.42	76.87	40.71	81.92	22.15
LinearRAG	73.75	72.87	81.03	79.82	84.71	61.03	67.15	71.27	85.81	81.74	84.13	78.17	87.92	88.09	87.01	71.02
Hypergraph-based RAG																
HyperGraphRAG	76.57	71.35	82.45	60.03	81.92	62.18	64.32	68.13	72.13	58.33	87.52	72.82	80.09	72.30	69.91	72.19
Hyper-RAG	80.06	73.59	80.96	61.68	85.77	63.27	63.37	69.60	70.80	57.91	80.03	72.11	82.82	74.18	68.94	73.28
HyperSU (Ours)	**83.17**	78.45	**89.30**	67.38	**94.04**	88.19	**78.52**	76.34	**88.90**	77.52	**92.58**	70.14	**93.73**	**90.04**	**93.56**	74.93

Table 3: Retrieval quality on GraphRAG-Bench. All results are obtained from our local reproduction under the unified evaluation protocol.

protocol as HyperSU. We keep the official dataset splits, corpora, task-specific metrics, reader LLM, reader prompt, the number of retrieved passages given to the reader, judge model, and evaluation scripts fixed across methods. Each method retains its official indexing, retrieval, and ranking procedure before the final reader-context selection, so the comparison standardizes answer generation and evaluation while preserving method-specific retrieval designs. Unless otherwise specified, all reader generations use temperature 0. Appendix C provides the full implementation details and hyperparameter settings. To validate our reproduction, we compare our results with the scores originally reported in the corresponding baseline papers or benchmark paper. The reproduced scores generally match those reported in the original papers, and the observed deviations do not change the main rankings or conclusions.

5.2 Performance on GraphRAG-Bench

Tables 2 and 3 report generation and retrieval results on GraphRAG-Bench.

Answer accuracy HyperSU obtains the highest ACC among the compared methods on all eight domain–task settings. The gains are modest on Fact Retrieval, but become larger on reasoning-intensive tasks such as Complex Reasoning, Contextual Summarization, and Creative Generation. These results are consistent with our motivation: source-grounded SU hyperedges preserve local multi-entity evidence, while bidirectional expansion helps compose such evidence across passages. Hypergraph-based baselines are often stronger than standard graph-based methods, suggesting the usefulness of higher-order structure; HyperSU further improves over them by avoiding LLM-generated relational summaries and using clue-guided retrieval.

Retrieval quality HyperSU obtains the highest Context Recall on all eight domain–task settings.

Method	HotpotQA		2Wiki		MuSiQue	
	C.	G.	C.	G.	C.	G.
Graph-based RAG						
KGP	61.0	61.4	31.4	30.2	25.3	30.4
MS-GraphRAG	53.1	53.4	48.1	29.6	21.1	19.6
G-retriever	42.6	40.2	47.0	26.9	14.5	15.3
RAPTOR	56.3	57.8	50.5	41.7	23.5	27.1
E^2GraphRAG	61.5	63.4	54.7	38.4	23.6	26.4
LightRAG	59.8	60.0	54.7	39.3	27.6	28.3
HippoRAG	57.5	58.8	65.5	60.4	29.0	24.3
GFM-RAG	62.2	64.1	67.3	59.1	30.1	34.4
HippoRAG 2	63.4	63.8	62.2	55.4	30.8	35.3
LinearRAG	63.8	67.1	**69.6**	64.2	33.6	37.3
Hypergraph-based RAG						
HyperGraphRAG	63.9	64.9	68.8	64.1	32.7	37.5
Hyper-RAG	65.2	65.6	66.3	69.1	32.1	36.4
HyperSU	**66.0**	**68.1**	69.2	**72.3**	**35.0**	**40.2**

Table 4: Multi-hop QA results. C./G. denote Contain-/GPT-Acc. All results are obtained from our local reproduction under the unified evaluation protocol. Best and second-best results are bolded and underlined.

Context Relevance is more mixed: conservative methods such as dense retrieval or HippoRAG 2 can achieve higher relevance on some tasks. Nevertheless, HyperSU maintains competitive relevance while substantially improving recall, suggesting that forward expansion helps reach distant evidence and backward anchoring with convergence verification helps control noise.

5.3 Results on Multi-hop QA Benchmarks

We further include several RAG frameworks specifically designed for multi-hop question answering and evaluate them on multi-hop QA datasets. Table 4 shows that HyperSU achieves the best GPT-Acc. on all three datasets, improving over the strongest baseline by +1.0 on HotpotQA, +3.2 on 2WikiMultiHopQA, and +2.7 on MuSiQue. The gains are more pronounced on 2WikiMultiHopQA and MuSiQue, where longer reasoning chains require stronger evidence composition. HyperSU is slightly below LinearRAG on 2WikiMultiHopQA under Contain-Acc.; however, HyperSU still achieves higher GPT-Acc., suggesting that exact string matching may underestimate performance on semantically correct answers with different surface forms.

5.4 Component-wise Contribution

We ablate two GraphRAG-Bench variants: **w/o Clue Agent**, replacing multi-clue activation with a single query vector, and **w/o Backward Anchoring**, removing backward expansion and convergence verification (Figure 3).

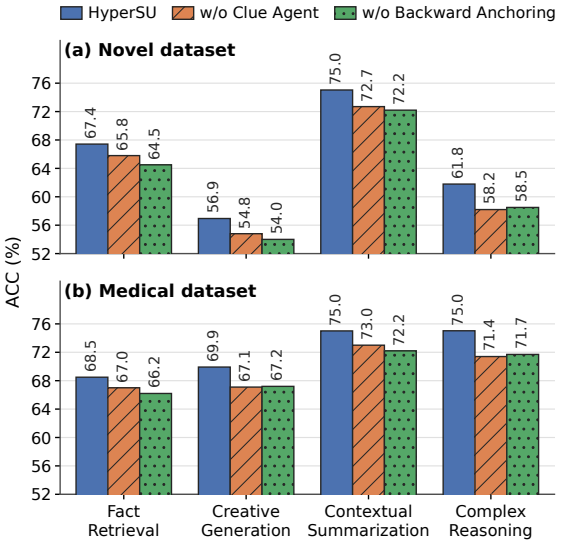

Figure 3: HyperSU component ablation on GraphRAG-Bench; largest drops occur on Complex Reasoning.

Method	Index LLM Tokens	API Cost	Index Time
HyperSU (Ours)	0	$0.00	212 s
HippoRAG 2	1,321K	$0.47	850 s
Hyper-RAG	2,047K	$0.54	1,085 s
LightRAG	2,835K	$0.68	2,400 s
HyperGraphRAG	5,838K	$1.25	4,909 s

Table 5: Offline indexing cost on Medical corpus

Removing the Clue Agent causes the largest drop on Complex Reasoning, where queries often require several pieces of evidence. This is consistent with the role of parallel clues as a noise-tolerant activation mechanism: a well-matched clue can still activate relevant SUs when other clues are noisy. Removing backward anchoring also degrades performance across tasks, suggesting that forward-only expansion may retrieve broader evidence but can accumulate noisy SUs without candidate-evidence anchoring. Appendix E.1 isolates the propagation rule itself by replacing it with PPR on the same SU hypergraph, and reports SU-level precision for the same ablation.

5.5 Computational Efficiency

We profile cost on the Medical corpus ($\sim$1.05 M characters). Tables 5 and 6 report offline indexing and per-query retrieval. A formal time and space complexity analysis is provided in Appendix B.5.

Offline indexing HyperSU uses no generative-LLM calls for entity extraction or hyperedge construction, relying on GLiNER-based entity extraction and MDL-based SU induction. The cost of running GLiNER is included in the indexing time reported in Table 5, so what HyperSU removes is

Method	Tokens	Lat. (s)	Med. Avg
HyperSU (Ours)	406	3.0	**72.11**
Hyper-RAG	521	4.8	65.50
LightRAG	617	5.0	63.90
HippoRAG 2	2,739	9.5	64.61
HyperGraphRAG	3,757	22.1	64.55

Table 6: Per-query retrieval cost on Medical; tokens exclude final generation, and Med. Avg is mean accuracy. For HyperSU, the 406 tokens are the complete Clue Agent usage, 256 prompt and 150 completion tokens on average.

the offline LLM token and API cost, not all indexing computation. As shown in Table 5, HyperSU removes offline LLM indexing tokens and API cost while reducing indexing time. This reduces the cost of indexing new corpora and adapting HyperSU to new domains.

Online retrieval At query time, HyperSU operates over the pre-indexed SU hypergraph. The Clue Agent first produces a compact set of query clues, after which bidirectional expansion and convergence verification are performed without additional LLM-based graph construction. The reported token cost includes all LLM usage during the retrieval stage, excluding final answer generation. The Clue Agent is optional: with it disabled, the original query embedding acts as the single clue and HyperSU performs retrieval with no online LLM call, at the accuracy cost reported in Figure 3. Thus, HyperSU requires no generative-LLM calls for offline indexing while maintaining practical retrieval latency and strong answer accuracy.

6 Conclusion

We presented HyperSU, a corpus-driven semantic-unit hypergraph framework for RAG. HyperSU induces source-grounded SU hyperedges via entity-aware MDL segmentation and retrieves evidence through clue-guided bidirectional expansion, avoiding LLM-generated hyperedges and offline LLM indexing cost. Experiments on GraphRAG-Bench and three multi-hop QA benchmarks demonstrate consistent improvements over standard, graph-based, and hypergraph-based RAG baselines, with larger gains on reasoning-intensive tasks.

Limitations

HyperSU constructs local, source-grounded SU hyperedges rather than pre-materializing predicate-level cross-document n-ary relations. This reduces combinatorial indexing cost and improves reuse, but retrieval still requires the relevant evidence chain to be connected through the induced local SUs. Failures in entity extraction, entity canonicalization, or bridge-entity coverage may therefore prevent the system from recovering otherwise valid reasoning chains.

