Distill-SynthKG: Distilling Knowledge Graph Synthesis Workflow for Improved Coverage and Efficiency

Anonymous ACL submission

Abstract

Knowledge graphs (KGs) generated by large language models (LLMs) are becoming increasingly valuable for Retrieval-Augmented Generation (RAG). However, existing KG extraction methods predominantly rely on prompt-based approaches, which are inefficient for processing large-scale corpora and suffer from information loss as document length increases. Additionally, methods and datasets for evaluating ontology-free KG construction are lacking. To address these shortcomings, we propose SynthKG, a multi-step, document-level ontology-free KG synthesis workflow. By further fine-tuning a smaller LLM on synthesized document-KG pairs, we streamline the multi-step process into a single-step KG generation approach called Distill-SynthKG. Furthermore, we re-purpose existing question-answering datasets to establish KG evaluation datasets and introduce new evaluation metrics. Using KGs produced by Distill-SynthKG, we also design a novel graph-based retrieval framework for RAG. Experimental results demonstrate that Distill-SynthKG not only surpasses all baseline models in KG quality (including models up to eight times larger) but also consistently excels in retrieval and question-answering tasks. Additionally, our proposed graph retrieval framework outperforms all KG-retrieval methods across multiple benchmark datasets. We make SynthKG and Distill-SynthKG publicly available.

1 Introduction

Retrieval Augmented Generation (RAG) has gained widespread application for effectively connecting large language models (LLMs) with external knowledge sources. Recently, Knowledge Graph (KG) augmented RAG methods have demonstrated strong potential, offering several advantages such as effective corpus-level information summarization (Edge et al., 2024), improved reasoning capabilities (Gutiérrez et al., 2024; Li et al., 2024), and accurate modeling of historical customer issue resolutions for QA (Xu et al., 2024).

Recent works (Edge et al., 2024; Gutiérrez et al., 2024) have begun exploring the use of LLMs to automate the construction of KGs, which then serve as knowledge sources for specific tasks such as question answering or building intelligent agentic frameworks. However, these existing approaches have several limitations. First, they rely on simple zero-shot or few-shot in-context learning methods to construct knowledge graphs in a single step using LLMs like GPT-4o (OpenAI, 2024). Consequently, such approaches can incur significant inference costs when applied across large corpora due to the need for many commercial API calls. These methods also lack a rigorous and reliable design specifically tailored for KG construction. Having LLMs process entire documents, particularly long texts, has been shown to potentially lead to issues such as information loss (Edge et al., 2024). Second, there is a lack of existing datasets or evaluation methods to effectively evaluate document-level ontology-free KGs. This absence makes it difficult to identify whether errors in RAG systems stem from issues in specific reasoning components or from poor-quality KGs that propagate errors throughout the system.

To address these limitations, we introduce SynthKG, a novel LLM-based KG construction workflow. We further distill this workflow into a smaller LLM named Distill-SynthKG, which enables efficient, one-step generation of high-quality document-level KGs. In SynthKG, we begin by splitting the input document into manageable, semantically complete text chunks. Each chunk is then processed through a decontextualization step where entity disambiguation occurs based on the previous context, making each chunk an independent, self-contained unit. We then prompt the LLM to extract entities, relations, and relevant propositions from each text chunk, which are combined to form the final KG. Finally, we fine-tune our smaller Distill-SynthKG LLM on the KGs produced by SynthKG, enabling it to generate the KG for a given document in a single inference step.

Additionally, we propose a method for constructing an evaluation dataset for document-level ontology-free KGs, along with a corresponding KG evaluation framework. Specifically, we re-purpose existing multihop QA datasets by converting questions and answers into ground truth relation triplets, where the answer appears as either the head, tail, or predicate in a triplet. Using these ground truth triplets for each document, we introduce semantic similarity and keyword-based metrics to assess the coverage of triplets from a KG.

Finally, we present a new graph-based retrieval framework based on the KGs generated by Distill-SynthKG. We design a progressive retrieval method that begins with proposition retrieval, leveraging the graph structure to retrieve related triplets, propositions, and text chunks relevant to the input query. Our proposed retriever outperforms state-of-the-art retrieval methods in both retrieval accuracy and question-answering accuracy, showing improvements across three multihop QA datasets: MuSiQue (Trivedi et al., 2022), 2WikiMultiHopQA (Ho et al., 2020), and HotpotQA (Yang et al., 2018). Furthermore, our KG coverage evaluation framework correlates strongly with both QA and retrieval performance, demonstrating its effectiveness in evaluating document-level KG coverage.

