TopoRAG: Graph-based RAG via Topology-aware Approximate Nearest Neighbor Search

Anonymous ACL submission

Abstract

Retrieval-augmented generation (RAG) has become a core technique for improving the factuality and reasoning ability of large language models. Recent efforts extend RAG with graph-structured knowledge, enhancing retrieval to capture relational context beyond isolated text chunks. However, many graph-based RAG systems rely on a two-stage pipeline: (i) classical approximate nearest neighbor (ANN) search to identify top-k entities in the embedding space, (ii) heuristic neighbor expansion which augments the retrieved set by traversing immediate neighbors. This design underutilizes graph topology during retrieval and often introduces noisy or high-degree neighbors, leading to suboptimal evidence selection. In this paper, we propose TopoRAG, a retrieval framework that directly integrates structural constraints into ANN search via a diameter-constrained formulation. By selecting entities whose induced subgraph satisfies a diameter bound, TopoRAG enables topology-aware and noise-controlled graph retrieval. Experiments show that our approach consistently improves precision and significantly reduces context redundancy compared to existing methods.

1 Introduction

Retrieval-augmented generation (RAG) enhances large language models (LLMs) with external knowledge (Lewis et al., 2020; Gao et al., 2023; Fan et al., 2024; Chen et al., 2024). By retrieving relevant information from a large corpus and conditioning the generation process on retrieved evidence, RAG significantly improves reasoning, trustworthiness, and interpretability across a variety of tasks such as question answering (He et al., 2024; Xu et al., 2024), dialogue (Wang et al., 2024), and scientific discovery (Lála et al., 2023). However, classical RAG systems typically rely on vector-based retrieval over unstructured document collections, which is difficult to capture the rich relationships among entities and facts that naturally form graph-structured knowledge. To address this limitation, recent studies have explored graph-based RAG frameworks (Edge et al., 2024; Guo et al., 2024; Zhu et al., 2025; Zhou et al., 2025), enabling retrieval and reasoning to exploit both textual and structural contexts. As illustrated in Figure 1, graph-based RAG first transforms the corpus into a graph where entities are represented as nodes and semantic or relational links form the edges. At query time, the system retrieves graph information relevant to the user query, such as nodes, edges, subgraphs, or their associated textual data. The retrieved graph context is then combined with the query as part of the prompt, enabling the LLM to produce more grounded and context-aware responses. This graph-enhanced paradigm offers better expressiveness than vanilla RAG, supporting relational reasoning and multi-hop connections.

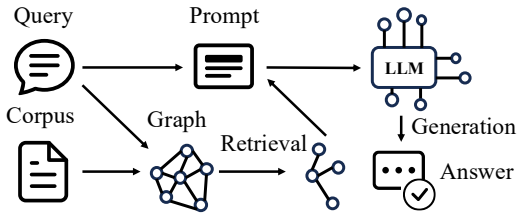
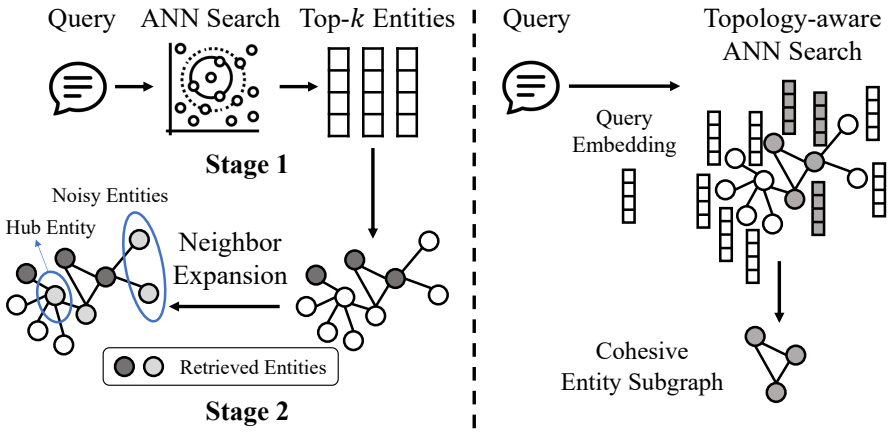


Figure 1: Overview of Graph-based RAG.

Despite these advantages, existing graph-based RAG pipelines still exhibit several limitations. Many current methods (Edge et al., 2024; Guo et al., 2024) first perform classical approximate nearest neighbor (ANN) search (Liu et al., 2004; Li et al., 2019) to obtain the top-k entities purely in the embedding space, and then heuristically expand these entities to include their neighbors in the pre-constructed graph. This two-stage retrieval strategy fails to integrate graph structure during the initial search, leading to suboptimal entity selection and redundant expansion. The naive neighbor expansion often introduces noisy or high-degree



(a) Vanilla Graph-based RAG (b) TopoRAG

Figure 2: Frameworks of Existing Graph-based RAG and TopoRAG.

("hub") nodes that weaken semantic relevance and reduce precision, as has been empirically noted in the earlier literature (Guo et al., 2025). As a result, the retrieval process becomes both computationally inefficient and structurally inconsistent, highlighting the limitations of current neighbor expansion pipelines and the need for more principled, structure-aware retrieval strategies.

To tackle these limitations, we formulate a new retrieval problem that incorporates explicit structural constraints into the ANN search process. Specifically, we aim to retrieve a set of entities whose induced subgraph satisfies a diameter bound, ensuring that the selected evidence is both semantically relevant and topologically coherent. Building on this formulation, we develop a diameter-constrained approximate nearest neighbor search framework that jointly reasons over embedding similarity and graph structure. This integrated design avoids the inefficiency of post-hoc expansion while preventing the inclusion of noisy or hub nodes, providing a principled alternative to existing neighbor expansion pipelines. Moreover, our framework is fully plug-and-play: it can operate directly on any pre-constructed graph derived from the corpus without requiring changes to the upstream graph-building pipeline.