SU quality depends on the entity-aware dual-channel MDL segmentation. The semantic channel relies on sentence embeddings to capture local discourse coherence, while the entity channel relies on accurate sentence-level mention counts. These assumptions may be less reliable for heterogeneous corpora, highly technical domains, multilingual text, or domains with sparse or ambiguous entity mentions. More adaptive segmentation objectives and domain-aware entity extraction are natural future directions.

Although clue-guided bidirectional expansion balances recall and precision, its per-query overhead still grows with corpus size and hypergraph density, which may become a bottleneck for very large-scale or latency-sensitive deployments. HyperSU also uses several hyperparameters, and the LLM-based Clue Agent introduces an online LLM call when multi-clue planning is enabled; domain transfer may therefore require re-tuning and further robustness analysis.

Evaluation is limited to English-language QA benchmarks. Multilingual settings and other knowledge-intensive tasks remain future work.

Acknowledgments

AI assistants were used only for language polishing of the manuscript and for minor coding assistance. All research ideas, experimental design, results, and analysis are the authors' own, and the authors verified all content.

References

Sameer Agarwal, Kristin Branson, and Serge Belongie. 2006. Higher order learning with graphs. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 17–24.

Arindam Banerjee, Inderjit S. Dhillon, Joydeep Ghosh, and Suvrit Sra. 2005. Clustering on the unit hypersphere using von Mises-Fisher distributions. *Journal of Machine Learning Research*, 6(46):1345–1382.

Circlemind AI. 2024. Fast-GraphRAG.

Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, Dasha Metropolitansky, Robert Osazuwa Ness, and

Jonathan Larson. 2024a. From local to global: A graph RAG approach to query-focused summarization. *CoRR*, abs/2404.16130.

Darren Edge, Ha Trinh, and Jonathan Larson. 2024b. LazyGraphRAG: Setting a new standard for quality and cost.

Yifan Feng, Hao Hu, Shihui Ying, Xingliang Hou, Shiquan Liu, Mingyuan Yang, Junchang Li, Shaoyi Du, Nanning Zheng, Han Hu, and Yue Gao. 2026. Hyper-RAG: Combating LLM hallucinations using hypergraph-driven retrieval-augmented generation. *Nature Communications*.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Meng Wang, and Haofen Wang. 2024. Retrieval-augmented generation for large language models: A survey. *Computing Research Repository*, arXiv:2312.10997.

Peter D. Grünwald. 2007. *The Minimum Description Length Principle*. Adaptive Computation and Machine Learning. MIT Press, Cambridge, MA.

Zirui Guo, Lianghao Xia, Yanhua Yu, Tu Ao, and Chao Huang. 2025. LightRAG: Simple and fast retrieval-augmented generation. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 10746–10761, Suzhou, China. Association for Computational Linguistics.

Bernal Jiménez Gutiérrez, Yiheng Shu, Yu Gu, Michihiro Yasunaga, and Yu Su. 2024. HippoRAG: Neurobiologically inspired long-term memory for large language models. In *Advances in Neural Information Processing Systems*, volume 37.

Bernal Jiménez Gutiérrez, Yiheng Shu, Weijian Qi, Sizhe Zhou, and Yu Su. 2025. From RAG to memory: Non-parametric continual learning for large language models. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 21497–21515. PMLR.

Xiaoxin He, Yijun Tian, Yifei Sun, Nitesh V. Chawla, Thomas Laurent, Yann LeCun, Xavier Bresson, and Bryan Hooi. 2024. G-Retriever: Retrieval-augmented generation for textual graph understanding and question answering. In *Advances in Neural Information Processing Systems*, volume 37.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. 2020. Constructing a multi-hop QA dataset for comprehensive evaluation of reasoning steps. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6609–6625, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems*, volume 33, pages 9459–9474.

Fan Li, Xiaoyang Wang, Dawei Cheng, Ying Zhang, Wenjie Zhang, and Xuemin Lin. 2026. Fairness-aware hypergraph self-supervised learning with sampling-efficient signals. *IEEE Transactions on Knowledge and Data Engineering*.

Jiate Liu, Zebin Chen, Shaobo Qiao, Mingchen Ju, Danting Zhang, Bocheng Han, Shuyue Yu, Xin Shu, Jinglin Wu, Dong Wen, Xin Cao, Guanfeng Liu, and Zhengyi Yang. 2026. A2RAG: Adaptive agentic graph retrieval for cost-aware and reliable reasoning. *CoRR*, abs/2601.21162.

Haoran Luo, Haihong E, Guanting Chen, Yandan Zheng, Xiaobao Wu, Yikai Guo, Qika Lin, Yu Feng, Zemin Kuang, Meina Song, Yifan Zhu, and Anh Tuan Luu. 2025a. HyperGraphRAG: Retrieval-augmented generation via hypergraph-structured knowledge representation. In *Advances in Neural Information Processing Systems*, volume 38, pages 152206–152234. Curran Associates, Inc.

Linhao Luo, Zicheng Zhao, Gholamreza Haffari, Dinh Phung, Chen Gong, and Shirui Pan. 2025b. GFM-RAG: Graph foundation model for retrieval augmented generation. In *Advances in Neural Information Processing Systems*, volume 38.

Qi Luo, Wenjie Zhang, Zhengyi Yang, Dong Wen, Xiaoyang Wang, Dongxiao Yu, and Xuemin Lin. 2024. Hierarchical structure construction on hypergraphs. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*.

Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. The PageRank Citation Ranking: Bringing Order to the Web. Technical Report 1999-66, Stanford InfoLab.

Mengting Pan, Fan Li, Chen Chen, Xiaoyang Wang, and Wenjie Zhang. 2025. HiTeC: Hierarchical contrastive learning on text-attributed hypergraph with semantic-aware augmentation. *CoRR*, abs/2508.03104.

Jorma Rissanen. 1978. Modeling by shortest data description. *Automatica*, 14(5):465–471.

Parth Sarthi, Salman Abdullah, Aditi Tuli, Shubh Khanna, Anna Goldie, and Christopher D. Manning. 2024. RAPTOR: Recursive abstractive processing for tree-organized retrieval. In *The Twelfth International Conference on Learning Representations*.

Gideon Schwarz. 1978. Estimating the dimension of a model. *The Annals of Statistics*, 6(2):461–464.

Liang Sun, Shuiwang Ji, and Jieping Ye. 2008. Hypergraph spectral learning for multi-label classification. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 668–676.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2022. Musique: Multi-hop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554.

Shu Wang, Yixiang Fang, Yingli Zhou, Xilin Liu, and Yuchi Ma. 2026. ArchRAG: Attributed community-based hierarchical retrieval-augmented generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 15868–15876.

Yu Wang, Nedim Lipka, Ryan A. Rossi, Alexa Siu, Ruiyi Zhang, and Tyler Derr. 2024. Knowledge graph prompting for multi-document question answering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 19206–19214.

Zhishang Xiang, Chuanjie Wu, Qinggang Zhang, Shengyuan Chen, Zijin Hong, Xiao Huang, and Jinsong Su. 2026. When to use graphs in RAG: A comprehensive analysis for graph retrieval-augmented generation. In *The Fourteenth International Conference on Learning Representations*.

Zhengyi Yang, Wenke Yang, Guanfeng Liu, and Lu Qin. 2026. RAIDS: Rethinking data systems as responsible intelligent infrastructure. *CoRR*, abs/2606.21831.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

Urchade Zaratiana, Nadi Tomeh, Pierre Holat, and Thierry Charnois. 2024. GLiNER: Generalist model for named entity recognition using bidirectional transformer. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 5364–5376, Mexico City, Mexico. Association for Computational Linguistics.

Yibo Zhao, Jiapeng Zhu, Ye Guo, Kangkang He, and Xiang Li. 2025. E^{2}GraphRAG: Streamlining graph-based RAG for high efficiency and effectiveness. *Computing Research Repository*, arXiv:2505.24226.

Dengyong Zhou, Jiayuan Huang, and Bernhard Schölkopf. 2006. Learning with hypergraphs: Clustering, classification, and embedding. In *Advances in Neural Information Processing Systems*, volume 19, pages 1601–1608.

Luyao Zhuang, Shengyuan Chen, Yilin Xiao, Huachi Zhou, Yujing Zhang, Hao Chen, Qinggang Zhang, and Xiao Huang. 2026. LinearRAG: Linear graph retrieval augmented generation on large-scale corpora. In *Proceedings of the Fourteenth International Conference on Learning Representations*.

A Additional Related Work Comparison

Table 7 provides a compact taxonomy of representative graph- and hypergraph-based RAG methods, comparing their structural representation, edge construction source, and retrieval mechanism. This comparison complements the related-work discussion in the main text and highlights that HyperSU differs from prior systems by inducing source-grounded semantic-unit hyperedges and performing clue-guided bidirectional expansion, rather than relying on LLM-generated triples, summaries, or local hyperedges.

B Theoretical Foundations and Derivations

B.1 Proof of Proposition 1

Proposition 1. *For a source-grounded multi-entity evidence unit e with entity set V_e, representing e as a hyperedge preserves the unit-level membership and provenance of its co-mentioned entities. In contrast, projecting e onto pairwise entity–entity edges may lose the original group identity unless auxiliary event or provenance nodes or labels are added.*

Proof. Let V be a universe of entities, and let an evidence unit be represented by a subset $h \subseteq V$ with $|h| \geq 2$. A collection of evidence units is denoted by $\mathcal{E} \subseteq \mathcal{P}(V)$.

A hypergraph encoding maps $\mathcal{E}$ to $G_H = (V_{\mathcal{E}}, E_H)$, where

$$V_{\mathcal{E}} = \bigcup_{h \in \mathcal{E}} h \qquad \text{and} \qquad E_H = \mathcal{E}.$$

Thus, each evidence unit is retained as a distinct hyperedge, and its group identity is directly recoverable from E_H.

