KAMR: Grounding Generation via Knowledge-Aligned Multi-hop Retrieval

Anonymous ACL submission

Abstract

Graph-based retrieval-augmented generation increasingly relies on multi-hop retrieval, where answering a query requires composing multiple connected knowledge-graph triplets. However, existing retrievers often rank triplets independently with global semantic matching, and common multi-hop benchmarks provide only final answers, leaving retrievers without adaption on query–triplet alignment and causing structurally necessary but weakly aligned facts to be missed. To address these issues, we propose a novel knowledge-aligned multi-hop retriever, namely KAMR, which distinguishes anchor triplets that are strongly constrained by the query from connected triplets that are weakly aligned yet structurally linked to the anchors. To tackle the absence of query–triplet alignment, we build a partial alignment dataset by masking triplet elements and prompting an LLM to generate corresponding queries, and optimize two contrastive objectives for pair-level and element-level matching. At inference time, KAMR retrieves anchors globally and expands locally to return connected evidence. Across two benchmarks, three LLM backbones, and fourteen baselines, KAMR consistently improves multi-hop retrieval and downstream question answering tasks. [1]

1 Introduction

Retrieval-Augmented Generation (RAG) systems increasingly rely on **multi-hop retrieval** to support complex reasoning over external knowledge sources. Unlike single-hop retrieval, which assumes that relevant evidence is directly aligned with the input query, multi-hop retrieval must cover multiple pieces of information that are only indirectly connected to the query and must be composed to produce correct answers. This setting naturally arises in knowledge graph (KG) reasoning, where answering a query often requires traveling multiple connected triplets. As a result, graph-based multi-hop retrieval has become a critical component for tasks such as multi-hop question answering and knowledge-grounded reasoning.

Previous work tends to treat the graph-based multi-hop retrieval with a conventional searching strategy. Many of these approaches retrieve triplets independently using global semantic matching between the query and candidate evidence with simple sparse retriever (Hu et al., 2025; Li et al., 2023) and dense retriever (Dong et al., 2023; Li et al., 2023; Hu et al., 2025). While effective in single-hop settings, such strategies struggle in multi-hop scenarios, where some required evidence exhibits weak or no direct textual alignment with the query. Recent Graph GAG methods (Hu et al., 2024; He et al., 2024; Li et al., 2024) partially address this issue by leveraging structural connectivity in the knowledge graph. Still, they largely rely on off-the-shelf semantic retrievers pretrained on generic text corpora whose objectives focus on text-to-text matching rather than query-triplet alignment. Consequently, existing methods frequently fail to balance semantic relevance and structure composability, leading to either incomplete reasoning chains or the retrieval of noisy, weakly related evidence.

At the core of this difficulty lies a fundamental challenge of graph-based multi-hop retrieval: queries are often underspecified with respect to the required triplets. In many cases, a query directly constrains only a part of the relevant evidence, while the remaining triplets are connected through intermediate entities or relations. As shown in Figure 1, the query "*What country is the city where Joan of Arc was captured in?*" can be answered by composing two triplets: (*Joan of Arc, captured in, Compiègne*) and (*Compiègne, located in, France*). The first triplet is strongly constrained by the entity-relation pair "*Joan of Arc*" and "*captured in*" and thus exhibits high semantic alignment with the

[1] The source code and data are included in the supplementary materials as Software and will be made publicly available after the review stage.

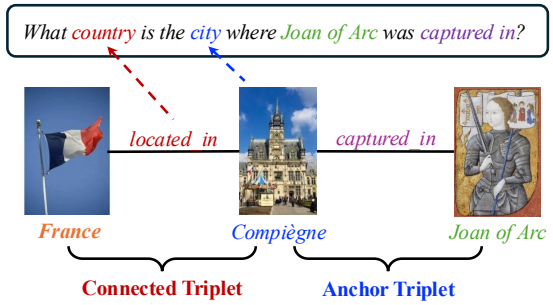

Figure 1: An example demonstrating the necessity of differentiating heterogeneous triplet types and explicitly modeling structural connectivity for multi-hop reasoning.

query. In contrast, the second triplet is connected to the query only through the intermediate entity "*Compiègne*" and the implicitly query-related relation "*located in*". Therefore, it exhibits weak textual alignment with the query and is unlikely to be retrieved by global semantic matching alone. As a result, treating all triplets equally aligned with the query is suboptimal. Instead, different triplets play different roles in the reasoning process and should be retrieved using different criteria. However, existing retrievers typically lack mechanisms to explicitly model such differences.

Another major challenge is pretraining graph-based multi-hop retrievers *without query-triplet alignment supervision*. Although some methods have attempted to tune retrievers (Li et al., 2024), they typically assume access to explicit query–triplet alignment signals for supervision. However, benchmarks such as Complex Web Questions (CWQ) (Talmor and Berant, 2018) provide only final answers and do not explicitly annotate which knowledge triplets should be retrieved to support multi-hop reasoning. Thus, it is challenging to train retrievers that can distinguish between strongly aligned evidence and weakly aligned but structurally necessary evidence. Moreover, naive supervision at the triplet level ignores the fact that queries often specify only partial elements of a triplet, further exacerbating the mismatch between training objective and inference requirements.

To address these challenges, we propose a novel **K**nowledge-**A**ligned **M**ulti-hop **R**etriever (KAMR) that explicitly models partial query-triplet alignment and structural connectivity. Our key insight is to distinguish between two types of retrieval targets: **anchor triplets**, which are strongly constrained by the query and exhibit high semantic alignment, and **connected triplets**, which are weakly aligned with the query but are structurally connected to anchor triplets and essential for the multi-hop reasoning.

To mitigate the issue of lack of training supervision, we construct a partial query-triplet alignment dataset by masking individual triplet elements and prompting a large language model (LLM) to generate corresponding queries. This allows us to pretrain the proposed retriever KAMR using two complementary contrastive objectives: a pair-level loss that aligns queries with partial triplets for anchor retrieval, and an element-level loss that aligns queries with individual triplet elements to support connected triple retrieval. Importantly, this design avoids the need for explicit multi-hop supervision while aligning naturally with our two-stage inference procedure: we first retrieve anchor triplets, then expand to structurally connected triplets within their local neighborhoods to complete multi-hop evidence via graph connectivity.

Our contribution can be summarized as follows: (1) We identify and formalize the distinction between anchor triplets and connected triplets in graph-based multi-hop retrieval, highlighting their different roles and alignment properties. (2) We propose a novel graph-based multi-hop retrieval pretraining framework KAMR that combines pair-level and element-level matching to balance semantic relevance and structural connectivity. (3) We construct a partial query-triplet alignment dataset that enables effective retriever pretraining without explicit multi-hop annotation. (4) We demonstrate that the proposed KAMR consistently improves multi-hop retrieval performance across one multi-hop retrieval task and two multi-hop question-answering tasks on three LLM backbones, outperforming fourteen baselines.