Figure 2 compares our framework (dubbed as TopoRAG) with the classic graph-based RAG pipeline. In Figure 2(a), vanilla graph-based RAG follows a two-stage pipeline: Stage 1 performs embedding-only ANN search to retrieve the top-k nearest entities for semantic information extraction, and Stage 2 applies a neighbor expansion over the graph for structural information extraction. Since the semantic retrieval and structural reasoning are decoupled, this design often retrieves an entity set that is neither semantically coherent nor topologically consistent. More specifically, neighbor expansion often introduces two types of problematic entities. (i) Noisy entities have weak semantic correlation with the query, offering little useful evidence to the LLM; including them in the prompt not only adds irrelevant information but also increases token consumption. (ii) Hub entities exhibit high node degrees and are connected to a large number of unrelated documents. When such hubs are expanded, the retrieval process tends to pull in many extraneous documents, leading to oversized contexts that can mislead the LLM. In contrast, Figure 2(b) illustrates our topology-aware retrieval framework, which conducts a single-stage ANN search that directly incorporates graph structure. Instead of expanding after retrieval, our method jointly evaluates embedding similarity and graph topology during search, enabling the model to directly obtain a cohesive entity subgraph without introducing noisy or high-degree nodes.

Furthermore, to enhance the practical effectiveness of our framework, we incorporate several complementary optimizations. First, we augment the graph by leveraging strong entity co-occurrence within text chunks, thereby enriching structural connectivity beyond the corpus-derived links. Second, for complex or multi-faceted queries, we decompose the input into meaningful sub-queries and perform retrieval for each component to ensure broader and more robust coverage. Third, we adopt a hybrid retrieval strategy that complements entity-level search with chunk-level retrieval, providing additional textual evidence when entity signals are sparse or incomplete.

Our method achieves consistent performance gains across multiple QA benchmarks, outperforming both vanilla RAG and recent graph-based approaches. It delivers higher accuracy and recall on multi-hop reasoning tasks and shows markedly improved generation quality on narrative datasets. In contrast to retrieval pipelines that rely on post-hoc neighbor expansion, our unified retrieval design remains both effective and scalable.

Our contributions are summarized as follows:

- We introduce the first formulation of topology-aware ANN search for graph-based RAG, enabling retrieval to jointly consider embedding similarity and graph structure.
- We develop an efficient algorithm that directly retrieves a coherent, diameter-bounded subgraph, avoiding noisy or hub-dominated neighbor expansion and enabling structure-aware retrieval in a single stage.
- We conduct extensive experiments across multiple QA benchmarks, demonstrating substantial improvements over both vanilla RAG and recent graph-based retrieval methods.

2 Related Work

In this section, we review prior work on graph-based retrieval methods and approximate nearest neighbor search techniques.

2.1 Graph-based RAG

Graph-based RAG methods (Cao et al., 2024; Zhou et al., 2025) extend traditional retrieval-augmented generation by fusing the structure of knowledge graphs or document graphs into the RAG pipeline. Early works, such as (Lewis et al., 2020; Guu et al., 2020), rely on flat retrieval of textual chunks without considering inter-chunk relations. More recent methods, such as GraphRAG (Edge et al., 2024), HippoRAG (Jimenez Gutierrez et al., 2024), RAPTOR (Sarthi et al., 2024), and LightRAG (Guo et al., 2024), explicitly model the relationships between entities or chunks as graph edges, and perform multi-hop reasoning over the graph to enhance context selection. ArchRAG (Wang et al., 2025) introduces an attributed community-based hierarchical retrieval framework that leverages graph structure at multi-levels of abstraction. These approaches highlight the benefit of incorporating graph connectivity into retrieval, which motivates our study on graph-structured embeddings.

2.2 Approximate Nearest Neighbor Search

Approximate nearest neighbor (ANN) search is widely used for fast retrieval in high-dimensional embedding spaces (Li et al., 2019). Classic ANN methods are typically grouped into three representative categories: (i) partition-based methods, such as IVF (Moffat and Zobel, 1996) and LSH (Indyk and Motwani, 1998); (ii) graph-based methods, including NN-Descent (Dong et al., 2011), HNSW (Malkov and Yashunin, 2018), NSG (Fu et al., 2019), RNN-Descent (Ono and Matsui, 2023), MIRAGE (Voruganti and Özsu, 2025), and other proximity graphs (Yang et al., 2024); and (iii) quantization-based methods, such as PQ (Jegou et al., 2010), LSQ (Martinez et al., 2018), RabitQ (Gao et al., 2025), and SAQ (Li et al., 2025). These methods efficiently retrieve vectors similar to a query, but generally do not take graph structure into account, and therefore cannot be directly applied to diameter-bounded ANN search.

3 Topology-aware Framework

We now introduce our topology-aware framework, called TopoRAG, built on diameter-constrained ANN retrieval.

3.1 Overview

Our framework consists of an offline indexing stage and an online retrieval stage.

Offline Indexing. We begin by constructing an entity graph from the corpus and augmenting it with additional edges derived from entity co-occurrence within text chunks, which enriches the structural connectivity and improves downstream retrieval coherence. Moreover, we index the text chunks themselves to support flexible chunk-level retrieval. These components form a unified graph–text indexing structure that is compatible with any existing corpus-to-graph construction pipeline.

Online Retrieval. Given an input query, we first decompose complex queries into several focused sub-queries to ensure adequate coverage. For each sub-query, we perform *diameter-constrained approximate nearest neighbor search*, which retrieves a coherent set of entities whose induced subgraph satisfies a predefined diameter bound. This step forms the core of our framework, enabling retrieval that jointly reflects embedding similarity and graph topology. The retrieved entities are then mapped to their associated text chunks, and we optionally

supplement them with chunk-level retrieval to create a hybrid evidence set that combines entity-level precision with broader textual coverage. This integrated process yields structurally consistent, noise-controlled retrieval results while avoiding the pitfalls of neighbor expansion pipelines.