Now consider the standard pairwise projection of $\mathcal{E}$:

$$\pi(\mathcal{E}) = \{\, \{u, v\} \mid u \neq v, \quad \exists h \in \mathcal{E} \text{ s.t. } \{u, v\} \subseteq h \,\}. \tag{4}$$

This projection is not injective. For three distinct entities $a, b, c \in V$, define

$$\mathcal{E}_1 = \big\{\{a, b, c\}\big\},$$
$$\mathcal{E}_2 = \big\{\{a, b\}, \{a, c\}, \{b, c\}\big\}.$$

The first collection contains one three-entity evidence unit, while the second contains three separate pairwise units. However,

$$\pi(\mathcal{E}_1) = \pi(\mathcal{E}_2) = \big\{\{a, b\}, \{a, c\}, \{b, c\}\big\}.$$

Method	Struct.	Edge source	Retrieval
MS-GraphRAG	Binary KG	LLM triples	Community detection + summaries
LightRAG	Binary KG	LLM triples	Keyword-guided 1-hop collection
RAPTOR	Tree	LLM summaries	Tree / collapsed-tree search
HippoRAG	Binary KG	LLM triples	PPR from query entities
HippoRAG 2	Binary KG	LLM triples	PPR with passage integration
GFM-RAG	Binary KG	LLM triples	GNN message passing
LinearRAG	Relation-free graph	Entity extraction	Entity–passage bridging
Fast-GraphRAG	Binary KG	LLM triples	PageRank graph exploration
Lazy-GraphRAG	Binary KG	Deferred LLM summaries	Query-time relevance testing
KGP	Binary KG	LLM triples	LLM-guided graph traversal
G-Retriever	Binary KG	LLM triples	GNN subgraph retrieval
E^2GraphRAG	Hybrid tree + graph	SpaCy entities	Entity-guided chunk lookup
HyperGraphRAG	Hypergraph	LLM local n-ary facts	Entity/hyperedge search
Hyper-RAG	Hypergraph	LLM local correlations	Hypergraph diffusion
HyperSU	Hypergraph	Source-grounded MDL SUs	Clue-guided bidir. expansion + SU ranking

Table 7: Comparison of representative RAG methods.

Therefore, after pairwise projection, one cannot determine whether the three entities appeared in a single shared evidence unit or in three independent pairwise units.

The same argument holds for any evidence unit h with $|h| \geq 3$: the single hyperedge $\{h\}$ and the set of all pairwise edges $\binom{h}{2}$ yield the same pairwise projection but encode different evidence-unit memberships. If each evidence unit is associated with a source span or provenance identifier, this group-level provenance is also not recoverable from unlabeled pairwise edges alone. Hence, pairwise projection loses group identity and provenance. A binary graph can recover this information by introducing auxiliary event, document, or provenance nodes, but such nodes are additional modeling structure beyond the plain pairwise entity projection considered here. □

B.2 Equivalence to the Entity–SU Incidence Bipartite Graph

Proposition 1 contrasts the hyperedge representation with the pairwise entity–entity projection. A different encoding, the entity–SU incidence bipartite graph, preserves the same information as the hypergraph, and we state the correspondence here.

Let $G_H = (V, E_H, H)$ be the SU hypergraph of §4.1. Star expansion (Agarwal et al., 2006; Zhou et al., 2006; Sun et al., 2008) replaces each SU hyperedge $e_u \in E_H$, with entity set V_u, by an auxiliary SU node h_u connected to every $v \in V_u$. The resulting bipartite graph B has biadjacency matrix

$$B[v, h_u] = H[v, e_u] = 1 \quad \text{iff} \quad v \in V_u.$$

As long as the two node types are preserved, the correspondence is bijective: the neighborhood of each SU node recovers its hyperedge, and each hyperedge determines its SU node and incident edges. The encoding is therefore lossless in both directions, so the entity–SU bipartite graph and the SU hypergraph are equivalent representations of the same incidence structure. A provenance-node graph is the same construction with h_u read as a provenance node.

B.3 Derivation of the Entity-Aware Dual-Channel MDL Objective

This section derives the objective in Eq. 3. Let $p = (s_1, \ldots, s_n)$ be a passage with L2-normalized sentence embeddings $X = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$, and let $\mathcal{U}_p = \{u_1, \ldots, u_K\}$ be a contiguous non-overlapping partition. HyperSU scores each candidate SU with a semantic channel and an entity channel under the minimum description length (MDL) principle (Rissanen, 1978; Grünwald, 2007).

Semantic channel. For a segment u, the semantic channel uses the embedding resultant length

$$R_u = \Big\| \sum_{i \in u} \mathbf{x}_i \Big\|.$$

This can be viewed as the sufficient statistic of a von Mises–Fisher-style directional model on normalized sentence embeddings (Banerjee et al., 2005). Up to constants that do not depend on the partition, the semantic reward is κR_u, where κ controls the semantic reward scale.

Entity channel. GLiNER provides sentence-level entity mentions before SU induction. For a candidate segment u, let $c_u(v)$ be the count of normalized entity name v, $N_u = \sum_v c_u(v)$,

$U_u = \{v : c_u(v) > 0\}$, and $\pi_u(v) = c_u(v)/N_u$. If $N_u > 0$, the negative log-likelihood of the entity mention multiset under its segment-level multinomial MLE is

$$N_u \mathcal{H}(\pi_u) = -\sum_{v \in U_u} c_u(v) \log \pi_u(v). \quad (5)$$

A BIC-style code length for the entity distribution contributes (Schwarz, 1978)

$$\frac{|U_u| - 1}{2} \log N_u.$$

When $N_u = 0$, both entity terms are set to zero.

Model complexity and objective. The semantic channel also contributes a segment-level complexity term using an effective semantic dimension d_{eff}:

$$\frac{d_{\text{eff}} - 1}{2} \log n.$$

Combining the semantic reward, entity data code, and complexity terms yields the segment reward

$$r(u) = \kappa R_u - N_u \mathcal{H}(\pi_u) - \frac{|U_u| - 1}{2} \log N_u - \frac{d_{\text{eff}} - 1}{2} \log n, \quad (6)$$

with the entity terms omitted when $N_u = 0$. The full objective is segment-additive:

$$\mathcal{Q}_{\text{MDL}}(\mathcal{U}_p) = \sum_{u \in \mathcal{U}_p} r(u), \quad (7)$$

subject to the word-count constraints

$$w_{\min} \leq \text{words}(u) \leq w_{\max}.$$

The semantic channel favors merging directionally coherent adjacent spans, whereas the entity channel can prevent merges that mix unrelated entity contexts. Thus, the objective produces SUs that are coherent as text spans and compact as entity-bearing hyperedges.

B.4 Proof of Dynamic Programming Optimality

Proposition 2. *For a passage of n sentences with pre-computed candidate-segment rewards, the optimal SU partition maximizing $\mathcal{Q}_{\text{MDL}}$ under contiguity, non-overlap, and word-count constraints can be found exactly by dynamic programming.*

Proof. The objective is segment-additive. For any contiguous non-overlapping partition $\mathcal{U}_p$,

$$\mathcal{Q}_{\text{MDL}}(\mathcal{U}_p) = \sum_{u \in \mathcal{U}_p} r(u),$$

where $r(u)$ depends only on the boundary statistics of segment u. Let $\mathcal{Q}^*(i)$ denote the optimal objective value for prefix $(s_1, \ldots, s_i)$. If the last segment of an optimal partition ends at i and starts at $j+1$, then the preceding prefix $(s_1, \ldots, s_j)$ must itself be optimally partitioned. Otherwise, replacing it with a better prefix partition would improve the full objective.

This yields the Bellman recurrence

$$\mathcal{Q}^*(i) = \max_{j \in \mathcal{J}(i)} \{\mathcal{Q}^*(j) + r(j, i)\}, \quad (8)$$

where $\mathcal{J}(i)$ is the set of feasible previous boundaries under the word-count constraints, and $r(j, i)$ is the dual-channel segment reward for $(s_{j+1}, \ldots, s_i)$. Setting $\text{dp}[0] = 0$ and computing $\text{dp}[i] = \mathcal{Q}^*(i)$ for $i = 1, \ldots, n$ gives the global optimum at $\text{dp}[n]$. Parent pointers recover the corresponding SU partition.

Embedding prefix sums compute R_u from segment boundaries, and sentence-level entity-count statistics compute the entity terms. Since the recurrence enumerates all feasible last-segment choices and combines them with optimal prefix solutions, the recovered partition is globally optimal. □

B.5 Overall Complexity Analysis

Let P be the number of passages, n_p the number of sentences in passage p, $N = \sum_{p=1}^{P} n_p$, $U = |\mathcal{E}_H|$ the number of induced semantic units or hyperedges, $V = |\mathcal{V}|$ the number of canonical entities, and $I = \sum_{u \in \mathcal{E}_H} |V_u|$ the number of entity–SU incidences. Let d be the embedding dimension, M the number of query clues, T_{fwd} and T_{bwd} the forward and backward expansion depths, and K the number of retained entities per hop.

Offline indexing. Excluding neural encoder and GLiNER inference, the main algorithmic cost comes from entity-aware MDL segmentation and hypergraph assembly. For passage p, let C_p be the number of feasible candidate segments satisfying the word-count constraints. Using sentence-embedding prefix sums and sentence-level entity-count statistics, each candidate segment can be scored from boundary statistics, and the one-dimensional dynamic program considers all feasible previous boundaries. Therefore, the total offline

segmentation cost is

$$O\left(\sum_{p=1}^{P} C_p(d+\bar{m}_p)\right),$$

where $\bar{m}_p$ denotes the average number of distinct entity types touched when scoring a candidate segment. In the worst case, $C_p = O(n_p^2)$. In practice, because the word-count constraints bound the maximum segment length and passages are bounded chunks, C_p is near-linear in n_p. Entity canonicalization by normalized exact matching and sparse incidence construction are linear in the number of extracted mentions and incidences, giving $O(I)$ additional time and memory.