2 The Proposed KAMR Framework

2.1 Task Definition

In contrast to existing graph-based RAG methods that retrieve triplets independently, multi-hop retrieval explicitly enforces **structural connectivity** among the retrieved triplets. Specifically, the retrieved set must contain *at least one connected pair of triplets* that share a common head or tail entity, thereby enabling multi-hop reasoning over the graph. Thus, we mathematically define graph-based multi-hop retrieval as follows:

Definition 1 (Graph-based Multi-hop Retrieval)**.** *Let $\mathcal{G} = (\mathcal{E}, \mathcal{R}, \mathcal{T}_\mathcal{G})$ be a knowledge graph, where $\mathcal{E}$ is the entity set, $\mathcal{R}$ is the relation set, and $\mathcal{T}_\mathcal{G} = \{\mathcal{E} \times \mathcal{R} \times \mathcal{E}\}$ is the set of all triplets. Given a*

textual query q, the objective of graph-based multi-hop retrieval is to select a subset $\mathcal{T}_q \subset \mathcal{T}_\mathcal{G}$ that is semantically relevant to q, i.e.,

$$\mathcal{T}_q = \textit{Retriever}(q, \mathcal{G}),$$

and satisfies the following multi-hop connectivity constraint:

$$\begin{aligned} &\exists (h_i, r_i, t_i), (h_j, r_j, t_j) \in \mathcal{T}_q, \\ &\textit{s.t.} \quad \{h_i, t_i\} \cap \{h_j, t_j\} \neq \emptyset, \\ &|\mathcal{T}_q| = K \ll |\mathcal{T}_\mathcal{G}|, \end{aligned}$$

where $h_i, t_i \in \mathcal{E}$ and $r_i \in \mathcal{R}$.

2.2 Targets of Multi-hop Retrieval

An ideal graph-based multi-hop retriever should return a set of triplets $\mathcal{T}_q$ that are not only well aligned with the query q, but also provide the missing information required to answer it. In the multi-hop setting, however, these two objectives often conflict, as queries are frequently underspecified and do not fully constrain all relevant facts. This observation motivates targeted mechanisms that explicitly account for query–triplet misalignment in multi-hop retrieval. Accordingly, we distinguish two types of retrieval targets based on the degree to which a triplet is constrained by the current context.

Definition 2 (Anchor Triplets)**.** *Let $\mathcal{S}_q$ denote the sets of entities and relations extracted from the query q. A triplet $\tau_i = (h_i, r_i, t_i) \in \mathcal{T}_\mathcal{G}$ is called an **anchor triplet** if the query q specifies exactly two elements of τ_i, i.e.,*

$$\tau_i \in \mathcal{A}_q \quad \textit{s.t.} \quad |\mathcal{S}_q \cap \textit{Elem}(\tau_i)| = 2,$$

where $\mathcal{A}_q$ denotes the set of anchor triplets for q, and $\textit{Elem}(\tau_i)$ represents the element set of τ_i.

Definition 3 (Connected Triplets)**.** *A triplet τ_j is called a **connected triplet**, which satisfies the following constraint:*

$$\begin{aligned} &\tau_j \in \mathcal{C}_q, \quad \textit{s.t.} \quad |\mathcal{S}_q \cap \textit{Elem}(\tau_j)| = 1, \ \textit{and} \\ &\exists\, \tau_i \in \mathcal{A}_q \ \textit{ such that } \ \{h_i, t_i\} \cap \{h_j, t_j\} \neq \emptyset, \end{aligned}$$

where $\mathcal{C}_q$ is the set of connected triplets for q.

(*Joan of Arc, captured in, Compiègne*) shown in Figure 1 is the anchor triple, which shares two elements with the query, and (*Compiègne, located in, France*) is the connected triplet, which has one semantically shared element "*located in*" with the query and one shared element "*Compiègne*" with the anchor triplet. The anchor triplets are typically easier to retrieve due to their strong semantic alignment with the query, but they are insufficient for multi-hop reasoning on their own. Connected triplets complement anchor triplets by introducing new, structurally connected evidence, thereby enabling effective multi-hop expansion.

2.3 KAMR Pretraining

Training a multi-hop retriever poses significant challenges for two main reasons. First, there is a lack of query–triplet aligned supervision. For instance, although benchmarks such as Complex Web Questions (CWQ) (Talmor and Berant, 2018) have been widely adopted for knowledge graph-based question answering, they provide only final answers and do not include explicit annotations indicating which triplets should be retrieved during reasoning.

Second, as discussed above, multi-hop reasoning requires retrieving both anchor and connected triplets, yet the query typically specifies only partial elements of the relevant triplets. Consequently, effective retrieval cannot rely on matching the query against fully specified triplets. Instead, the retriever must model the correspondence between the textual query and **partially specified triplets**, which substantially complicates pretraining and supervision.

To address these challenges, we first use LLMs to construct a supervision dataset for multi-hop retriever pretraining, and then introduce a tailored pretraining loss that accounts for partial query–triplet alignment.

2.3.1 Partial Alignment Dataset Construction

Given a complete triplet $\tau_i = (h_i, r_i, t_i) \in \mathcal{T}_\mathcal{G}$, we generate three partial triplets by randomly masking one of its elements, yielding the set,

$$\mathcal{U}(\tau_i) = \{(h_i, r_i), (r_i, t_i), (h_i, t_i)\}. \qquad (1)$$

Each partial triplet can be converted into a natural language query by prompting an LLM[2].

Let $u_i = (e_i^1, e_i^2) \in \mathcal{U}(\tau_i)$ denote a partial triplet, where $e_i^k \in \{h_i, r_i, t_i\}$ and $k = 1, 2$. q_i denotes the corresponding query generated from u_i. We construct a pretraining dataset $\mathcal{U} = \{(q_i, u_i)\}_{i=1}^{|\mathcal{U}|}$, where $|\mathcal{U}|$ is the total number of partial query–triplet alignment pairs. This dataset, detailed in Appendix C.2, is used to pretrain the multi-hop retriever.

[2]The prompt is provided in Appendix B.1.

2.3.2 Pretraining Loss Design

Since graph-based multi-hop retrieval aims to retrieve both anchor and connected triplets, we define the overall pretraining objective as the sum of two loss terms:

$$\mathcal{L} = \mathcal{L}_a + \mathcal{L}_c, \tag{2}$$

where $\mathcal{L}_a$ corresponds to the pretraining loss for anchor triplets, and $\mathcal{L}_c$ corresponds to the pretraining loss for connected triplets. We next describe the formulation of each loss term in detail.

Anchor Triplet Retrieval Pretraining. Given a partial query-triplet alignment pair (q_i, u_i), we first encode the query and the corresponding partial triplet using their corresponding text encoders $\text{ENC}_q()$ and $\text{ENC}_u()$ as follows:

$$\mathbf{q}_i = \text{ENC}_q(q_i), \quad \mathbf{u}_i = \text{ENC}_u(e_i^1 \oplus e_i^2), \tag{3}$$

where $\oplus$ denotes concatenation. We then optimize their alignment using the InfoNCE loss (Oord et al., 2018):

$$\mathcal{L}_a = -\frac{1}{|\mathcal{B}|}\sum_{i=1}^{|\mathcal{B}|}\log\frac{\exp(\text{sim}(\mathbf{q}_i, \mathbf{u}_i)/T)}{\sum_{j=1}^{|\mathcal{B}|}\exp(\text{sim}(\mathbf{q}_i, \mathbf{u}_j)/T)}, \tag{4}$$

where $\mathcal{B} \subseteq \mathcal{U}$ denotes the mini-batch of training samples, T is a temperature hyperparameter, and the remaining partial triplets within the batch serve as negatives.