3.2 Problem Definition

We now introduce a novel retrieval formulation termed the diameter-constrained ANN search.

Given a graph-structured database $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where each node $v_i \in \mathcal{V}$ has an embedding vector $\mathbf{h}_i \in \mathbb{R}^d$, and a query embedding $\mathbf{h}_q$, we aim to find a subset of nodes $\mathcal{S}_q \subseteq \mathcal{V}$ of fixed size k that are semantically close to the query while forming a subgraph with bounded diameter.

Formally, the Diameter-Constrained Nearest Neighbor Search (DC-NNS) can be stated as:

$$\begin{aligned} &\max_{\mathcal{S}\subseteq\mathcal{V}} \Phi(q, \mathcal{S}) \\ \text{s.t.} \quad &|\mathcal{S}| = k, \\ &\operatorname{diam}(\mathcal{G}[\mathcal{S}]) \le D, \end{aligned} \tag{1}$$

where $\Phi(q, \mathcal{S})$ measures semantic relevance between query and selected nodes (e.g., average or sum of cosine similarities), and $\operatorname{diam}(\mathcal{G}[\mathcal{S}])$ is the diameter of the subgraph induced by $\mathcal{S}$. Intuitively, this problem retrieves the k most relevant nodes that are tightly connected in $\mathcal{G}$, enforcing locality and coherence of the retrieved subgraph.

Discussion. The DC-NNS problem can be seen as a generalization of several classical retrieval tasks. If the query embedding exactly matches the embedding of a node in the graph, the problem reduces to identifying a local community around that node. On the other hand, if the graph structure is ignored, the problem reduces to the classical approximate nearest neighbor search in the embedding space. This shows that our formulation naturally interpolates between purely semantic retrieval and graph-structured community detection. Nevertheless, as we will show, the problem is NP-hard, making exact solutions intractable and motivating the need for efficient approximation algorithms.

NP-hardness. We now show that the above problem is NP-hard even under simplified conditions.

Theorem 1. *The diameter-constrained nearest neighbor search is NP-hard, even when the semantic relevance term $\Phi(q, \mathcal{S})$ is replaced by a simple additive weight function $w : \mathcal{V} \to \mathbb{R}_{\ge 0}$ and the query embedding $\mathbf{h}_q$ is fixed.*

Proof. We reduce from the MAXIMUM WEIGHT CLIQUE problem, which is NP-hard.

An instance of MAXIMUM WEIGHT CLIQUE consists of an undirected graph $\mathcal{G}' = (\mathcal{V}', \mathcal{E}')$ and node weights $w' : \mathcal{V}' \to \mathbb{R}_{\ge 0}$. The goal is to find a clique $\mathcal{C} \subseteq \mathcal{V}'$ maximizing $\sum_{v\in\mathcal{C}} w'(v)$.

Given such an instance, we construct an instance of diameter-constrained nearest neighbor search as follows: let $\mathcal{G} = \mathcal{G}'$, let $w = w'$, fix the query embedding $\mathbf{h}_q$ arbitrarily, and set the diameter bound $D = 1$. We define the simplified relevance as $\Phi(q, \mathcal{S}) = \sum_{v\in\mathcal{S}} w(v)$.

We claim that a subset $\mathcal{S} \subseteq \mathcal{V}$ satisfies $\operatorname{diam}(\mathcal{G}[\mathcal{S}]) \le 1$ if and only if $\mathcal{S}$ is a clique in $\mathcal{G}$. Indeed, $\operatorname{diam}(\mathcal{G}[\mathcal{S}]) \le 1$ means every pair of distinct nodes in $\mathcal{S}$ has graph distance 1, i.e., every pair is connected by an edge; this is exactly the definition of a clique. Conversely, any clique has induced diameter 1.

Therefore, maximizing $\Phi(q, \mathcal{S})$ subject to $\operatorname{diam}(\mathcal{G}[\mathcal{S}]) \le 1$ is equivalent to finding a maximum-weight clique in $\mathcal{G}'$. Since MAXIMUM WEIGHT CLIQUE is NP-hard, the DC-NNS problem is NP-hard. □

Thus, exact solutions are computationally intractable for large-scale retrieval and we next propose efficient approximate methods for DC-NNS.

3.3 Algorithm

In this section, we propose two retrieval algorithms under the diameter-constrained setting: a post-filter approach and an in-filter approach. The post-filter algorithm performs standard ANN retrieval first, and then selects a subset of nodes that satisfy the diameter constraint in a separate post-processing step. In contrast, the in-filter algorithm integrates the diameter constraint directly into the retrieval process, selecting nodes that satisfy both semantic relevance and graph connectivity during the search.

Post-filter Algorithm. The post-filter algorithm consists of two main steps.

ANN Retrieval: Given a query embedding $\mathbf{h}_q$, retrieve a candidate set $\mathcal{C}_q$ of k' nearest neighbors from the embedding database using any standard ANN search method (e.g., HNSW, IVF, PQ). That is, $\mathcal{C}_q = \text{ANNS}(\mathbf{h}_q, k')$, where $k' \ge k$ is an oversampling factor to ensure sufficient candidates for the subsequent subgraph selection.

Diameter-Constrained Subgraph Selection: From the candidate set $\mathcal{C}_q$, select a subset $\mathcal{S}_q \subseteq \mathcal{C}_q$

of size k such that the induced subgraph $\mathcal{G}[\mathcal{S}_q]$ satisfies $\mathrm{diam}(\mathcal{G}[\mathcal{S}_q]) \leq D$. This can be formulated as a combinatorial selection problem, which is approximately solved by a greedy strategy: iteratively add nodes from $\mathcal{C}_q$ that minimally increase the subgraph diameter, until $|\mathcal{S}_q| = k$.