Online retrieval. For each query, clue-conditioned SU activation over all SUs costs $O(MUd)$ with exhaustive similarity computation. Bidirectional frontier expansion then visits only the local neighborhoods of retained frontier entities:

$$O\left(\sum_{t=1}^{T_{\text{fwd}}+T_{\text{bwd}}} \sum_{z\in F_t} \sum_{u:z\in V_u} |V_u|\right).$$

With top-K pruning, average incident-SU degree $\bar{h}$, and average hyperedge cardinality $\bar{r}$, this becomes $O((T_{\text{fwd}}+T_{\text{bwd}})K\bar{h}\bar{r})$. Projecting activated SUs back to passages is linear in the number of activated SUs. Thus, with fixed M, T_{fwd}, T_{bwd}, K, and capped-mean parameter L, the online graph traversal scales with the size and density of the activated local neighborhood, while the main corpus-level term is SU activation.

C Implementation and Reproducibility Details

This section reports implementation details that are not expanded in the main text. HyperSU uses GLiNER for offline entity extraction. LLM calls are used for the Clue Agent, answer generation, and GPT-Acc evaluation.

C.1 Experimental Protocol Details

Unified reproduction protocol. All baseline results reported in the main tables are reproduced in our environment. For each benchmark, we keep the official question splits, corpora, task-specific metrics, final reader LLM, reader prompt, final top-k reader-context budget, judge model, and evaluation scripts fixed across methods. Each method retains its own official indexing, retrieval, and ranking procedure before the final reader-context selection, so the comparison standardizes answer generation and evaluation while preserving method-specific retrieval designs. We use released implementations when available and faithful reimplementations otherwise, following official configurations or paper-reported defaults. For all reproduced methods, we pass the top-5 retrieved passages to the final answer generator and do not add an extra reranker unless it is part of the released method itself. Unless otherwise specified, all LLM calls are run with temperature 0. All local indexing, retrieval, reproduction, and efficiency-profiling runs are conducted on the machine configuration reported in Table 8.

Table 8: Machine configuration used in our experiments.

Component	Specification
GPU	NVIDIA RTX PRO 6000 Blackwell Workstation Edition, 96 GB VRAM
CPU	AMD Ryzen 9 9950X3D, 16 cores / 32 threads
Memory	91 GiB RAM
OS	Ubuntu 22.04.4 LTS
CUDA	CUDA 13.1 driver support; CUDA Toolkit 12.4

C.2 Entity Extraction and Canonicalization Details

HyperSU uses `urchade/gliner_large-v2.1` for sentence-level entity extraction with confidence threshold 0.3. Each extracted mention keeps its surface span, entity label, confidence score, sentence ID, and passage ID. Mention names are normalized by lowercasing, Unicode normalization, punctuation cleanup, and low-value mention filtering.

We use two GLiNER label sets, shown in Table 9. The medical label set is used for the Medical domain, and the general label set is used for the Novel domain and multi-hop QA datasets. At query time, extracted query entities are normalized with the same rules as corpus entities and are linked to indexed canonical entities by normalized exact matching. The implementation supports optional high-threshold embedding-based merging for near-identical entity names, but this option is not enabled in the main experiments.

C.3 Hyperparameter Settings

Table 10 summarizes the main HyperSU settings used in the main experiments. Unless otherwise specified, clue-conditioned SU activation uses a fixed global threshold $\delta = 0.5$ and sharpening factor $\gamma = 1.0$ across all datasets. Appendix E further analyzes the sensitivity of retrieval performance to

Table 9: GLiNER label sets used by HyperSU.

Label Set	Labels
Medical	`disease`, `disorder`, `symptom`, `clinical sign`, `drug`, `treatment`, `procedure`, `body part`, `organ`, `diagnostic test`, `lab test`
General	`person`, `character`, `book`, `novel`, `film`, `song`, `work of art`, `location`, `country`, `city`, `organization`, `event`, `date`, `year`, `object`, `product`, `scientific concept`

Table 10: Main hyperparameter settings of HyperSU.

Component	Setting	Value
Entity extraction	GLiNER model	`urchade/gliner_large-v2.1`
Entity extraction	confidence threshold	0.3
Entity canonicalization	matching rule	normalized exact match
Entity canonicalization	embedding merge	disabled in main experiments
Embedding	embedding model	`BAAI/bge-large-en-v1.5`
SU induction	semantic reward scale	$\kappa = 75$
SU induction	effective semantic dimension	$d_{\text{eff}} = 32$
Clue Agent	maximum clues	$M = 3$
SU activation	activation threshold	$\delta = 0.5$
SU activation	sharpening factor	$\gamma = 1.0$
Forward expansion	depth	$T^{\text{fwd}} = 4$
Forward expansion	candidates per hop	30
Forward expansion	hop decay	$\beta = 0.5$
Backward anchoring	depth	$T^{\text{bwd}} = 2$
Backward anchoring	dense seeds	top-10 passages
Convergence verification	convergence bonus	$\mu = 2.0$
Passage projection	capped mean	$L = 3$
Reader context	final passages	top-5 passages
Post-retrieval ranking	reranker	none
LLM inference	temperature	0
Main LLM	Clue Agent and reader	GPT-4o-mini

activation and expansion hyperparameters.

C.4 Prompt A: Clue Agent

Used by the Clue Agent to extract independent reasoning clues from a multi-hop query. Each clue targets a different evidence aspect and is used in parallel during SU activation. In the main experiments, the retriever uses at most $M = 3$ valid clues.

[System] You are a clue extraction agent that identifies the independent evidence aspects needed to answer a multi-hop question. Each clue should describe a distinct piece of evidence that the retrieval system should look for.
Hard Rules: Return valid JSON only. Each clue MUST be a self-contained natural-language description of one aspect of the evidence needed. Ground each clue in entities or concepts explicitly mentioned or directly implied by the query. Prefer the fewest clues that cover all reasoning aspects (typically 2–4). Do NOT produce trivial paraphrases of the original query.

C.5 Prompt B: GPT-Acc Evaluation

Used to compute GPT-Acc. on multi-hop QA benchmarks.

[System] You are an expert evaluator.

[User] Please evaluate whether the generated answer is correct by comparing it with the gold answer.
Generated answer: `{pre_answer}`
Gold answer: `{gold_ans}`
The generated answer should be considered correct if it: (1) contains the key information from the gold answer; (2) is factually accurate and consistent with the gold answer; and (3) does not contain contradicting information. Respond with only "correct" or "incorrect".

Artifact Licenses and Terms of Use. We use third-party artifacts only through their official public releases and follow the corresponding licenses, access conditions, and terms of use. The benchmark datasets used in this paper, including GraphRAG-Bench, HotpotQA, 2WikiMultiHopQA, and MuSiQue, are used solely for research

evaluation. We do not redistribute the raw benchmark data; instead, our release provides code, configuration files, prompts, and preprocessing instructions that allow users to obtain the data from the official sources. We also use publicly released model and software artifacts, including the GLiNER entity extractor, the embedding model, baseline implementations, and evaluation scripts, under their respective licenses. We preserve copyright and license notices for third-party code and require users of our released code to comply with the original licenses of these dependencies. API-based language models used for clue generation, answer generation, or evaluation are accessed under the provider's terms of service, and no proprietary model weights are redistributed. Our own HyperSU implementation is released under MIT. Any derived indices, cached outputs, or processed artifacts are released only when permitted by the licenses or terms of the underlying datasets and models.

C.6 Data Considerations

We do not collect new data. All experiments use public benchmarks as released, and we rely on the filtering their creators report. The multi-hop QA corpora are built from Wikipedia and therefore name public figures; we use this text only as retrieval evidence and do not attempt to re-identify anyone. Because semantic units are verbatim source spans, HyperSU introduces no personal information of its own, but it can surface whatever a corpus already contains, so deployments over sensitive corpora need the same access controls as the corpus itself. We did not run an additional offensive-content audit.

D Analysis of Semantic-Unit Hyperedge Construction

This section analyzes the construction of semantic-unit (SU) hyperedges. We focus on four questions: whether the proposed entity-aware MDL construction outperforms alternative hyperedge construction strategies, whether the induced hyperedges are intrinsically faithful and useful for retrieval, how the MDL segmentation hyperparameters affect downstream performance, and what segmentation behavior the MDL objective produces in practice.

D.1 Controlled Comparison of Hyperedge Construction Strategies

This experiment isolates hyperedge construction from the retrieval algorithm. **Sentence-level hyperedge** uses each sentence as one hyperedge. **Fixed-window hyperedge** partitions a passage into non-overlapping three-sentence windows. **Semantic chunking hyperedge** uses adjacent embedding similarity to form variable-length spans. **MDL w/o entity channel** removes the entity-mention coding term and keeps only semantic cohesion and segmentation complexity. **MDL w/o semantic channel** removes the semantic reward and keeps only the entity channel and segmentation complexity. The two generated baselines follow the local hyperedge generation procedures of HyperGraphRAG and Hyper-RAG, but replace their native retrieval modules with the HyperSU retrieval pipeline.

Table 11 shows that entity-aware MDL achieves the best ACC, Context Recall, and Context Relevance. Sentence-level hyperedges are faithful but too fragmented, while fixed-window hyperedges impose rigid boundaries. Semantic chunking improves over these simple baselines but does not explicitly model entity compactness. The two single-channel MDL ablations further show that both channels are necessary. Removing the entity channel reduces Context Recall, indicating that semantic cohesion alone does not sufficiently control entity diffusion inside hyperedges. Removing the semantic channel preserves relatively high recall but still underperforms full MDL in ACC, suggesting that entity compactness alone does not guarantee coherent, self-contained evidence spans. The full dual-channel objective achieves the best result by jointly modeling sentence-level coherence and entity-mention compactness.

D.2 Intrinsic Evaluation of Hyperedge Representation Quality

We also evaluate the intrinsic quality of the hyperedge sets. For each query–passage instance, GPT-4o-mini is shown the query, the source passage, and the complete hyperedge set produced by one construction strategy. The judge scores the set on faithfulness, information preservation, self-containedness, and retrieval usefulness. We also conduct blind pairwise comparisons between entity-aware MDL and each baseline, with method identities hidden and output order randomized.