Connected Triplet Retrieval Pretraining. When constructing the partial alignment dataset, we do not explicitly create multi-hop supervision pairs for connected triplet pretraining. This is because the partial query–triplet alignment samples are sufficient to support the pretraining of both anchor and connected triplets. By Definition 3, a connected triplet shares only one element with the query, while its remaining element is connected to an anchor triplet. Consequently, retrieving connected triplets still requires identifying the element that overlaps with the query based on semantic similarity.

This observation motivates us to directly model *element-level similarity* between the query q_i and individual elements in u_i, rather than relying on explicit multi-hop supervision during pretraining. Accordingly, we define another InfoNCE loss to pretrain the retriever with connected triplets as follows:

$$\mathcal{L}_c = -\frac{1}{2|\mathcal{B}|}\sum_{i=1,k=1}^{|\mathcal{B}|,2}\log\frac{\exp(\text{sim}(\mathbf{q}_i, \mathbf{e}_i^k)/T)}{\sum_{j=1,k=1}^{|\mathcal{B}|,2}\exp(\text{sim}(\mathbf{q}_i, \mathbf{e}_j^k)/T)}, \tag{5}$$

where $\mathbf{e}_i^k = \text{ENC}_u(e_i^k)$ is the embedding of the k-th element $e_i^k \in u_i$ $(k = 1, 2)$.

2.4 KAMR Inference

After optimizing the pretraining objective in Eq. (2) on the constructed dataset $\mathcal{U}$, we directly deploy the pretrained graph-based multi-hop retriever to a variety of downstream tasks. During inference, in accordance with our retrieval objectives, the retriever first identifies *anchor triplets* via the pair-level matching learned from anchor-triplet pretraining, and then retrieves *connected triplets* from the neighborhoods of these anchor triplets. This two-stage retrieval process simultaneously promotes structural connectivity and improves local retrieval precision.

Anchor Triplet Retrieval. Each triplet $\tau_i \in \mathcal{T}_\mathcal{G}$ is first converted into a set of partial triplets $\mathcal{U}(\tau_i)$ according to Eq. (1). We then compute the cosine similarity between the query embedding $\mathbf{q}$ and the embedding of each partial triplet in $\mathcal{U}(\tau_i)$, where embeddings are obtained using Eq. (3). The maximum similarity score among the three partial query–triplet pairs is taken as the overall similarity between the query q and the triplet τ_i. Finally, we retrieve the top-M triplets as anchor candidates, i.e., $|\mathcal{A}_q| = M$.

Connected Triplet Retrieval. For each anchor triplet $\tau_i \in \mathcal{A}_q$, we first extract its g-hop neighborhood from the knowledge graph $\mathcal{G}$. We then select a set of connected triplet candidates $X(\tau_i)$ according to Definition 3. For each candidate triplet $x \in X(\tau_i)$, we remove the element shared with τ_i and then embed the remaining two elements in $\text{Elem}(x) - \text{Elem}(\tau_i)$ using the element encoder $\text{ENC}_u(\cdot)$. We then calculate the cosine similarity between the query embedding $\mathbf{q}$ and each element-level embedding. Similar to the anchor triplet selection, the maximum similarity score between the query and the remaining elements is used as the score for candidate triplet x.

We retrieve top-N connected triplets for each anchor triplet τ_i, denoted as $\mathcal{C}_q(\tau_i)$. The final set of connected triplets is obtained by taking the union over all anchor triplets, i.e., $\mathcal{C}_q = \bigcup_{\tau_i \in \mathcal{A}_q} \mathcal{C}_q(\tau_i)$.

Final Multi-hop Evidence. We return the retrieved multi-hop evidence set in the form required by Definition 1:

$$\mathcal{T}_q = \mathcal{A}_q \cup \mathcal{C}_q, \tag{6}$$

where $|\mathcal{A}_q| = M$, each anchor retrieves N connected triplets, and thus $|\mathcal{T}_q| = M + M * N = K$.

This inference strategy is well aligned with our pretraining objectives: anchor retrieval relies on pair-level matching to address partial query–triplet alignment, while connected retrieval leverages element-level matching under anchor-induced structural constraints to recover weakly aligned yet composable multi-hop evidence. Algorithm 1 in Appendix B.2 summarizes the overall retrieval procedure.

3 Experiments

3.1 Experiment Settings

Datasets. We evaluate KAMR in a staged manner, from multi-hop retrieval quality to multi-hop retrieval-based applications. For the retrieval quality evaluation, we use PathQuestion (**PQ-2H** for questions depending on 2-hop facts and **PQ-3H** for those relying on 3-hop triplets) (Zhou et al., 2018), which provides gold reasoning paths and enables explicit measurement of whether a retriever can recover the supporting triplets. Question answering is a representative task in multi-hop retrieval. Correspondingly, we consider two datasets with complementary characteristics: PathQuestion, which features a single answer and fixed 2- and 3-hop reasoning chains, and Complex Web Questions (**CWQ**) (Talmor and Berant, 2018), which allows multiple valid answers and involves mixed-hop reasoning.

Backbones. Our retrieval strategies are tested with several large language models (LLMs) backbones pretrained on general-domain corpora, including ChatGPT-3.5 Turbo (Achiam et al., 2023), LLaMA2-7B (Touvron et al., 2023), and Qwen3-8B (Yang et al., 2025). These models span different scales and include both open- and closed-source LLMs, ensuring a comprehensive evaluation across architectures.

Baselines. We compare KAMR against four categories of retrievers. (1) *Lexical-based* methods include BM25 (Robertson et al., 2009) and TF-IDF (Sparck Jones, 1972). (2) *Semantic-based* dense retrievers include DistilBERT (Sanh et al., 2019), BGE (Chen et al., 2024), BCE (Youdao, 2023), Jina (Günther et al., 2023), MXBAI (Shakir et al., 2024), SPLADE (Formal et al., 2021), ColBERTv2 (Santhanam et al., 2022), Dragon (Lin et al., 2023), and Hybrid (Li et al., 2023). (3) *Structure-based* graph retrievers include G-RAG (Hu et al., 2024) and G-Retriever (He et al., 2024). (4) *Graph-pretraining* method SKP (Dong et al., 2023). Detailed configurations and implementation for all baselines as well as KAMR are provided in Appendix C.2.

Table 1: Retrieval performance on PathQuestion.