Algorithm 1: Post-filter DC-ANNS

Input: Graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, node embeddings $\{\mathbf{h}_i\}$, query $\mathbf{h}_q$, subset size k, diameter bound D, oversampling factor k'

Output: Subset $\mathcal{S}_q \subseteq \mathcal{V}$ with $|\mathcal{S}_q| = k$ and $\mathrm{diam}(\mathcal{G}[\mathcal{S}_q]) \leq D$

$\mathcal{C}_q \leftarrow \mathrm{ANNS}(\mathbf{h}_q, k')$ `// Retrieve candidate set via ANN`

$\mathcal{S}_q \leftarrow \{\}$

while $|\mathcal{S}_q| < k$ **do**

  $v^* \leftarrow \underset{v \in \mathcal{C}_q \setminus \mathcal{S}_q}{\mathrm{argmin}}\ \mathrm{diam}(\mathcal{G}[\mathcal{S}_q \cup \{v\}])$

  if $\mathrm{diam}(\mathcal{G}[\mathcal{S}_q \cup \{v^*\}]) \leq D$ **then**

    $\mathcal{S}_q \leftarrow \mathcal{S}_q \cup \{v^*\}$

  else

    Remove v^* from $\mathcal{C}_q$

return $\mathcal{S}_q$

Algorithm 1 shows the pseudo-code of the post-filter Diameter-Constrained ANN Search (DC-ANNS) approach. This approach leverages existing ANN techniques for efficient candidate retrieval and enforces the diameter constraint in a post-processing step. This post-filtering strategy is plug-and-play and can be easily combined with existing ANN methods without modifying the index. However, it may discard structurally important nodes and requires careful tuning of the oversampling factor k'. If k' is set too small, the ANN stage may fail to retrieve enough structurally compatible candidates, making it impossible to select k entities that satisfy the required diameter constraint. Conversely, choosing a large k' substantially increases the size of the candidate pool, forcing the post-filter step to examine many irrelevant or infeasible entities and resulting in significant computational overhead. This motivates a more integrated in-filter design, as described next.

In-filter Algorithm. To address the aforementioned issues, we now introduce an in-filter approach that enforces structural constraints directly during ANN search.

To avoid computing all pairwise distances when enforcing the diameter constraint, we first exploit a simple yet effective relationship between radius and diameter. Intuitively, if all nodes in a candidate set $\mathcal{C}$ lie within distance $D/2$ from some center node x, then any two nodes in $\mathcal{C}$ are at distance at most D. This yields a *radius-based sufficient condition* for satisfying the diameter constraint.

Lemma 1. *Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be an undirected graph with shortest-path distance $d_{\mathcal{G}}(\cdot, \cdot)$, and let $\mathcal{C} \subseteq \mathcal{V}$ be a candidate set of nodes. Suppose there exists a node $x \in \mathcal{C}$ such that*

$$d_{\mathcal{G}}(x, v) \leq \frac{D}{2}, \quad \forall v \in \mathcal{C}.$$

Then the induced subgraph on $\mathcal{C}$ satisfies the diameter constraint, i.e.,

$$\mathrm{diam}(\mathcal{C}) \;=\; \max_{u,w \in \mathcal{C}} d_{\mathcal{G}}(u, w) \;\leq\; D.$$

Proof. See Appendix A. □

The lemma provides a fast acceptance test: Once we find a center x such that all nodes in $\mathcal{C}$ lie inside its $D/2$-neighborhood, the diameter constraint is guaranteed to hold and we do not need to compute all pairwise distances.

To obtain a practical pruning strategy, we adopt a multi-center radius test as a heuristic filter. Instead of assuming that the candidate set admits a single center with radius at most $D/2$, we progressively enforce radius consistency with respect to multiple randomly selected centers. Concretely, we randomly select R temporary centers from the current candidate set. For each center x, we remove nodes v that violate the radius test, i.e., $d_{\mathcal{G}}(x, v) > D/2$. After iterating over all centers, any remaining node is within distance $D/2$ of all selected centers. By Lemma 1, using any of these centers as x, the induced subgraph over the remaining nodes has diameter at most D.

We further refine the in-filter procedure with a replacement-based heuristic. When the candidate set reaches its capacity, we first apply a necessary rejection test: a new node is considered for insertion only if it lies within distance D of at least one existing node, as formalized in Lemma 2.

Lemma 2. *Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be an undirected graph with shortest-path distance $d_{\mathcal{G}}(\cdot, \cdot)$. Let $\mathcal{S} \subseteq \mathcal{V}$ be a nonempty node set and let $v \in \mathcal{V} \setminus \mathcal{S}$. If*

$$d_{\mathcal{G}}(v, u) > D, \quad \forall\, u \in \mathcal{S},$$

Dataset	MultiHop-RAG	HotpotQA	NarrativeQA
Passages	609	9,221	1,572
Tokens	1,426,396	1,284,956	121,152,448
Nodes	23,353	37,436	650,571
Edges	30,716	30,758	679,426
Questions	2,556	1,000	43,304

Table 1: Datasets.

then we have $\mathrm{diam}(\mathcal{G}[\mathcal{S} \cup \{v\}]) > D$.

Proof. See Appendix A. □

We then attempt to replace a non-articulation node with the largest semantic distance to the query, and explicitly verify the diameter constraint after replacement. This strategy preserves structural connectivity while allowing the candidate set to adaptively incorporate semantically stronger nodes. More details can be found in Appendix B.

3.4 Optimization

We further introduce several optimizations to enhance the robustness and coverage of the topology-aware retrieval framework.