Table 12 explains the performance trends in Table 11. Sentence-level hyperedges remain highly faithful because they are directly copied from source sentences, but they are less self-contained and less useful for multi-hop retrieval. Generated hyperedges improve preservation and self-

Table 11: Controlled comparison of hyperedge construction strategies on GraphRAG-Bench, averaged over the eight domain–task settings. All variants use the same downstream entity canonicalization, Clue Agent, bidirectional expansion, SU-to-passage projection, and generation LLM. Only the offline hyperedge construction strategy is changed.

Method	ACC	Context Recall	Context Relevance
Sentence-level hyperedge	63.42	83.07	68.4
Fixed-window hyperedge (3 sent.)	65.18	85.11	71.1
Semantic chunking hyperedge	64.37	86.24	73.9
MDL w/o entity channel	65.29	84.78	69.7
MDL w/o semantic channel	65.36	85.62	70.2
HyperGraphRAG-style generated hyperedges	64.92	85.01	74.5
HyperRAG-style generated hyperedges	61.18	82.36	73.6
Entity-aware MDL (ours)	**68.71**	**89.23**	**77.9**

Table 12: LLM-based evaluation of hyperedge representation quality using GPT-4o-mini. Scores are on a 1–5 scale. "Overall" is the average of faithfulness, information preservation, self-containedness, and retrieval usefulness. "MDL Win Rate" denotes the blind pairwise preference rate of entity-aware MDL against each baseline.

Method	Faith.	Pres.	Self-Cont.	Retr. Use.	Overall	MDL Win Rate
Sentence-level hyperedge	4.61	3.42	3.18	3.27	3.62	71.6%
Fixed-window hyperedge (3 sent.)	4.33	3.89	3.71	3.85	3.94	64.3%
Semantic chunking hyperedge	4.41	4.07	3.98	4.11	4.14	60.2%
MDL w/o entity channel	4.82	4.24	4.09	3.96	4.28	63.2%
MDL w/o semantic channel	4.77	4.19	4.25	4.01	4.31	59.6%
HyperGraphRAG-style generated hyperedges	4.09	4.31	4.37	4.19	4.24	56.9%
HyperRAG-style generated hyperedges	4.05	4.27	4.34	4.15	4.20	58.4%
Entity-aware MDL (ours)	**4.91**	**4.52**	**4.49**	**4.58**	**4.63**	–

containedness, but their faithfulness scores are lower, reflecting the risk of unsupported or over-abstracted generated relations. The two single-channel MDL variants are stronger than simple segmentation baselines, yet both are consistently below full MDL. Entity-aware MDL achieves the best overall score by preserving source faithfulness while forming coherent and retrieval-useful multi-entity evidence units.

D.3 Sensitivity to MDL Segmentation Hyperparameters

The entity-aware dual-channel MDL objective contains two important segmentation hyperparameters: the semantic reward scale κ and the effective semantic dimension d_{eff}. The former controls the strength of the semantic cohesion reward, while the latter controls the segment-level complexity penalty. Since these parameters affect SU granularity and hyperedge structure, we analyze their impact on GraphRAG-Bench performance.

Unless otherwise stated, all settings follow the main experiments. For each configuration, we rebuild the SU hypergraph and run the full retrieval pipeline. We report average ACC and Context Recall over the eight GraphRAG-Bench domain–task settings. We also report the average number of SUs per passage $\bar{k}$ and the average number of sentences per SU $\bar{\ell}$ to characterize segmentation behavior.

Table 13: Sensitivity to the semantic reward scale κ with $d_{\text{eff}} = 32$. ACC and Recall are averaged over the eight GraphRAG-Bench domain–task settings.

κ	$\bar{k}$	$\bar{\ell}$	ACC	Recall
25	1.42	11.75	64.28	81.34
50	2.39	7.12	67.03	86.06
75	**3.12**	**5.44**	**68.71**	**89.23**
100	3.83	4.47	68.26	87.01
150	5.20	3.20	66.17	84.74
200	6.42	2.66	63.43	82.17

Sensitivity to κ. Table 13 shows the effect of varying κ with $d_{\text{eff}} = 32$ fixed. Performance follows an inverted-U trend. When κ is too small, the semantic reward is weak and the dynamic program tends to form coarse SUs, which dilute entity co-mention signals. When κ is too large, the semantic reward dominates and the segmentation becomes too fine, reducing the advantage of higher-order hyperedges. The default value $\kappa = 75$ achieves the best balance.

Sensitivity to d_{eff}. Table 14 varies d_{eff} with $\kappa = 75$ fixed. The role of d_{eff} is not to estimate the

Table 14: Sensitivity to the effective semantic dimension d_{eff} with $\kappa = 75$. ACC and Recall are averaged over the eight GraphRAG-Bench domain–task settings.

d_{eff}	$\bar{k}$	$\bar{\ell}$	ACC	Recall
8	5.67	3.04	65.73	84.22
16	4.25	4.02	67.86	86.86
32	**3.12**	**5.44**	**68.71**	**89.23**
64	2.55	6.73	68.00	87.18
128	1.91	8.84	66.48	85.47
256	1.39	12.93	63.05	81.02

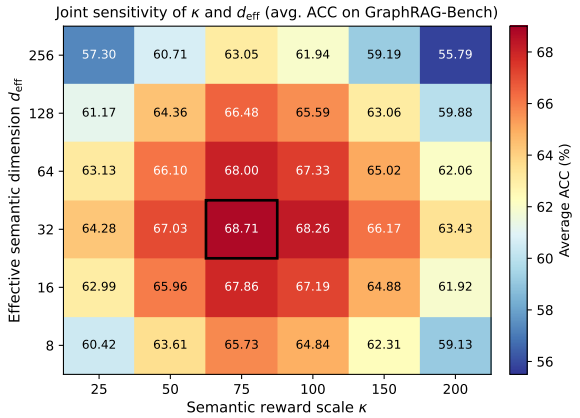

Figure 4: Joint sensitivity of κ and d_{eff} on GraphRAG-Bench. Each cell reports average ACC over the eight domain–task settings. The central region is stable, while extreme settings lead to overly coarse or overly fine SU segmentation.

full embedding dimension. Instead, it acts as an effective capacity parameter in the MDL complexity term. From the MDL perspective, which operationalizes Occam's principle, a good segmentation should explain local semantic coherence without paying unnecessary segmentation complexity. When d_{eff} is too small, the complexity penalty is weak and the dynamic program tends to oversegment passages. When d_{eff} is too large, the penalty becomes too strong and the model favors overly coarse segments. The stable region around $d_{\text{eff}} \in [16, 64]$ suggests that HyperSU only requires a moderate complexity-control scale.

Joint sensitivity. Figure 4 visualizes the joint $\kappa \times d_{\text{eff}}$ sweep on GraphRAG-Bench. The best performance occurs around the central parameter settings rather than at the extremes. Configurations around $\kappa \in [50, 100]$ and $d_{\text{eff}} \in [16, 64]$ remain close to the best score, indicating that the MDL objective remains robust over a reasonably broad range of parameter values. In contrast, extreme combinations lead to degenerate segmentations: overly coarse SUs behave like passage-level chunks, while overly fine SUs lose the multi-entity binding advantage of hyperedges.

Overall, the sensitivity analysis supports two conclusions. First, the default setting $(\kappa, d_{\text{eff}}) = (75, 32)$ achieves the best or near-best performance. Second, HyperSU is not highly sensitive to moderate changes in these hyperparameters: a broad range of values still yields strong performance, while the performance drops at extreme settings can be explained by clear structural failure modes in the induced SU segmentation.

D.4 Hyperedge Statistics and Segmentation Behavior

Table 16 summarizes the segmentation behavior of the MDL objective. On HotpotQA and 2WikiMultiHopQA, the induced SUs contain about three sentences on average, indicating that the model does not degenerate into sentence-level hyperedges. On MuSiQue, the average SU is longer, which is consistent with the longer and more compositional reasoning chains in this dataset. Across datasets, MDL segmentation produces local evidence units rather than passage-level hyperedges, supporting HyperSU's design goal of constructing reusable, source-grounded hyperedges from compact contiguous spans.

D.5 Examples of MDL-Induced Semantic Units

Table 15 shows two semantic units induced from 2WikiMultiHopQA. These examples illustrate that the MDL objective can keep a coherent event, object, or biographical fact bundle as a single source-grounded hyperedge while separating it from neighboring topics.

The first example forms a biographical and career-oriented evidence unit: the person, origin, current location, bands, predecessor band, and bandmate are all part of the same local context. The second example forms a production-centered evidence unit: the film, source work, adapter, director, actors, and roles are tied to the same film record. Both cases would be awkward to represent only as independent pairwise edges, because the original group membership and source context are important for retrieval.

Table 15: Examples of MDL-induced semantic units from 2WikiMultiHopQA. Each SU is a contiguous source span and induces a hyperedge over its canonical entity vertices.

Type	Semantic Unit Text	Hyperedge Vertices
Musician and band affiliations	Billy Milano is a Bronx-born heavy metal musician now based in Austin, Texas. He is the singer and occasionally guitarist and bassist of crossover thrash band M.O.D., and he was also the singer of its predecessor, Stormtroopers of Death. He was also the singer of United Forces, which also featured his Stormtroopers of Death bandmate Dan Lilker.	Billy Milano; Bronx; Austin; Texas; M.O.D.; Stormtroopers of Death; United Forces; Dan Lilker
Film adaptation and production	The Private Life of Helen of Troy is a 1927 American silent film about Helen of Troy based on the 1925 novel of the same name by John Erskine, and adapted to screen by Gerald Duffy. The film was directed by Alexander Korda and starred Maria Corda as Helen, Lewis Stone as Menelaus, and Ricardo Cortez as Paris.	The Private Life of Helen of Troy; Helen of Troy; John Erskine; Gerald Duffy; Alexander Korda; Maria Corda; Helen; Lewis Stone; Menelaus; Ricardo Cortez; Paris

Table 16: Statistics of MDL-induced semantic-unit hyperedges on multi-hop QA datasets. #SUs counts unique semantic-unit hyperedges.