Retriever Category	Method	PQ-2H		PQ-3H	
		Triplet Recall	Path Recall	Triplet Recall	Path Recall
Lexical Based	BM25	63.47	26.94	63.48	40.32
	TF-IDF	63.78	27.57	64.45	40.52
Semantic Based	BCE	77.83	55.66	72.03	49.13
	BGE	83.49	66.98	76.73	55.89
	ColBERT	80.37	62.16	72.60	48.15
	DistilBERT	34.83	17.77	27.98	10.04
	Dragon	85.95	71.91	78.71	59.10
	Hybrid	85.88	71.75	79.18	59.35
	Jina	82.34	65.04	74.18	51.31
	MXBAI	10.59	1.47	6.25	0.50
	SPLADE	80.16	60.32	75.07	53.10
Structure Based	G-RAG	92.92	86.90	59.04	30.92
	G-Retriever	14.47	9.91	4.14	2.81
Graph Pretraining	SKP	75.68	57.76	53.26	31.05
	KAMR	**96.36**	**92.87**	**85.60**	**67.06**

3.2 Multi-hop Retrieval Quality Evaluation

We explicitly evaluate retrieval quality on PathQuestion (PQ-2H and PQ-3H) using two metrics with a 50 budget. **Triplet Recall** measures whether each gold triplet appearing in the annotated reasoning path is retrieved within the budget. **Path Recall** measures whether the retriever recovers the entire reasoning path, i.e., all triplets required to form the gold multi-hop chain.[3]

Table 1 reports retrieval performance on PathQuestion under the same retrieval budget. We can observe that KAMR achieves the best results on both PQ-2H and PQ-3H, with particularly strong Path Recall, indicating more stable recovery of complete multi-hop chains. Even the stronger baselines still fall short of a consistently high Path Recall, especially on PQ-3H, indicating that recovering complete multi-hop chains remains challenging under a fixed retrieval budget. In addition, we observe two representative failure cases among baselines. MXBAI is primarily a reranker and typically requires a lexical retriever (e.g., Solr (Shahi, 2015)) for coarse filtering, so using it alone leads to many irrelevant candidates. G-Retriever's heuristic prize-based subgraph extraction optimizes the overall subgraph score, which can introduce non-essential triplets and dilute evidence-critical retrieval.

[3]Note that we retrieve two different types of triplets, making it impossible to use other ranking metrics such as nDCG in the evaluation.

Table 2: Question answering accuracy (%) on PathQuestion (PQ-2H / PQ-3H) across three LLM backbones. The best and second-best results per backbone and dataset are highlighted in **bold** and underline, respectively.

Retriever Category	Method	Qwen3-8B		LLaMA2-7B		ChatGPT-3.5	
		PQ-2H	PQ-3H	PQ-2H	PQ-3H	PQ-2H	PQ-3H
✗	LLM-only	46.09	33.54	12.47	8.50	39.47	26.24
Lexical Based	BM25	42.66	56.21	48.32	49.77	51.83	53.89
	TF-IDF	42.08	55.19	46.70	49.25	50.10	52.23
Semantic Based	BCE	57.04	57.63	56.18	48.53	60.80	52.09
	BGE	63.65	60.39	60.32	50.98	67.35	54.15
	ColBERT	59.04	56.64	59.06	50.08	62.78	54.17
	DistilBERT	24.60	30.17	44.70	31.72	45.70	37.18
	Dragon	65.96	59.77	62.57	50.59	68.37	54.32
	Hybrid	64.99	58.75	61.79	50.27	67.16	54.42
	Jina	62.68	58.23	60.48	48.90	65.46	52.75
	MXBAI	9.51	7.27	31.13	19.93	38.20	26.01
	SPLADE	58.96	60.10	60.64	52.13	63.42	54.02
Structure Based	G-RAG	72.40	51.21	**68.61**	46.33	74.16	50.00
	G-Retriever	12.05	5.96	47.59	25.00	46.12	27.74
Graph Pretraining	SKP	50.79	43.52	60.06	41.22	66.35	45.15
	KAMR	**73.53**	**62.50**	**68.61**	**52.98**	**75.86**	**56.54**

Table 3: Evaluation Results on CWQ Dataset.

Retriever Category	Method	Qwen3-8B				LLaMA2-7B				ChatGPT-3.5			
		Acc	Prec	Rec	F1	Acc	Prec	Rec	F1	Acc	Prec	Rec	F1
✗	LLM-only	25.99	29.88	24.42	25.49	28.23	9.99	28.23	14.70	37.88	42.19	35.63	38.63
Lexical Based	BM25	20.44	24.02	19.55	20.44	29.41	34.81	28.09	29.38	31.70	36.22	30.05	31.27
	TF-IDF	25.83	29.76	24.51	25.54	33.26	38.69	31.41	32.82	35.84	40.92	33.71	35.13
Semantic Based	BCE	32.36	36.48	29.57	30.96	36.11	41.74	33.72	35.31	40.28	44.77	37.00	38.57
	BGE	35.59	39.45	32.72	34.08	38.35	44.21	35.85	37.55	42.90	47.24	39.35	40.95
	ColBERT	36.48	40.27	33.05	34.49	37.31	42.96	34.82	36.44	42.66	47.52	39.33	40.99
	DistilBERT	19.22	23.76	18.27	19.30	27.38	32.57	26.12	29.01	30.85	36.34	29.60	32.64
	Dragon	37.25	41.22	34.12	35.53	38.70	44.80	36.23	37.93	44.45	49.22	40.91	42.57
	Hybrid	36.95	40.44	33.66	32.02	35.12	40.70	33.25	36.61	39.55	44.52	37.06	40.47
	Jina	34.00	37.92	31.16	32.50	36.16	42.34	33.98	35.64	40.94	45.82	37.59	39.25
	MXBAI	17.23	20.56	16.21	17.05	25.58	30.70	24.24	25.49	32.39	37.81	30.76	32.16
	SPLADE	34.94	38.40	31.70	33.02	37.52	43.08	34.79	36.41	43.45	47.64	39.39	41.04
Structure Based	G-RAG	29.17	33.29	26.40	27.76	34.19	39.88	31.88	33.47	31.87	36.44	30.55	33.21
	G-Retriever	27.87	31.42	25.65	26.80	38.37	43.56	35.73	37.30	33.51	39.42	31.84	34.99
Graph Pretraining	SKP	29.37	32.16	27.56	29.67	28.54	33.48	27.04	29.86	33.43	38.71	31.71	34.83
	KAMR	**38.71**	**42.40**	**34.90**	**36.43**	**40.19**	**45.40**	**36.87**	**38.93**	**45.10**	**49.59**	**41.09**	**42.70**

3.3 Evaluation on Question Answering Tasks

Tables 2 and 3 summarize question answering evaluation results on PathQuestion and CWQ datasets across three LLM backbones. We use accuracy as the evaluation metric for the single-answer dataset PathQuestion, and accuracy, precision, recall, and F1 scores for the multi-answer dataset CWQ. Overall, KAMR consistently achieves the best performance across all three backbones, indicating that the proposed alignment-oriented training and structure-aware inference yield stable gains across diverse generators and evaluation settings.