Graph Augmentation via Entity Co-occurrence. Given the original entity graph $G = (V, E)$, we augment its edge set using co-occurrence information extracted from text chunks. For each chunk c with entity set $\mathcal{E}(c) = \{e_1, e_2, \ldots, e_m\}$, we add undirected edges between all entity pairs:

$$(e_i, e_j) \in E_{\text{co}} \quad \text{for all } e_i, e_j \in \mathcal{E}(c),\ i \neq j.$$

Optionally, we assign co-occurrence weights $w_{ij} = \text{freq}(e_i, e_j)$, where $\text{freq}(e_i, e_j)$ counts the number of chunks in which e_i and e_j co-occur. The augmented graph is $G' = (V,\ E \cup E_{\text{co}})$.

Query Decomposition into Sub-queries. For complex user queries Q, we decompose the input into multiple focused sub-queries $\{q_1, q_2, \ldots, q_r\}$. Each sub-query is encoded independently $\mathbf{h}_{q_i} = f_{\text{enc}}(q_i)$. We then perform diameter-constrained ANN search for each sub-query:

$$S_i = \text{DC-ANNS}(G', \mathbf{h}_{q_i}, k, D),$$

and aggregate all retrieved entities: $S = \bigcup_{i=1}^{r} S_i$.

Hybrid Retrieval over Entities and Chunks. To complement entity-level retrieval, we additionally retrieve relevant text chunks. For each sub-query q_i, we compute similarity scores: $\text{score}(c, q_i) = \langle \mathbf{h}_{q_i}, \mathbf{h}_c \rangle$, and obtain the top-$k_c$ relevant chunks. The final evidence set combines entity-derived and chunk-derived content:

$$\mathcal{X} = \bigcup_{e \in S} \text{chunk}(e) \ \cup \ \bigcup_{i=1}^{k_c} \mathcal{C}_i.$$

4 Experiments

We now evaluate TopoRAG against RAG baselines across multiple QA benchmarks.

4.1 Setting

Datasets. We evaluate TopoRAG on three QA benchmarks commonly used in prior research (Ji et al., 2023; Shinn et al., 2023; Zhou et al., 2025; Wang et al., 2025). Specifically, we adopt MultiHop-RAG (Tang and Yang, 2024), HotpotQA (Yang et al., 2018), and NarrativeQA (Kočiskỳ et al., 2018). The detailed dataset statistics are summarized in Table 1.

Baselines. We compare TopoRAG with a broad set of baselines commonly used in recent studies (Zhou et al., 2025; Wang et al., 2025). These methods are selected as they are representative of current state-of-the-art retrieval and graph-based RAG approaches. As a retrieval-only baseline, we include **Vanilla RAG**, which performs vector-based nearest neighbor search over chunk embeddings. For graph-based systems, following (Wang et al., 2025), we evaluate representative methods including **RAPTOR** (Sarthi et al., 2024), **HippoRAG** (Jimenez Gutierrez et al., 2024), **LightRAG** (Guo et al., 2024), **LGraphRAG** and **GGraphRAG** (Edge et al., 2024), and **ArchRAG** (Wang et al., 2025). GGraphRAG conducts global community-level retrieval, whereas LGraphRAG performs local search.

Metrics & Implementation. For the first two QA datasets (MultiHop-RAG and HotpotQA), we follow prior work (Wang et al., 2025) and evaluate performance using **Accuracy** and **Recall** based on whether the generated answer contains the gold evidence rather than requiring an exact match. For NarrativeQA, we use its official generation metrics: **BLEU-1**, **METEOR**, and **ROUGE-L F1**, which measure lexical overlap and semantic coverage.

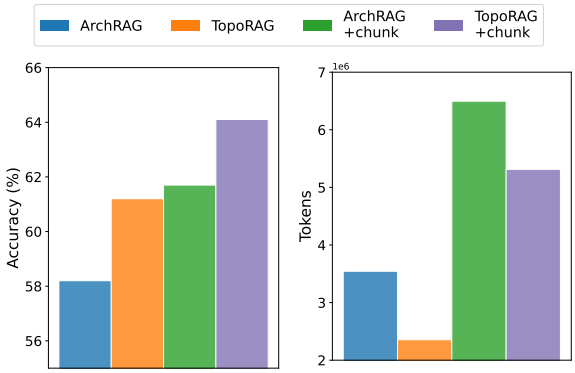
All baseline methods and our system use **GPT-3.5-turbo** as the default LLM and **text-embedding-small** as the embedding model. To ensure fairness, each method is required to complete both index construction and query execution within 24 hours.

Method	MultiHop-RAG		HotpotQA		NarrativeQA		
	Accuracy	Recall	Accuracy	Recall	BLEU-1	METEOR	ROUGE-L F1
Vanilla RAG	54.7	19.5	51.0	55.7	2.3	5.4	3.2
RAPTOR	59.9	27.6	OOT	OOT	5.7	12.2	9.3
HippoRAG	39.4	19.6	50.8	56.5	2.4	5.3	2.9
LightRAG	46.1	25.5	39.2	45.8	4.2	8.4	6.3
LGraphRAG	40.6	23.3	29.2	36.0	4.1	3.5	3.8
GGraphRAG	46.4	27.9	33.1	43.0	OOT	OOT	OOT
ArchRAG	58.2	27.1	53.9	56.5	11.2	18.5	17.6
Ours	**61.2**	**28.3**	**61.0**	**63.3**	**14.5**	**25.4**	**23.7**

Table 2: Performance comparison of different methods across various datasets. OOT: Not finished within 24 hours.

Method	MultiHop-RAG		HotpotQA	
	Time (s)	Tokens	Time (s)	Tokens
Vanilla RAG	2.4	3.6K	1.1	0.65K
RAPTOR	3.2	3.2K	OOT	OOT
HippoRAG	3.5	3.3K	2.3	0.72K
LightRAG	19.3	5.8K	13.9	3.1K
LGraphRAG	3.0	6.1K	2.7	4.8K
GGraphRAG	36.9	9.3K	34.5	9.0K
ArchRAG	2.58	6.0K	2.05	6.3K
Ours	1.88	4.8K	1.68	5.5K

Table 3: Time and token costs of different methods.