Dataset	#Sent.	#SUs	Avg. Sent./SU
HotpotQA	58,029	18,412	3.15
2WikiMultiHopQA	27,866	8,961	3.11
MuSiQue	70,446	12,392	5.68

E Analysis of Clue-Guided Bidirectional Retrieval

This section provides a more detailed analysis of the online retrieval stage of HyperSU. While the main text shows that clue-guided bidirectional retrieval improves end-to-end performance, here we first isolate the propagation rule itself and measure SU-level retrieval precision, and then examine how its forward-exploration and backward-verification hyperparameters affect multi-hop QA performance. The results reveal that the two directions play different roles: forward exploration is more sensitive to the reasoning depth of the dataset, whereas backward verification is most effective when kept shallow and selective.

E.1 Retrieval-Mechanism Ablation

§4.2 defines retrieval as clue-gated bidirectional expansion over the semantic-unit hypergraph. This subsection isolates that choice. All variants below share the same corpus, the same entity-aware MDL semantic units, the same hypergraph, and the same SU-to-passage projection; they differ only in how activation propagates over that structure. Two variants replace our propagation with personalized PageRank (PPR), seeded either at query-linked entities or at query-linked entities together with the entities of the top-10 dense-retrieved passages, so that PPR receives the same dense-retrieval signal that candidate-evidence anchoring consumes. A third variant keeps clue-gated forward expansion but removes backward anchoring and convergence verification.

Metrics. Besides evidence Recall@5 and GPT-Acc., we report MAP@5, which measures how highly gold passages are ranked inside the reader's top-5 context budget. Let p_k be the passage ranked at position k and G the set of gold supporting passages. Then

$$\mathrm{AP@5} = \frac{1}{|G|} \sum_{k=1}^{5} \mathrm{P@}k \cdot \mathbf{1}[p_k \in G],$$

where $\mathrm{P@}k$ is the precision among the top-k retrieved passages, and MAP@5 is the macro-average of $\mathrm{AP@5}$ over questions.

Table 17 shows where PPR loses. It still reaches most gold evidence—on HotpotQA its Recall@5 trails the full model by 7.1 points—but it fails to rank that evidence into the top-5 reader budget, where MAP@5 trails by 26.5 points. The cause is that the PPR transition kernel is purely structural: stationary mass grows with entity degree, so high-degree entities attract activation regardless of query relevance. Measured directly on HotpotQA, with hubs defined as the top-1% highest-degree entities and seed entities excluded, PPR places 37.6% of its propagated entity-side mass on hub entities, against 22.9% under clue-gated activation, and 53.2% of its top-5 passages are hub-containing non-gold passages, against 27.3% for HyperSU. Among the non-gold passages each method retrieves, 82% of PPR's are hub-containing versus 44% of ours. The second condition gives PPR the same dense-retrieval infor-

Table 17: Retrieval-mechanism ablation on the same SU hypergraph. All variants share the same corpus, entity-aware MDL semantic units, hypergraph, and SU-to-passage projection, and differ only in how activation propagates over that structure.

Propagation	HotpotQA			2WikiMultiHopQA			MuSiQue		
	R@5	MAP@5	GPT-Acc	R@5	MAP@5	GPT-Acc	R@5	MAP@5	GPT-Acc
PPR from query entities	88.4	61.3	65.1	83.6	66.8	64.2	67.4	49.5	36.2
PPR from query ∪ dense-top-10 entities	89.1	62.2	65.4	82.9	65.7	64.9	68.6	48.4	36.5
Clue-gated expansion, no backward anchoring	91.2	84.3	68.0	85.7	68.3	69.2	71.6	52.8	38.9
HyperSU (full)	**95.5**	**87.8**	**68.1**	**89.1**	**75.2**	**72.3**	**75.3**	**59.5**	**40.2**

Table 18: SU-level retrieval precision with and without backward anchoring. An SU counts as relevant if it contains a gold supporting sentence; SU-Prec@K is the proportion of relevant SUs among the top-K ranked SUs, and the noisy-SU ratio at K is $100-\text{SU-Prec@}K$.

Variant	SU-Prec@3	SU-Prec@10	Noisy-SU@3
HotpotQA			
Full	**77.2**	**37.4**	**22.8**
w/o bwd	72.5	34.8	27.5
2WikiMultiHopQA			
Full	**65.9**	**31.6**	**34.1**
w/o bwd	58.0	25.5	42.0
MuSiQue			
Full	**51.8**	**30.7**	**48.2**
w/o bwd	46.1	30.1	53.9

mation that backward anchoring consumes, so the remaining gap is not explained by that extra signal.

Removing backward anchoring degrades ranking more than coverage. On HotpotQA, GPT-Acc. is essentially unchanged, from 68.1 to 68.0, while MAP@5 drops 3.5 points; on 2WikiMultiHopQA and MuSiQue both degrade. This is consistent with the role of backward anchoring as a precision-oriented refinement rather than an additional source of recall.

Table 18 evaluates the same ablation at the SU level. Removing backward anchoring reduces SU-Prec@3 by 4.7, 7.9, and 5.7 points on HotpotQA, 2WikiMultiHopQA, and MuSiQue, and the noisy-SU ratio at 3 rises by the same margins. The drops are smaller at SU-Prec@10 than at SU-Prec@3, indicating that backward anchoring and convergence-based reweighting mainly improve the highest-ranked SUs, which are the ones that dominate the capped mean used for passage scoring.

Bridge-evidence correctness. Whether an SU actually bridges two reasoning hops cannot be read off the existing gold annotations, which mark supporting evidence but not its bridging role. We therefore evaluate it manually. For each dataset we sample 100 bridge or compositional questions and inspect every doubly confirmed SU in $\mathcal{U}_{\text{meet}}$, counting an SU as correct if it comes from a gold supporting passage and contains an intermediate entity or fact linking the reasoning hops. Two of the authors independently annotated the sampled SUs; disagreements were resolved by discussion. Correctness is 69.7% on HotpotQA, 77.5% on 2WikiMultiHopQA, and 68.3% on MuSiQue, so roughly seven in ten SUs promoted by convergence verification carry valid bridge evidence.

Table 19: Context Relevance on GraphRAG-Bench with and without backward anchoring.

Domain–Task	Full	w/o Backward
Novel Fact Retrieval	78.45	74.21
Novel Complex Reasoning	67.38	61.92
Novel Contextual Summ.	88.19	82.16
Novel Creative Generation	76.34	71.89
Medical Fact Retrieval	77.52	74.13
Medical Complex Reasoning	70.14	67.65
Medical Contextual Summ.	90.04	85.17
Medical Creative Generation	74.93	73.58
Average	**77.87**	73.84

On GraphRAG-Bench the corresponding precision-oriented metric is Context Relevance. Table 19 shows that removing backward anchoring lowers it in all eight domain–task settings, from 77.87 to 73.84 on average. Together with the SU-level precision and manual bridge-evidence results, this gives direct evidence that backward anchoring and convergence-based reweighting improve the quality of top-ranked evidence rather than only its quantity.

E.2 Sensitivity to Forward-Exploration Hyperparameters

Forward exploration starts from query-linked entities and expands through clue-activated SUs to discover bridge evidence. We analyze two hyperparameters that control this process: the SU activation threshold δ and the forward expansion depth T^{fwd}.

Figure 5(a) shows that the effect of δ depends on

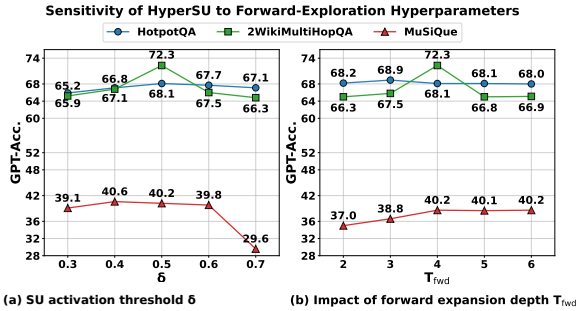

Figure 5: Sensitivity of HyperSU to forward-exploration hyperparameters on HotpotQA, 2WikiMultiHopQA, and MuSiQue. Each point reports GPT-Acc. on the corresponding dataset. (a) Impact of the activation threshold δ. (b) Impact of the forward expansion depth T^{fwd}.

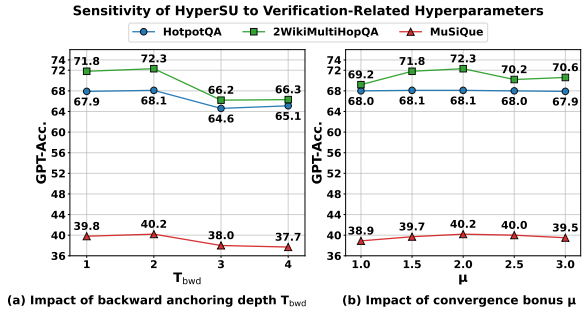

Figure 6: Sensitivity of HyperSU to verification-related hyperparameters on HotpotQA, 2WikiMultiHopQA, and MuSiQue. Each point reports GPT-Acc. on the corresponding dataset. (a) Impact of the backward anchoring depth T^{bwd}. (b) Impact of the convergence bonus μ.

the difficulty of the dataset. On HotpotQA, performance is relatively stable across different thresholds, ranging from 65.2 to 68.1 GPT-Acc. This suggests that for shorter reasoning chains, HyperSU is not overly sensitive to the exact activation threshold as long as extremely noisy SUs are not widely activated. 2WikiMultiHopQA shows a sharper optimum at $\delta = 0.5$, where GPT-Acc. reaches 72.3. This indicates that 2WikiMultiHopQA benefits from a balanced threshold: a lower threshold admits more weakly related SUs, while a higher threshold may remove useful bridge SUs needed for cross-entity reasoning.

The trend is more pronounced on MuSiQue. Although moderate thresholds produce similar performance, setting δ too high causes a large drop, with GPT-Acc. falling to 29.6 at $\delta = 0.7$. MuSiQue contains longer and more difficult reasoning chains, where some necessary intermediate SUs may be only indirectly related to the surface form of the query. A strict activation threshold can therefore filter out semantically indirect but structurally necessary bridge evidence before expansion can connect it to answer-bearing passages.