Aligning with the retrieval trends in Table 1, on PathQuestion, structure-aware retrieval can be effective when the required evidence is relatively local. G-RAG performs competitively on PQ-2H, which aligns with its design of retrieving and encoding a 1-hop subgraph as evidence. However, its performance degrades notably on PQ-3H, where the required reasoning chain extends beyond a single local neighborhood. In contrast, KAMR maintains more competitive performance on PQ-3H, suggesting that integrating anchors with connected triplets better retrieves longer compositional chains, notably facilitating the grounded generation.

The QA task is more challenging on the CWQ dataset due to mixed reasoning depths, multiple correct answer candidates, and noisier query phras-

Table 4: Ablation Study of KAMR.

Setting	Model	Accuracy
Full Model	KAMR	**75.86**
Pretraining Ablation	Without $\mathcal{L}_a$	48.79
	Without $\mathcal{L}_c$	75.10
Inference Ablation	Anchor Only	63.00
	Full Anchor Triplets	75.26
	Full Connected Triplets	74.79

ing and entity mentions. In this setting, strong semantic retrievers, e.g., Dragon, remain competitive because semantic matching is more tolerant to lexical variation and noisy mentions. Nevertheless, KAMR still achieves the best overall results across all backbones, suggesting that targeted pretraining better bridges query–triplet mismatch under noisy inputs, while the inference strategy enables broader evidence acquisition that can support multiple correct answers.

By combining partial-alignment-oriented pretraining with a structure-aware inference strategy, KAMR better balances relevance and connectivity during retrieval, yielding stable, backbone-agnostic gains on both PathQuestion and CWQ datasets. A further investigation on scalability of KAMR during the inference stage is available in Appendix E.

3.4 Ablation Study

To quantify the contribution of each design choice in KAMR, we conduct ablations on the PQ-2H dataset with ChatGPT-3.5-Turbo and report accuracy in Table 4. Each variant is constructed to directly correspond to a specific component in our methodology, allowing us to isolate its effect on end-to-end generation.

Ablating the pretraining objectives. Our pretraining jointly handles partial alignment for anchor triplets $\mathcal{L}_a$ (i.e., Eq. (4)) and connected triplets $\mathcal{L}_c$ (i.e., Eq. (5)), mirroring the two retrieval targets in Section 2.2. To assess their roles, we remove each pretraining objective individually. Excluding $\mathcal{L}_a$ leads to a severe degradation, consistent with our formulation that accurate anchor identification is the prerequisite for inducing a reliable local search region and subsequent expansion. Removing $\mathcal{L}_c$ also hurts performance, indicating that learning element-level alignment is necessary for selecting informative connected triplets, given the limited partial overlap between semantics in the query and connected triplet.

Ablating the connected triplet retrieval in inference. A key component of KAMR is the connected

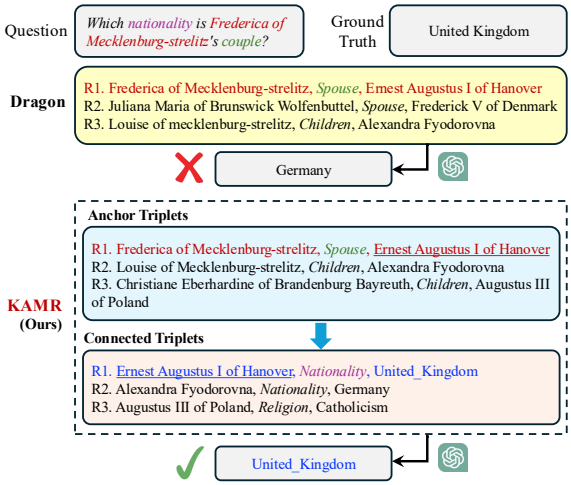

Figure 2: Case study comparison.

triplet retrieval module, which complements globally retrieved anchors by adding structurally connected triplets from their local neighborhoods. To test its necessity during inference, we remove the connected triplet retrieval module and keep only the anchor triplets, denoted "*Anchor Only*". This variant exhibits clear degradation, confirming that local expanding supplies are essential intermediate evidence, weakly aligned to the query, but required to complete multi-hop reasoning chains.

Ablating partial alignment design in inference. As we mentioned in Section 2.4, only partial triplets are used in inference. We use the following two inference variants to validate the effectiveness of such a design. First, for anchor retrieval, instead of computing similarity between the query and partial triplets, we use full triplets. This ablation is called "*Full Anchor Triplets*" in Table 4. We can observe a drop in accuracy. Second, when retrieving the connected triplets, we proposed using the elements of the triplets to calculate similarity with the query. Similarly, we use full triplets in the ablation, denoted as "*Full Connected Triplets*". Still, the performance drops. Together, these ablations show that KAMR has a redundancy-free design, offering a clearer view of the principles that drive its performance.

3.5 Qualitative Study

Beyond the quantitative results, we present two qualitative analyses to provide an intuitive understanding of how KAMR retrieves evidence and how its structure-aware search supports downstream generation.

Case study of retrieved evidence. We examine a natural-language query from the PQ-2H dataset and compare the retrieved evidence produced by

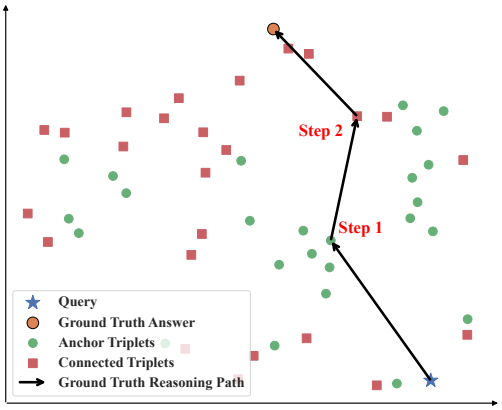

Figure 3: Visualization of how KAMR retrieves semantically distant yet structurally relevant triplets via connected triplet retrieval, enabling access to knowledge-rich intermediate facts that support reaching the ground-truth answer.

KAMR against one of the strongest baseline retrievers, i.e., Dragon. The results are shown in Figure 2. While a semantic retriever often captures the key anchor triplet (*Frederica of Mecklenburg-strelitz, Spouse, Ernest Augustus I of Hanover*), it fails to retrieve structurally necessary but weakly aligned supporting triplets such as (*Ernest Augustus I of Hanover, Nationality, United_Kingdom*), resulting in fragmented and incomplete evidence and limiting answer groundedness. In contrast, KAMR explicitly retrieves such weakly aligned yet structurally relevant facts by expanding from anchor triplets via graph connectivity. This yields a more complete and better-connected evidence set, leading to a more grounded and accurate generation.

Semantic investigation via visualization. To further isolate the role of connected triplet retrieval, we conduct a semantic-space analysis using t-SNE (Maaten and Hinton, 2008) on another set of retrieval examples in PQ-2H. Figure 3 visualizes the query, the retrieved anchor triplets $\mathcal{A}_q$, and the expanded triplets $\mathcal{C}_q$. As expected, when retrieval relies primarily on query semantics, the model tends to select anchor triplets that lie close to the query embedding. As a complement, searching connected triplets with graph structure brings in additional triplets that may be farther away semantically, but are connected through the graph and contain knowledge-rich intermediate information. These connected triplets acquired through KAMR help bridge missing hops in the reasoning chain toward the final answer, thereby improving the stability and completeness of the evidence used for generation.