(a) Accuracy. (b) Token number.

Figure 3: Experimental results of TopoRAG and ArchRAG on the MultiHopRAG dataset.

Method	Time (s/query)	Tokens	Accuracy (%)
Post-filter	2.02	**4.5K**	59.8
In-filter	**1.88**	4.8K	**61.2**
In-filter w/o pruning	2.30	4.8K	60.5

Table 4: Comparison among post-filter, in-filter, and in-filter without pruning.

4.2 Experimental Results

We now present the empirical results of TopoRAG across multiple QA benchmarks, comparing it against (Graph-based) RAG baselines.

Main Results. As shown in Table 2, across all datasets, TopoRAG achieves consistently strong performance. On multi-hop QA benchmarks (MultiHop-RAG and HotpotQA), TopoRAG obtains the highest Accuracy and Recall, outperforming both retrieval-only baselines and graph-based RAG methods such as RAPTOR, LightRAG, and ArchRAG. Note that RAPTOR fails to complete on HotpotQA within 24 hours due to the high computational cost of its GMM-based text chunk clustering. On NarrativeQA, which requires deeper long-context semantic understanding, TopoRAG achieves clear improvements in BLEU-1, METEOR, and ROUGE-L F1, surpassing the strongest baseline (ArchRAG) by a large margin. GGraphRAG is marked as OOT on NarrativeQA due to excessive community-level overhead.

The underlying reason is that embedding-only ANN retrieval often produces fragmented subgraphs because embedding similarity does not necessarily align with graph connectivity (Kim et al., 2025). Such fragmentation limits semantic and inferential coherence. By enforcing structural constraints during retrieval, TopoRAG directly retrieves cohesive subgraphs, leading to more reliable downstream generation.

In terms of efficiency, TopoRAG maintains competitive retrieval latency and token usage. The results are reported in Table 3. While previous methods such as LightRAG and GGraphRAG, incur high computational overhead due to multi-stage graph traversal, TopoRAG achieves comparable or lower runtime on both MultiHop-RAG and HotpotQA. It also produces shorter prompts than most graph-based systems, reflecting the benefit of retrieving structurally coherent subgraphs without noisy or hub-induced expansion.

Ablation Study. We conduct a series of ablation studies on different datasets, evaluating the impact of modifying modules within our framework.

Figure 3 compares the performance of TopoRAG and ArchRAG with and without the additional chunk-level retrieval module. Without chunk-level retrieval, TopoRAG already surpasses ArchRAG

Method	Time (s/query)	Tokens	Accuracy (%)
Ours	2.51	5.8K	64.1
w/o query decomp.	1.62	4.6K	60.5
w/o graph augment.	1.55	4.2K	57.6
w/o chunk	1.88	4.8K	61.2

Table 5: Comparison among opimizations.

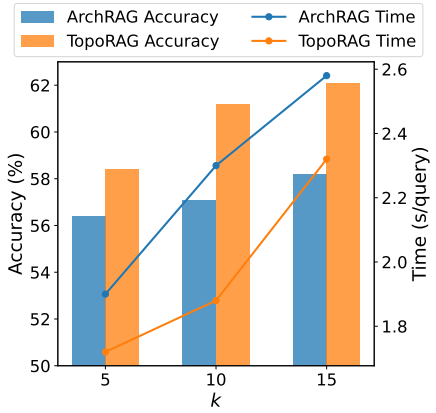
in accuracy while maintaining competitive query time and substantially lower token usage. This confirms that our topology-aware retrieval is effective and provides a more efficient alternative to vanilla graph-based RAG pipelines.

We then evaluate a variant of each method, denoted as +chunk, which augments entity-level retrieval with an extra retrieval step performed over chunk embeddings. This component allows the model to access multi-granularity contextual evidence rather than relying solely on entities.

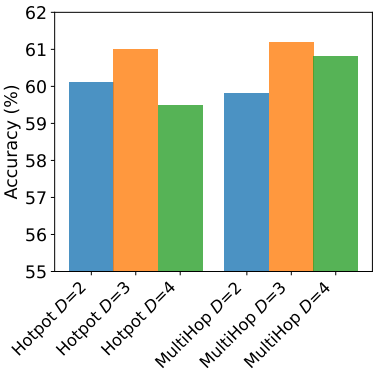
As shown in Figure 3, the +chunk module brings additional gains for both methods, but the improvements are particularly pronounced for TopoRAG. The TopoRAG+chunk configuration achieves the highest accuracy while still preserving a favorable token consumption. These results demonstrate that chunk-level retrieval complements our diameter-constrained entity retrieval by providing richer context, thereby enhancing multi-hop reasoning.

We further perform module-level ablations to isolate the contribution of our other design components. Table 4 examines different retrieval pipelines (post-filter, in-filter, and in-filter without pruning), highlighting the importance of performing diameter-constrained filtering within the entity graph. Table 5 presents additional ablations on three optimization modules of TopoRAG, including query decomposition, graph augmentation, and chunk-level retrieval. Together, these results demonstrate that each component contributes to the overall efficiency and accuracy of the framework.

Parameter Sensitivity Analysis. In addition to the module-level ablations, we investigate the sensitivity of TopoRAG to two key hyperparameters in the retrieval process. Figure 4(a) shows the effect of varying the number of ANN candidates k. Increasing k consistently improves accuracy for both TopoRAG and ArchRAG, as a larger candidate pool provides more relevant context for downstream reasoning. However, larger k also leads to longer inference time and higher token consumption. Across all values of k, TopoRAG achieves higher accuracy than ArchRAG, indicating that the effectiveness of its structure-aware filtering.



(a) Effectiveness with different k.



(b) TopoRAG accuracy with different D.

Figure 4: Experimental results of TopoRAG and ArchRAG with different parameters.