Figure 5(b) further shows that the preferred forward depth also varies with reasoning complexity. On HotpotQA, performance remains nearly flat as T^{fwd} increases, suggesting that shallow or moderate expansion is already sufficient for most questions. On 2WikiMultiHopQA, the best performance is obtained at $T^{\text{fwd}} = 4$, while deeper expansion degrades performance. This suggests that moderate expansion helps reach bridge evidence, but excessive hops can propagate into irrelevant hyperedges and introduce answer-irrelevant SUs.

MuSiQue exhibits a different pattern. Its performance is lower when T^{fwd} is too small, but improves once the depth reaches 4 and then remains stable. This indicates that harder multi-hop questions require deeper forward exploration: with too few expansion steps, HyperSU may fail to collect a complete evidence chain. Overall, these results show that forward exploration should be sufficiently deep for long-hop reasoning, but bounded to avoid uncontrolled diffusion into noisy hyperedges.

E.3 Sensitivity to Verification-Related Hyperparameters

Backward anchoring starts from entities in top dense-retrieved passages and verifies forward-reached evidence against those candidates. We analyze two hyperparameters that control this verification process: the backward anchoring depth T^{bwd} and the convergence bonus μ.

Figure 6(a) shows a consistent trend across all three datasets: backward verification works best with a small depth. In particular, $T^{\text{bwd}} = 2$ gives the best performance on HotpotQA, 2WikiMultiHopQA, and MuSiQue. Increasing the depth beyond 2 consistently hurts performance. For example, on 2WikiMultiHopQA, GPT-Acc. drops from 72.3 at $T^{\text{bwd}} = 2$ to 66.2 at $T^{\text{bwd}} = 3$; on MuSiQue, it decreases from 40.2 to 38.0. This suggests that candidate-evidence anchoring should remain local and selective. When the backward frontier expands too far from dense-retrieved passages, it is more likely to include distractor entities and topically related but answer-irrelevant SUs, weakening its ability to verify useful forward evidence.

Figure 6(b) shows that the convergence bonus

is most effective at a moderate value. Across the three datasets, $\mu = 2.0$ achieves the best or tied-best performance. The gain is especially visible on 2WikiMultiHopQA, where GPT-Acc. increases from 69.2 at $\mu = 1.0$ to 72.3 at $\mu = 2.0$, and on MuSiQue, where it improves from 38.9 to 40.2. This confirms that bidirectional convergence is a useful verification signal: an SU reached from both the query-side expansion and the candidate-evidence anchoring frontier is more likely to serve as reliable bridge evidence.

However, larger convergence bonuses do not further improve performance. When μ is too large, doubly confirmed SUs can dominate the final passage ranking, even though convergence alone does not guarantee that an SU is answer-critical. Some converged SUs may correspond to hub-like entities or broadly relevant background passages, while some necessary bridge SUs may only be reached from the forward direction within the limited backward depth. Therefore, μ should be treated as a soft preference for bidirectionally verified evidence rather than a hard preference that suppresses forward-only bridge evidence.

Taken together, these results show that forward exploration is the main source of retrieval flexibility in HyperSU. Longer-hop datasets benefit from sufficiently deep expansion and can be more sensitive to overly strict activation thresholds. Nevertheless, the fixed global setting used in the main experiments, $\delta = 0.5$ and $T^{\text{fwd}} = 4$, provides a balanced operating point across the evaluated multi-hop QA datasets.

F Qualitative Analysis of Retrieval Behavior

This section provides a qualitative analysis of how HyperSU retrieves and ranks multi-hop evidence. We focus on a representative 2WikiMultiHopQA example where the question requires connecting a film page to the biography page of its director.

Table 20 compares HyperSU with representative RAG baselines. The baselines either retrieve only partial film-side evidence or stay within a topical neighborhood of films and directors, but fail to surface the answer-bearing biography chunk. In contrast, HyperSU ranks both supporting chunks in the final context and produces the correct answer.

Table 21 further shows the SU-level activation trace for the same instance. The film-title/director SU and the director-biography SU are both reached by forward and backward activation, so their projected chunks are promoted in the final ranking. This example illustrates that HyperSU does not feed isolated SUs directly to the reader LLM. Instead, clue-activated and convergence-confirmed SUs are projected back to passages, allowing the final context to preserve complete chunk-level evidence while still benefiting from fine-grained SU-level retrieval.

G Robustness and Evaluation Reliability

This appendix complements the main evaluation with additional checks on metric choice, model choice, and evaluator choice. The main text reports end-to-end performance using benchmark-specific LLM-based evaluation protocols. While such protocols are standard for open-ended generation tasks, they may introduce sensitivity to evaluator preferences or surface-form variation. We therefore conduct four additional analyses. First, we evaluate multi-hop QA with deterministic answer and evidence metrics. Second, we test whether the conclusions depend on the embedding model or the generation backbone. Third, we fix generated answers and vary only the judge LLM to examine evaluator stability. Fourth, we repeat the comparison against hypergraph-based baselines over multiple runs and test whether the improvements are statistically significant. These analyses are intended to assess whether HyperSU's gains persist under alternative, controlled evaluation settings rather than under a single model or scoring configuration.

G.1 Non-LLM Evaluation on Multi-hop QA

To reduce dependence on LLM-as-judge evaluation, we further evaluate multi-hop QA using deterministic metrics. In addition to Contain-Acc. and GPT-Acc. in the main text, we report answer token-level F1 and passage-level evidence Recall@5. Answer F1 is computed with standard normalization, including lowercasing and punctuation/article removal. Evidence Recall@5 measures whether gold supporting passages appear in the top-5 retrieved passages. These two metrics capture complementary aspects of the task: answer F1 measures whether the final response matches the gold answer at the lexical level, while Recall@5 directly evaluates whether the retriever surfaces the evidence required for multi-hop reasoning.

Table 22 shows that HyperSU achieves the highest Recall@5 on all three datasets, indicating that

Table 20: Qualitative case study with top-5 retrieved chunks on a real 2WikiMultiHopQA question. Bold chunk IDs indicate gold supporting chunks: Chunk 577 provides the film–director bridge, and Chunk 578 contains the answer-bearing death-place evidence.

Field / Method	**Evidence in Retrieved Context**	**Outcome**	**Diagnosis**
Question ID	352		
Question	*Where did the director of film Dancing In The Rain (Film) die?*		
Ground Truth	Ljubljana		
Support Context	√ Chunk 577: *Dancing in the Rain* was directed by Boštjan Hladnik. √ Chunk 578: Hladnik died in Ljubljana in 2006.		
HippoRAG 2	Top-5 contains the film chunk and several topical distractors, but misses the Hladnik biography. *Top-5 retrieved chunks:* [1] **577**: Ples v dežju is a 1961 Slovene film directed by Boštjan Hladnik. Its international English title is "Dance in the..." [2] 579: Dancing in the Rain may refer to: [3] 5489: Roman Polański born 18 August 1933 in Paris; original name Raymond Thierry Liebling is a French-Polish film director, producer... [4] 4931: Barry Levinson born April 6, 1942 is an American filmmaker, screenwriter, and actor... [5] 950: Bernardo Bertolucci 16 March 1941–26 November 2018 was an Italian director and screenwriter...	× Not specified	Finds the film page but diffuses through the broad film-director neighborhood.
HyperGraphRAG	Local generated hyperedges can capture the film–director association, but the death-place evidence is in a separate biography chunk. *Top-5 retrieved chunks:* [1] **577**: Ples v dežju is a 1961 Slovene film directed by Boštjan Hladnik. Its international English title is "Dance in the..." [2] 579: Dancing in the Rain may refer to: [3] 950: Bernardo Bertolucci 16 March 1941–26 November 2018 was an Italian director and screenwriter... [4] 4050: Ernst Ingmar Bergman 14 July 1918–30 July 2007 was a Swedish director, writer, and producer... [5] 4931: Barry Levinson born April 6, 1942 is an American filmmaker, screenwriter, and actor...	× No answer evidence	Local hyperedges preserve the bridge name but do not reliably compose to the answer-bearing biography.
Hyper-RAG	Diffusion keeps evidence around films and directors, but does not surface Chunk 578. *Top-5 retrieved chunks:* [1] **577**: Ples v dežju is a 1961 Slovene film directed by Boštjan Hladnik. Its international English title is "Dance in the..." [2] 5489: Roman Polański born 18 August 1933 in Paris; original name Raymond Thierry Liebling is a French-Polish film director, producer... [3] 4931: Barry Levinson born April 6, 1942 is an American filmmaker, screenwriter, and actor... [4] 950: Bernardo Bertolucci 16 March 1941–26 November 2018 was an Italian director and screenwriter... [5] 4050: Ernst Ingmar Bergman 14 July 1918–30 July 2007 was a Swedish director, writer, and producer...	× No answer evidence	The signal remains in a topical director region instead of converging on the specific Hladnik page.
HyperSU	Ranks both supporting chunks in the top context: Chunk 577 identifies Boštjan Hladnik as the director, and Chunk 578 states that Hladnik died in Ljubljana. *Top-5 retrieved chunks:* [1] **577**: Ples v dežju is a 1961 Slovene film directed by Boštjan Hladnik. Its international English title is "Dance in the..." [2] **578**: Boštjan Hladnik 30 January 1929–30 May 2006 was a Yugoslavian/Slovene filmmaker. Hladnik died in Ljubljana... [3] 579: Dancing in the Rain may refer to: [4] 4931: Barry Levinson born April 6, 1942 is an American filmmaker, screenwriter, and actor... [5] 950: Bernardo Bertolucci 16 March 1941–26 November 2018 was an Italian director and screenwriter...	√ Ljubljana	Forward and backward activation meet on the bridge entity and promote both supporting chunks.

the proposed SU hypergraph retrieval more consistently brings gold supporting evidence into the top retrieved context. This pattern is particularly important for multi-hop QA, where answer generation often fails when any bridge evidence is missing. At the answer level, HyperSU obtains the best F1 on 2WikiMultiHopQA and MuSiQue, and remains within 0.3 points of the best method on HotpotQA. The HotpotQA result suggests that HyperSU does not uniformly maximize lexical overlap under every dataset, but its consistently higher evidence recall provides stronger support for the retrieval-side claim of the paper. Overall, the deterministic metrics are consistent with the main GPT-based re-

Table 21: Qualitative HyperSU activation trace for the retrieval instance in Table 20.