4 Related Work

Multi-hop Retrieval for Generation. Multi-step generation requires evidence spanning multiple hops. Text-based RAG achieves this via iterative retrieval or query decomposition (Jiang et al., 2023; Trivedi et al., 2023; Lee et al., 2024; Verma et al., 2024), while Graph RAG retrieves structured evidence from knowledge graphs, enabling more interpretable compositional reasoning (Gao et al., 2023; Guo et al., 2023; Ma et al., 2023; Min et al., 2019; Zhang et al., 2025). Predefined graph indices, however, often rely on bespoke KGs and limit generalization. Most work instead applies pretrained LM retrievers over graph elements (He et al., 2024; Li et al., 2023, 2024; Hu et al., 2024). Although LLM-based consumption of retrieved structures has been explored (Scarselli et al., 2008), retrieval remains largely text-oriented, often ranking isolated items without enforcing coherent multi-hop structure.

Dense Retrievers for RAG. Dense retrievers link natural-language queries to external knowledge by ranking candidates via embedding-based similarity, forming the backbone of many RAG systems (Robertson et al., 2009; Salton and Buckley, 1988; Karpukhin et al., 2020; Gao et al., 2021; Ni et al., 2022). Advanced variants could further enhance semantic matching beyond exact token overlap and boost the generation quality through the RAG procedure (Chen et al., 2024; Youdao, 2023; Günther et al., 2023; Shakir et al., 2024; Formal et al., 2021; Santhanam et al., 2022; Lin et al., 2023) . While effective for plain-text RAG, these methods are tuned for unstructured text and overlook graph topology and relational dependencies needed for multi-hop reasoning.

5 Conclusion

We presented KAMR, a retriever for multi-hop fact acquisition over knowledge graphs that addresses both partial query–triplet alignment and structural composability. KAMR constructs supervision via LLM-guided triplet masking and query generation, and applies targeted contrastive pretraining with pair-level and element-level objectives to better align natural-language queries with symbolic triplets. At inference time, KAMR retrieves anchor triplets globally and expands locally to collect connected subgraph-level evidence. Experiments show that KAMR consistently improves multi-hop retrieval quality and yields stronger downstream question answering performance.

Limitations

While KAMR demonstrates strong performance in graph retrieval, it still has one key limitation. Although our pretraining strategy is broadly applicable, we implement KAMR using a conventional two-tower retriever in this study for clarity and ease of deployment. Combining our partial-alignment pretraining strategy with more advanced retrieval systems may yield more promising results by enabling deeper evidence aggregation and better balancing relevance with connectivity. Investigating more expressive retriever architectures presents a promising direction for future work.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250.

Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. 2024. Bge m3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. *arXiv preprint arXiv:2402.03216*.

Guanting Dong, Rumei Li, Sirui Wang, Yupeng Zhang, Yunsen Xian, and Weiran Xu. 2023. Bridging the kb-text gap: Leveraging structured knowledge-aware pre-training for kbqa. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, pages 3854–3859.

Thibault Formal, Benjamin Piwowarski, and Stéphane Clinchant. 2021. Splade: Sparse lexical and expansion model for first stage ranking. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2288–2292.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. Simcse: Simple contrastive learning of sentence embeddings. In *2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021*, pages 6894–6910. Association for Computational Linguistics (ACL).

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.

Michel X Goemans and David P Williamson. 1995. A general approximation technique for constrained forest problems. *SIAM Journal on Computing*, 24(2):296–317.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Michael Günther, Louis Milliken, Jonathan Geuter, Georgios Mastrapas, Bo Wang, and Han Xiao. 2023. Jina embeddings: A novel set of high-performance sentence embedding models. In *Proceedings of the 3rd Workshop for Natural Language Processing Open Source Software (NLP-OSS 2023)*, pages 8–18.

Zhicheng Guo, Sijie Cheng, Yile Wang, Peng Li, and Yang Liu. 2023. Prompt-guided retrieval augmentation for non-knowledge-intensive tasks. In *Findings of the Association for Computational Linguistics*, pages 10896–10912. Association for Computational Linguistics.

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.

Yuntong Hu, Zhihan Lei, Zhongjie Dai, Allen Zhang, Abhinav Angirekula, Zheng Zhang, and Liang Zhao. 2025. Cg-rag: Research question answering by citation graph retrieval-augmented llms. *arXiv preprint arXiv:2501.15067*.

Yuntong Hu, Zhihan Lei, Zheng Zhang, Bo Pan, Chen Ling, and Liang Zhao. 2024. Grag: Graph retrieval-augmented generation. *arXiv preprint arXiv:2405.16506*.

Zhengbao Jiang, Frank F Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. Active retrieval augmented generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7969–7992.

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

Myeonghwa Lee, Seonho An, and Min-Soo Kim. 2024. Planrag: A plan-then-retrieval augmented generation for generative large language models as decision makers. *arXiv preprint arXiv:2406.12430*.

Mufei Li, Siqi Miao, and Pan Li. 2024. Simple is effective: The roles of graphs and large language models in knowledge-graph-based retrieval-augmented generation. *arXiv preprint arXiv:2410.20724*.

Shiyang Li, Yifan Gao, Haoming Jiang, Qingyu Yin, Zheng Li, Xifeng Yan, Chao Zhang, and Bing Yin. 2023. Graph reasoning for question answering with triplet retrieval. *arXiv preprint arXiv:2305.18742*.

Sheng-Chieh Lin, Akari Asai, Minghan Li, Barlas Oguz, Jimmy Lin, Yashar Mehdad, Wen-tau Yih, and Xilun Chen. 2023. How to train your dragon: Diverse augmentation towards generalizable dense retrieval. *arXiv preprint arXiv:2302.07452*.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Xinbei Ma, Yeyun Gong, Pengcheng He, Hai Zhao, and Nan Duan. 2023. Query rewriting in retrieval-augmented large language models. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 5303–5315. Association for Computational Linguistics.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605.

Sewon Min, Danqi Chen, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2019. Knowledge guided text retrieval and reading for open domain question answering. *arXiv preprint arXiv:1911.03868*.

Jianmo Ni, Gustavo Hernandez Abrego, Noah Constant, Ji Ma, Keith Hall, Daniel Cer, and Yinfei Yang. 2022. Sentence-t5: Scalable sentence encoders from pretrained text-to-text models. In *Findings of the association for computational linguistics: ACL 2022*, pages 1864–1874.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*.

Stephen Robertson, Hugo Zaragoza, and 1 others. 2009. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389.

Gerard Salton and Christopher Buckley. 1988. Term-weighting approaches in automatic text retrieval. *Information processing & management*, 24(5):513–523.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *ArXiv*, abs/1910.01108.