Figure 4(b) evaluates the diameter threshold used in our diameter-constrained ANN search. Increasing the allowable diameter enlarges the feasible subgraph and provides richer multi-hop evidence, leading to higher accuracy across datasets. However, an excessively large diameter leads to subgraphs that are less cohesive and the retrieved evidence becomes overly broad, which may introduce noise and negatively impact retrieval quality.

5 Conclusion

In this work, we presented TopoRAG, a structure-aware retrieval framework centered on a novel diameter-constrained ANN formulation. By enforcing topological coherence during retrieval, TopoRAG avoids noisy expansion and retrieves semantically meaningful subgraphs more effectively. Extensive experiments and ablations show that this design consistently improves QA performance while maintaining competitive efficiency. Our results underscore the value of integrating explicit graph-topological constraints into retrieval, paving the way for more principled and reliable graph-based RAG systems.

Limitations

Our framework focuses on retrieving compact, diameter-bounded subgraphs, which is effective for QA tasks but may not fully capture long-range or global dependencies required by broader reasoning settings. Additionally, TopoRAG currently does not support dynamically constructed graphs, as it relies on a pre-constructed entity graph.

References

Yukun Cao, Zengyi Gao, Zhiyang Li, Xike Xie, S Kevin Zhou, and Jianliang Xu. 2024. LEGO-GraphRAG: Modularizing graph-based retrieval-augmented generation for design space exploration. *arXiv preprint arXiv:2411.05844*.

Jiawei Chen, Hongyu Lin, Xianpei Han, and Le Sun. 2024. Benchmarking large language models in retrieval-augmented generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 17754–17762.

Wei Dong, Charikar Moses, and Kai Li. 2011. Efficient k-nearest neighbor graph construction for generic similarity measures. In *Proceedings of the 20th International Conference on World Wide Web*, pages 577–586.

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

Wenqi Fan, Yujuan Ding, Liangbo Ning, Shijie Wang, Hengyun Li, Dawei Yin, Tat-Seng Chua, and Qing Li. 2024. A survey on RAG meeting LLMs: Towards retrieval-augmented large language models. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 6491–6501.

Cong Fu, Chao Xiang, Changxu Wang, and Deng Cai. 2019. Fast approximate nearest neighbor search with the navigating spreading-out graph. *Proceedings of the VLDB Endowment*, 12(5):461–474.

Jianyang Gao, Yutong Gou, Yuexuan Xu, Jifan Shi, Zhonghao Yang, and Cheng Long. 2025. The RaBitQ library. In *The 1st Workshop on Vector Databases*.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yixin Dai, Jiawei Sun, Haofen Wang, and Haofen Wang. 2023. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*, 2(1).

Kai Guo, Xinnan Dai, Shenglai Zeng, Harry Shomer, Haoyu Han, Yu Wang, and Jiliang Tang. 2025. Beyond static retrieval: Opportunities and pitfalls of iterative retrieval in GraphRAG. *arXiv preprint arXiv:2509.25530*.

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

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. 2020. Retrieval augmented language model pre-training. In *International conference on machine learning*, pages 3929–3938. PMLR.

Xiaoxin He, Yijun Tian, Yifei Sun, Nitesh Chawla, Thomas Laurent, Yann LeCun, Xavier Bresson, and Bryan Hooi. 2024. G-retriever: Retrieval-augmented generation for textual graph understanding and question answering. *Advances in Neural Information Processing Systems*, 37:132876–132907.

Piotr Indyk and Rajeev Motwani. 1998. Approximate nearest neighbors: Towards removing the curse of dimensionality. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 604–613.

Herve Jegou, Matthijs Douze, and Cordelia Schmid. 2010. Product quantization for nearest neighbor search. *IEEE transactions on pattern analysis and machine intelligence*, 33(1):117–128.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38.

Bernal Jimenez Gutierrez, Yiheng Shu, Yu Gu, Michihiro Yasunaga, and Yu Su. 2024. HippoRAG: Neurobiologically inspired long-term memory for large language models. *Advances in Neural Information Processing Systems*, 37:59532–59569.

Soohyeong Kim, Seok Jun Hwang, JungHyoun Kim, Jeonghyeon Park, and Yong Suk Choi. 2025. ReGraphRAG: Reorganizing fragmented knowledge graphs for multi-perspective retrieval-augmented generation. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 5426–5443.

Tomáš Kočiskỳ, Jonathan Schwarz, Phil Blunsom, Chris Dyer, Karl Moritz Hermann, Gábor Melis, and Edward Grefenstette. 2018. The NarrativeQA reading comprehension challenge. *Transactions of the Association for Computational Linguistics*, 6:317–328.

Jakub Lála, Odhran O'Donoghue, Aleksandar Shtedritski, Sam Cox, Samuel G Rodriques, and Andrew D White. 2023. PaperQA: Retrieval-augmented generative agent for scientific research. *arXiv preprint arXiv:2312.07559*.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, and 1 others. 2020. Retrieval-augmented

generation for knowledge-intensive NLP tasks. *Advances in neural information processing systems*, 33:9459–9474.

Hui Li, Shiyuan Deng, Xiao Yan, Xiangyu Zhi, and James Cheng. 2025. SAQ: Pushing the limits of vector quantization through code adjustment and dimension segmentation. *Proceedings of the ACM on Management of Data*, 3(6):1–25.

Wen Li, Ying Zhang, Yifang Sun, Wei Wang, Mingjie Li, Wenjie Zhang, and Xuemin Lin. 2019. Approximate nearest neighbor search on high dimensional data—experiments, analyses, and improvement. *IEEE Transactions on Knowledge and Data Engineering*, 32(8):1475–1488.

Ting Liu, Andrew Moore, Ke Yang, and Alexander Gray. 2004. An investigation of practical approximate nearest neighbor algorithms. *Advances in Neural Information Processing Systems*, 17.