Activated SU / Evidence	Matched clue	Expansion	Projection	Role
Chunk 577: film title and director Boštjan Hladnik	film, director	forward + backward	Chunk 577	bridge evidence
Chunk 578: Hladnik biography with "died in Ljubljana"	director, death place	forward + backward	Chunk 578	answer evidence
Chunk 577: novel, Paris apprenticeship, DVD materials	partial film context	forward only	lower rank	local distractor
Disambiguation and generic film-director biographies	topical overlap	no meeting path	suppressed	distractor

Table 22: Non-LLM evaluation on multi-hop QA benchmarks. We report answer token-level F1 and passage-level evidence Recall@5 (R@5) against gold supporting passages. Best results are in bold and second-best results are underlined.

Method	HotpotQA		2WikiMultiHopQA		MuSiQue	
	F1	R@5	F1	R@5	F1	R@5
RAG	68.7	89.4	58.4	72.4	43.5	64.4
HippoRAG 2	72.5	92.1	67.3	85.2	47.2	70.2
LightRAG	71.8	87.7	63.8	79.5	45.7	65.1
LinearRAG	69.5	90.6	65.9	84.8	44.2	69.6
HyperGraphRAG	**73.2**	92.3	65.1	<u>87.3</u>	<u>48.3</u>	71.7
Hyper-RAG	70.6	<u>93.8</u>	<u>69.2</u>	84.8	46.6	<u>73.5</u>
HyperSU w/o Clue Agent	70.3	93.2	67.4	86.1	46.7	73.0
HyperSU	<u>72.9</u>	**95.5**	**69.7**	**89.1**	**49.0**	**75.3**

sults: HyperSU's advantage is not solely an artifact of LLM-as-judge evaluation.

The table also reports two variants relevant to online cost. LinearRAG performs retrieval without any online LLM call. HyperSU can operate in the same regime: disabling the Clue Agent makes the original query embedding the single clue, and the resulting variant still exceeds LinearRAG on all six dataset–metric combinations, by 0.8 to 2.5 F1 and 1.3 to 3.4 Recall@5 points, while the full model with one Clue Agent call is stronger again. HyperSU can therefore be run in either a zero-online-LLM or a multi-clue mode.

G.2 Robustness to Retrieval and Generation Models

We next examine whether HyperSU depends on a particular embedding model or LLM backbone. This is important because HyperSU uses embeddings for semantic matching among queries, clues, SUs, entities, and passages, while the online Clue Agent and reader depend on the selected LLM backbone. For these experiments, the evaluation setting is kept fixed within each comparison, and each cell reports average ACC over the eight GraphRAG-Bench domain–task settings. For HyperSU, GLiNER-based indexing remains fixed; thus, changes in the LLM backbone affect the Clue Agent and reader LLM, but not entity extraction or hyperedge construction.

Table 23 shows that HyperSU remains the strongest method under all four embedding models. Its average ACC ranges from 67.9 to 69.0, with an overall average of 68.5, whereas the strongest baseline achieves an average ACC of 63.4. The relative ranking is therefore stable even when the embedding model is changed. This suggests that the benefit of HyperSU is not tied to a single embedding model, but instead comes from the combination of source-grounded SU hyperedges and clue-guided bidirectional expansion. In particular, although the embedding model affects semantic matching quality for all retrieval methods, HyperSU consistently maintains a substantial margin over both graph-based and hypergraph-based baselines.

Table 24 further varies the LLM backbone used for generation. HyperSU obtains the best average ACC under GPT-4o-mini, Gemini 2.5 Flash, and Claude Sonnet 4.6, with an overall average of 70.9. The gains are consistent across backbones, rather than appearing only with one specific reader or planner. This result is important because HyperSU uses an LLM-based Clue Agent at query time; if the gains were mainly due to a particular planner or reader, one would expect the relative advantage

Table 23: Embedding model robustness on GraphRAG-Bench. Each cell reports average ACC over the eight GraphRAG-Bench domain–task settings.

Method	bge-large-en	e5-large-v2	jina-v3	text-emb-3-large	Avg.
RAG	54.4	54.9	55.7	56.5	55.4
HippoRAG 2	60.6	61.2	60.3	62.1	61.1
LightRAG	54.4	54.4	54.8	55.0	54.7
HyperGraphRAG	62.6	62.5	63.2	63.9	63.1
Hyper-RAG	63.2	63.7	62.9	63.9	63.4
HyperSU	**68.7**	**67.9**	**68.2**	**69.0**	**68.5**

Table 24: LLM backbone robustness on GraphRAG-Bench. For HyperSU, GLiNER-based indexing is fixed; the LLM backbone affects the Clue Agent and reader LLM.

Method	GPT-4o-mini	Gemini 2.5 Flash	Claude Sonnet 4.6	Avg.
RAG	54.4	58.7	59.9	57.7
HippoRAG 2	60.6	65.4	64.0	63.3
LightRAG	54.4	58.3	59.1	57.3
HyperGraphRAG	62.6	67.9	65.8	65.4
Hyper-RAG	63.2	68.2	67.5	66.3
HyperSU	**68.7**	**72.3**	**71.7**	**70.9**

Table 25: Judge LLM robustness on GraphRAG-Bench. Generated answers are fixed, and only the judge LLM is changed.

Method	GPT-4o-mini	Gemini 2.5 Pro	Claude Sonnet 4.6	Avg.
RAG	54.4	55.8	56.2	55.5
HippoRAG 2	60.6	61.1	60.9	60.9
LightRAG	54.4	55.0	54.7	54.7
HyperGraphRAG	62.6	63.6	62.3	62.8
Hyper-RAG	63.2	63.4	63.4	63.3
HyperSU	**68.7**	**69.9**	**70.6**	**69.7**

to be unstable when replacing the LLM backbone. Instead, the observed stability suggests that the retrieved SU evidence provides useful context across different generation models.

G.3 Robustness to Judge LLMs

Finally, we examine whether the GraphRAG-Bench conclusions are sensitive to the choice of judge LLM. Unlike the previous experiment, generated answers are fixed in this setting, and only the evaluator is changed. This isolates the effect of judge preference from the effect of retrieval or generation quality. A method whose advantage depends on one judge may indicate evaluator-specific bias; a method that remains consistently strong across judges provides more reliable evidence for the reported ranking.

As shown in Table 25, HyperSU remains the top-performing method under all three judge LLMs. Its average ACC is 69.7, compared with 63.3 for the strongest baseline average. The absolute scores vary moderately across judges, which is expected for LLM-based evaluation, but the method ranking remains stable. This reduces the concern that the main GraphRAG-Bench conclusion is driven by a single evaluator's preference. Together with the deterministic multi-hop QA results in Table 22, these findings provide complementary evidence that HyperSU's improvements are robust under different evaluation protocols, embedding models, generation backbones, and judge LLMs.

G.4 Statistical Significance Test

We further test whether HyperSU's improvements over hypergraph-based RAG baselines are statistically significant. We run HyperSU, HyperGraphRAG, and Hyper-RAG for 9 independent runs under the same benchmark protocols, and report mean $\pm$ standard deviation. For each setting, statistical testing compares HyperSU with the stronger hypergraph baseline between HyperGraphRAG and Hyper-RAG.

The runs differ in scope between HyperSU and the baselines. HyperSU builds its index once and repeats only query-time inference and evaluation. This discards no variability. HyperSU's indexing

Table 26: Statistical significance test over 9 independent runs, reported as mean ± standard deviation. GraphRAG-Bench columns average ACC over all eight domain–task settings (All) and over the six settings covering Complex Reasoning, Contextual Summarization, and Creative Generation (Reason.-Int.). Multi-hop QA columns use GPT-Acc. Underlined values mark the stronger reproduced hypergraph baseline, and Δ is the absolute gain of HyperSU over it. Test details are given in Appendix G.4.

Method	**GraphRAG-Bench**		**Multi-hop QA**		
	All	**Reason.-Int.**	**HotpotQA**	**2WikiMHQA**	**MuSiQue**
HyperGraphRAG	62.6±0.34	62.5±0.42	64.9±0.51	64.1±0.58	37.5±0.72
Hyper-RAG	63.2±0.37	63.1±0.46	65.6±0.47	69.1±0.52	36.4±0.76
HyperSU	**68.7±0.29**	**69.0±0.35**	**68.1±0.40**	**72.3±0.45**	**40.2±0.61**
Δ	+5.5	+5.8	+2.5	+3.2	+2.7

pipeline is deterministic. GLiNER extraction, embedding, dynamic-programming SU induction, and entity canonicalization return identical output for a fixed corpus, and no LLM is called during indexing. Rebuilding the index would therefore reproduce it exactly. The baselines instead repeat the full pipeline, including LLM-based hyperedge construction, so their runs differ at the index level as well. All LLM calls use temperature 0. The residual variation reflects nondeterminism of the serving API rather than deliberate sampling.

We compare HyperSU with the stronger baseline in each setting using a two-sided Welch's t-test. Welch's test is appropriate because the two groups have unequal variances by construction. A paired test does not apply, since runs are not matched across methods. We use Holm–Bonferroni correction over the five settings at $\alpha = 0.05$.

As shown in Table 26, HyperSU consistently outperforms both reproduced hypergraph baselines, and all corrected p-values fall below 0.001. The margins are large relative to the observed spread. The smallest gain, $+2.7$ on MuSiQue, is about four pooled standard deviations. The largest, $+5.5$ on GraphRAG-Bench All, is about sixteen. HyperSU also has the lowest standard deviation in all five settings. This is consistent with the difference in run scope described above. HyperSU's hyperedges are fixed once the corpus is fixed, whereas the baselines regenerate their hyperedges with an LLM in every run. The gains are therefore stable across repeated runs rather than being caused by a single random run or evaluation setting.