Keshav Santhanam, Omar Khattab, Jon Saad-Falcon, Christopher Potts, and Matei Zaharia. 2022. Colbertv2: Effective and efficient retrieval via lightweight late interaction. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3715–3734.

Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. 2008. The graph neural network model. *IEEE transactions on neural networks*, 20(1):61–80.

Dikshant Shahi. 2015. *Apache Solr: a practical approach to enterprise search*. Springer.

A Shakir, D Koenig, J Lipp, and S Lee. 2024. Boost your search with the crispy mixedbread rerank models.

Karen Sparck Jones. 1972. A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation*, 28(1):11–21.

Alon Talmor and Jonathan Berant. 2018. The web as a knowledge-base for answering complex questions. *arXiv preprint arXiv:1803.06643*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, and 1 others. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2023. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (ACL)*.

Prakhar Verma, Sukruta Prakash Midigeshi, Gaurav Sinha, Arno Solin, Nagarajan Natarajan, and Amit Sharma. 2024. Plan* rag: Efficient test-time planning for retrieval augmented generation. *arXiv preprint arXiv:2410.20753*.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Wen-tau Yih, Matthew Richardson, Christopher Meek, Ming-Wei Chang, and Jina Suh. 2016. The value of semantic parse labeling for knowledge base question answering. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 201–206.

I NetEase Youdao. 2023. Bcembedding: Bilingual and crosslingual embedding for rag. *Bcembedding: Bilingual and crosslingual embedding for rag*.

Qinggang Zhang, Shengyuan Chen, Yuanchen Bei, Zheng Yuan, Huachi Zhou, Zijin Hong, Junnan Dong, Hao Chen, Yi Chang, and Xiao Huang. 2025. A survey of graph retrieval-augmented generation for customized large language models. *arXiv preprint arXiv:2501.13958*.

Mantong Zhou, Minlie Huang, and Xiaoyan Zhu. 2018. An interpretable reasoning network for multi-relation question answering. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2010–2022.

A LLM Usage Statement

Large language models (LLMs) are a central object of study in this paper. Our research investigates how LLMs can be better supported by multi-hop retrieval for graph-based augmentation and generation, and we evaluate their behavior and outcomes within this retrieval-augmented framework. In this sense, LLMs are not merely auxiliary tools but an integral component of the problem setting and experimental analysis.

Separately, we used GPT-5 only for manuscript editing. Its use was limited to surface-level refinement, including improving phrasing, correcting grammar, and enhancing readability. All scientific contributions, including the problem formulation, methodology, experimental design, results, and interpretations, were developed by the authors.

All LLM-involved outputs were reviewed for accuracy and appropriateness. The authors take full responsibility for the validity and integrity of the final content.

B More information of KAMR

B.1 Prompt for Pretrainng Dataset Construction

The prompt for data construction from triplets is listed in Figure 4.

Prompt for Pretraining Dataset Construction

You are given a triple from a knowledge graph in the form of (head, relation, tail), but one element is missing, shown as '??'.

Your job is to generate a single natural language question based on the incomplete triple, such that the missing part is what the question is asking for.

The question should be a complete sentence ending with a question mark.

Do NOT generate multiple questions. Stop after the first question. Do NOT include the masked answer or the triple in your output.

Triple: {triplet}
Masked Answer: {answer}
Output:

Figure 4: Prompt for Pretraining Dataset Construction.

B.2 Algorithm of KAMR

Algorithm 1 provides an intuitive overview of the full inference pipeline of KAMR, facilitating clear understanding and reliable reproducibility.

C Experiment Details

C.1 Baselines

We include the following graph-retrieval baselines for comparison:

BM25 (Robertson et al., 2009) is a classic lexical retriever based on the probabilistic relevance framework. It ranks documents by aggregating exact term matches between the query and document, with a saturated term-frequency component and explicit document-length normalization (controlled by hyperparameters such as k_1 and b), making it a strong sparse baseline for keyword-style retrieval.

TF-IDF (Sparck Jones, 1972) is a sparse vector-space retriever that represents each document (and query) with term weights computed from term frequency (TF) and inverse document frequency (IDF). It retrieves documents by comparing these weighted bag-of-words vectors (commonly via cosine similarity), favoring terms that are frequent in a document but rare across the corpus, and serving as a widely used baseline for lexical matching.

G-Retriever (He et al., 2024) is a retrieval-augmented generation framework designed for question answering over textual graphs. It retrieves relevant nodes and edges based on semantic similarity and constructs subgraphs using the Prize-Collecting Steiner Tree (PCST) (Goemans and Williamson, 1995) algorithm to form concise, query-relevant subgraphs for generation .

G-RAG (Hu et al., 2024) is a graph retrieval-augmented generation method that enhances LLMs by retrieving and integrating textual subgraphs. It represents subgraphs as pooled embeddings of k-hop ego-graphs and retrieves them to incorporate both textual and topological information through dual prompting, improving performance on multi-hop reasoning tasks .

Hybrid (Li et al., 2023) is a hybrid retrieval model that combines sparse retrieval (BM25) and dense retrieval (DPR) for coarse retrieval, followed by reranking with a cross-encoder to improve retrieval performance.

SKP (Dong et al., 2023) leverages traditional approaches like contrastive learning and masked language prediction on graphs to obtain a more graph-concentrated encoder for retrieval, enhancing the

Algorithm 1: Two-Stage Multi-hop Retrieval at Inference Time

Input: Query q; knowledge graph $\mathcal{G}$; encoders $\text{ENC}_q(\cdot), \text{ENC}_u(\cdot)$; hop radius g; budgets M, N

Output: Multi-hop evidence set $\mathcal{T}_q$

Encode query: $\mathbf{q} \leftarrow \text{ENC}_q(q)$

Stage 1: Anchor triplet retrieval

foreach $\tau = (h, r, t) \in \mathcal{T}_\mathcal{G}$ **do**

  $\mathcal{U}(\tau) \leftarrow \{(h, r), (r, t), (h, t)\}$

  $s_A(q, \tau) \leftarrow \max_{u=(e^1,e^2)\in\mathcal{U}(\tau)} \texttt{sim}\big(\mathbf{q}, \text{ENC}_u(e^1 \oplus e^2)\big)$

$\mathcal{A}_q \leftarrow \texttt{Top-}M\big(\{(\tau, s_A(q, \tau)) : \tau \in \mathcal{T}_\mathcal{G}\}\big)$

Stage 2: Connected triplet retrieval

$\mathcal{C}_q \leftarrow \emptyset$

foreach $\tau_i \in \mathcal{A}_q$ **do**

  $X(\tau_i) \leftarrow \{x \in \texttt{Neighbors}(\tau_i, g) : x$ satisfies Definition 3 w.r.t. $\tau_i\}$

  foreach $x \in X(\tau_i)$ **do**

    $\mathcal{E}_x \leftarrow \text{Elem}(x) - \text{Elem}(\tau_i)$

    $s_C(q, x) \leftarrow \max_{e\in\mathcal{E}_x} \texttt{sim}\big(\mathbf{q}, \text{ENC}_u(e)\big)$

  $\mathcal{C}_q(\tau_i) \leftarrow \texttt{Top-}N\big(\{(x, s_C(q, x)) : x \in X(\tau_i)\}\big)$

  $\mathcal{C}_q \leftarrow \mathcal{C}_q \cup \mathcal{C}_q(\tau_i)$

Compose evidence:

$\mathcal{T}_q \leftarrow \mathcal{A}_q \cup \mathcal{C}_q$

return $\mathcal{T}_q$

model's ability to represent complex subgraphs.