Yu A Malkov and Dmitry A Yashunin. 2018. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(4):824–836.

Julieta Martinez, Shobhit Zakhmi, Holger H Hoos, and James J Little. 2018. LSQ++: Lower running time and higher recall in multi-codebook quantization. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 491–506.

Alistair Moffat and Justin Zobel. 1996. Self-indexing inverted files for fast text retrieval. *ACM Transactions on Information Systems (TOIS)*, 14(4):349–379.

Naoki Ono and Yusuke Matsui. 2023. Relative NN-Descent: A fast index construction for graph-based approximate nearest neighbor search. In *Proceedings of the 31st ACM International Conference on Multimedia*, pages 1659–1667.

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*.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. Reflexion: Language agents with verbal reinforcement learning. *Advances in Neural Information Processing Systems*, 36:8634–8652.

Yixuan Tang and Yi Yang. 2024. MultiHop-RAG: Benchmarking retrieval-augmented generation for multi-hop queries. In *First Conference on Language Modeling*.

Sairaj Voruganti and M Tamer Özsu. 2025. MIRAGE-ANNS: Mixed approach graph-based indexing for approximate nearest neighbor search. *Proceedings of the ACM on Management of Data*, 3(3):1–27.

Hongru Wang, Wenyu Huang, Yang Deng, Rui Wang, Zezhong Wang, Yufei Wang, Fei Mi, Jeff Z Pan, and Kam-Fai Wong. 2024. UniMS-RAG: A unified multi-source retrieval-augmented generation for personalized dialogue systems. *arXiv preprint arXiv:2401.13256*.

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

Zhentao Xu, Mark Jerome Cruz, Matthew Guevara, Tie Wang, Manasi Deshpande, Xiaofeng Wang, and Zheng Li. 2024. Retrieval-augmented generation with knowledge graphs for customer service question answering. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2905–2909.

Shuo Yang, Jiadong Xie, Yingfan Liu, Jeffrey Xu Yu, Xiyue Gao, Qianru Wang, Yanguo Peng, and Jiangtao Cui. 2024. Revisiting the index construction of proximity graph-based approximate nearest neighbor search. *arXiv preprint arXiv:2410.01231*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William 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.

Yingli Zhou, Yaodong Su, Youran Sun, Shu Wang, Taotao Wang, Runyuan He, Yongwei Zhang, Sicong Liang, Xilin Liu, Yuchi Ma, and 1 others. 2025. In-depth analysis of graph-based RAG in a unified framework. *arXiv preprint arXiv:2503.04338*.

Zulun Zhu, Tiancheng Huang, Kai Wang, Junda Ye, Xinghe Chen, and Siqiang Luo. 2025. Graph-based approaches and functionalities in retrieval-augmented generation: A comprehensive survey. *arXiv preprint arXiv:2504.10499*.

A Proofs

A.1 Proof of Lemma 1

Let $u, w \in \mathcal{C}$ be arbitrary. By the triangle inequality,

$$d_{\mathcal{G}}(u, w) \;\le\; d_{\mathcal{G}}(u, x) + d_{\mathcal{G}}(x, w).$$

By the assumption on x, we have $d_{\mathcal{G}}(u, x) \le D/2$ and $d_{\mathcal{G}}(x, w) \le D/2$, hence

$$d_{\mathcal{G}}(u, w) \;\le\; \frac{D}{2} + \frac{D}{2} \;=\; D.$$

Since this holds for any pair $u, w \in \mathcal{C}$, we obtain

$$\operatorname{diam}(\mathcal{C}) = \max_{u,w \in \mathcal{C}} d_{\mathcal{G}}(u, w) \le D,$$

which completes the proof.

Algorithm 2: Radius-Based Candidate Pruning with Multiple Random Centers

Input: candidate set $\mathcal{C}$, graph $\mathcal{G}$, diameter threshold D, number of random centers R

Output: pruned candidate set $\mathcal{C}'$

```
C' ← C
for r = 1 to R do
    select random center x ∈ C'
    for v ∈ C' do
        if d_G(x, v) > D/2 then
            remove v from C'
    if C' is empty then
        break
return C'
```

A.2 Proof of Lemma 2

Since $\mathcal{S}$ is nonempty, pick any $u \in \mathcal{S}$. By assumption, $d_{\mathcal{G}}(v, u) > D$. Therefore,

$$\begin{aligned}\operatorname{diam}(\mathcal{G}[\mathcal{S} \cup \{v\}]) &= \max_{x,y \in \mathcal{S} \cup \{v\}} d_{\mathcal{G}}(x, y) \\ &\geq d_{\mathcal{G}}(v, u) > D,\end{aligned}$$

which proves the claim.

B In-filter Algorithms

Algorithm 2 summarizes the randomized radius-based candidate pruning procedure.

Overall In-Filter Workflow. Given a pre-built proximity graph index such as HNSW or an RNG-based graph, the in-filter procedure efficiently identifies a diameter-feasible neighborhood around the query. Starting from the entry node of the graph, we perform a guided graph traversal to obtain an initial candidate set $\mathcal{C}_0$ consisting of nodes that are both close to the query embedding and structurally reachable under the local graph topology.

To enforce the global structural constraint without computing all pairwise distances, we apply the radius-based pruning mechanism described above. We first choose R temporary centers (either randomly or based on proximity to the query). For each center x, any node whose graph distance exceeds $D/2$ is removed from the candidate set. After iterating through all R centers, the remaining candidate set $\mathcal{C}$ is guaranteed to satisfy the radius condition with respect to at least one of the centers, which serves as a sufficient structural certificate for diameter feasibility. This multi-center pruning produces a compact, topologically consistent neighborhood that respects the diameter constraint while avoiding expensive pairwise distance computations. The resulting candidate set $\mathcal{C}$ is then passed to the final scoring step to select the top-k answers.