BGE (Chen et al., 2024) is a versatile embedding-based retrieval model that supports multiple languages and tasks. It leverages dense, sparse, and multi-vector modalities to offer strong generalization across domains and text granularities.

BCE (BCEmbedding) (Youdao, 2023) is a bilingual/cross-lingual embedding framework optimized for efficient first-stage retrieval and (optional) reranking. It is designed to work well when dealing with mixed-language content, providing embeddings that capture both semantic and cross-language signals.

Jina Embeddings (Günther et al., 2023) are embedding models provided by Jina that aim for high retrieval performance through rich sentence/text representations. They focus on scalability, multilingual support, and ease of integration into existing retrieval pipelines.

MXBAI (Shakir et al., 2024) is a retrieval/embedding family that emphasizes high performance on embedding benchmarks (e.g. sentence embeddings) and aims to balance between embedding quality and computational efficiency. It supports downstream tasks such as semantic search and retrieval.

SPLADE (Formal et al., 2021) is a sparse representation model that transforms text into high-dimensional sparse vectors, preserving interpretability and allowing efficient matching with traditional retrieval indexing (e.g., inverted indices) while still capturing semantic similarity beyond keywords.

ColBERT (Santhanam et al., 2022) is a late-interaction retrieval model which encodes queries and documents/token sequences into token-level embeddings. It retains fine-grained interaction at search time, but optimizes memory/storage and inference through compression and efficient matching (e.g. MaxSim), giving strong accuracy across both in-domain and out-of-domain benchmarks.

DistilBERT (Sanh et al., 2019) is a distilled, lighter-weight version of BERT. It has fewer layers and parameters, enabling faster inference and lower resource use, and is often used as a baseline encoder for retrieval tasks where speed and efficiency matter.

C.2 Implementation

All experiments were conducted on four NVIDIA A6000 GPUs with CUDA version 12.0, running on an Ubuntu 20.04.6 LTS server. Pretraining is performed for 5 epochs using AdamW (Loshchilov and Hutter, 2017) with a batch size of 512 and a learning rate of 2×10^{-5}.

For pretraining dataset construction, we derive triplets from Freebase (Bollacker et al., 2008), restricting them to entities linked to the CWQ and PathQuestion datasets while keeping the synthetic corpus independent of retrieval evaluation and downstream question answering to prevent data leakage. We employ LLaMA-3.1-8B-Instruct (Grattafiori et al., 2024) during this process, yielding 840,875 synthesized natural-language queries, which, together with their corresponding partial triplets, form the pretraining dataset. For methods that involve retriever pretraining (SKP and KAMR), pretraining is conducted on the same synthetic dataset.

During inference, 1-hop neighborhood for each anchor triplet is used as search space of expanding triplets. $\mathcal{A}_q$ and $\mathcal{C}_q$ are each set to size 25 by choosing $M = 25$ and $N = 1$, yielding the final retrieved set $\mathcal{T}_q$ with $|\mathcal{T}_q| = 50$. Baseline retrievers are required to return the top 50 triplets to ensure a comparable retrieval budget across all methods. To ensure a fair comparison, KAMR and all baseline

Table 5: Evaluation Results on WebQSP Dataset (ChatGPT).

Retriever Category	Methods	ChatGPT			
		Acc	Prec	Rec	F1
✗	LLM-only	46.62	65.97	39.48	49.40
Lexical Based	BM25	41.84	56.39	33.47	37.23
	TF-IDF	47.42	60.75	35.88	40.04
Semantic Based	BCE	58.49	68.92	42.18	46.76
	BGE	59.87	68.49	42.46	46.91
	ColBERT	66.95	77.70	48.01	53.07
	DistilBERT	39.14	57.30	34.30	42.67
	Dragon	63.71	73.25	44.97	49.82
	Hybrid	50.38	64.25	37.52	47.37
	Jina	58.75	68.43	42.03	46.63
	MXBAI	40.25	57.62	34.69	38.45
	SPLADE	59.12	68.18	41.46	45.96
Structure Based	G-RAG	50.53	67.69	40.64	45.24
	G-Retriever	52.82	68.92	42.68	47.17
Graph Pretraining	SKP	43.86	60.44	35.80	39.89
	KAMR	55.59	68.49	41.54	46.11

retrievers depending on external encoders use the same strong text encoder, i.e., Dragon (Lin et al., 2023). We also set the same fixed seed (42) to eliminate any randomness and ensure decent reproducibility.

All retrievers are evaluated under a zero-shot setting on the question-answering datasets without any supervised customization on the task.

D Experiments on One-hop Question Answering

Although KAMR is designed to tackle multi-hop retrieval for question answering, we still conduct experiments to valid whether it can be applied to the single-hop retrieval for question answering.

Table 5 presents the experiments on WebQSP (Yih et al., 2016), a question answering dataset primarily only desiring individual triplets for answering the question. The results indicate that, although KAMR is tailored for multi-hop retrieval, it still achieves comparable performance on single-hop question answering by reliably retrieving the necessary evidence, which is enabled by its anchor triplet retrieval mechanism.

E Scalability in Inference Depth

KAMR is implemented as a two-stage inference procedure: it first retrieves anchor triplets and then expands from these anchors to collect connected triplets. This design is naturally extensible in inference depth: the two-stage procedure can be extended to three or more stages via iterative neighborhood expansion. To evaluate this hypothesis, we conduct an additional study on PathQuestion (PQ) with ChatGPT by adding a third stage: we further expand from the connected triplets retrieved in the second stage to obtain another set of connected triplets.

To ensure a fair comparison, we use the same overall retrieval budget (50) and allocate it across stages by retrieving 17 anchor triplets in the first stage, 17 connected triplets in the second stage, and 16 connected triplets in the third stage. With this budget allocation, the third stage consistently improves QA accuracy, yielding gains on both splits (2-hop: **75.86** → **78.25**; 3-hop: **56.54** → **59.81**).

However, with a fixed total budget, increasing the number of expansion stages necessarily reduces the number of anchor triplets retained, which may hurt performance because connected-triplet retrieval strongly depends on anchor quality. We leave a principled budget allocation strategy for multi-stage retrieval as future work.

F Potential Risk

Despite the strong performance of KAMR, its retrieved evidence may still encode or amplify underlying biases. Uncritical use that treats retrieved results as factual without contextual validation can lead to societal and ethical concerns. We therefore encourage users of KAMR to evaluate the retrieved content carefully within their application setting and to apply appropriate safeguards to reduce potential risks.