In summary, our contributions are as follows: (1) We introduce SynthKG, a novel LLM-based workflow that generates high-quality, high-coverage document-level ontology-free KGs. (2) We train Distill-SynthKG, which leverage SynthKG to synthesize training data and fine-tune a much smaller LLM. This simplifies the multi-step process into a single inference step, significantly improving efficiency. (3) We propose new KG evaluation datasets by re-purposing existing multi-hop QA datasets and introducing new evaluation metrics. (4) We introduce a novel graph-based retrieval method that leverages KGs generated by Distill-SynthKG. (5) Our experiments across multiple datasets demonstrate that Distill-SynthKG not only produces KGs of higher quality than all baselines—including models up to eight times larger—but also consistently outperforms them in retrieval and question-answering tasks. Furthermore, the proposed graph-based retrieval framework surpasses all baseline KG-based retrieval methods.

2 Related Work

Recently, there has been a growing interest in using Knowledge Graphs (KG) for different Retrieval-Augmented Generation (RAG) applications. For instance, GraphRAG (Edge et al., 2024) shows the advantages of using KGs over a text corpus for answering global queries that require summarizing information from multiple documents. HippoRAG (Gutiérrez et al., 2024) demonstrates that applying personalized PageRank algorithms on LLM-derived KG can enhance retrieval accuracy for complex multi-hop reasoning questions. GraphReader (Li et al., 2024) shows how KGs can enable LLM agents to plan and reason in a long context to answer complex questions. These approaches focus on maximizing KG utility.

All the above work, along with many others such as Chia et al. (2022); Trajanoska et al. (2023); Chen and Bertozzi (2023); Kai Zhang (2023); Nayak and Timmapathini (2023); Mihindukulasooriya et al. (2023); Zhu et al. (2024); Jiao et al. (2023); Khorashadizadeh et al. (2023); Han et al. (2024); Yao et al. (2024); Bi et al. (2024); Ding et al. (2024); Sanmartin (2024); Sun et al. (2024); Yao et al. (2023); Chase (2022) have used LLM prompting to build KGs or extract semantic relation triplets from text. However, all prior works have overlooked improving the efficiency of ontology-free KG construction. We are the first to develop a specialized LLM for KG construction, enhancing efficiency by shifting from large models to smaller, more efficient models without sacrificing performance.

3 Distill-SynthKG

We present Distill-SynthKG, a framework for fine-tuning LLMs by distilling the multi-step KG synthesis process (SynthKG) into a streamlined, single-step approach. This allows for the direct generation of KGs from documents using smaller-scale LLMs. Specifically, we first apply SynthKG to generate KGs from documents using a larger LLM. We then distill this process by training a smaller LLM on the resulting document-KG pairs, producing the distilled model, Distill-SynthKG.

3.1 SynthKG

SynthKG consists of two main steps: (1) document chunking and decontextualization, followed by (2) entity, relation and proposition extraction. These steps ensure high coverage of extracted entities and relations while minimizing information loss. We

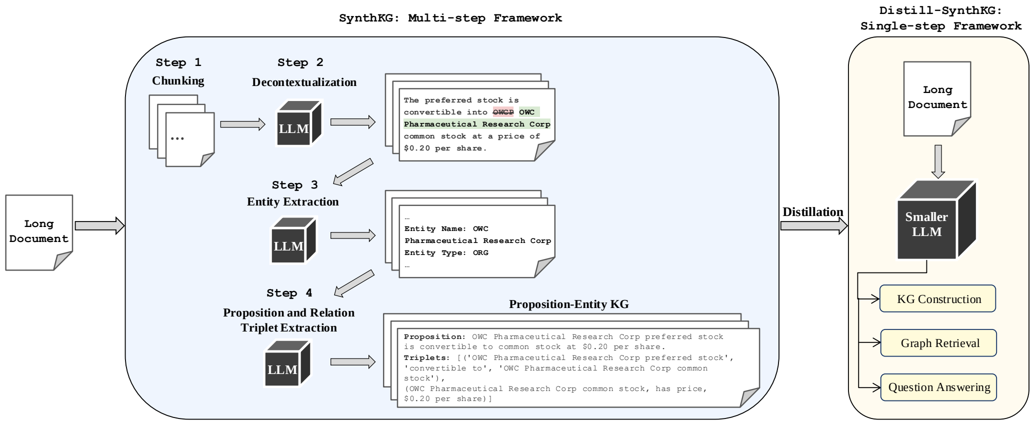

SynthKG: Multi-step Framework
Step 1
Chunking
…
Step 2
Decontextualization
LLM
The preferred stock is convertible into ~~OWCP~~ OWC Pharmaceutical Research Corp common stock at a price of $0.20 per share.
Step 3
Entity Extraction
LLM
…
Entity Name: OWC Pharmaceutical Research Corp
Entity Type: ORG
…
Step 4
Proposition and Relation Triplet Extraction
LLM
Proposition-Entity KG
Proposition: OWC Pharmaceutical Research Corp preferred stock is convertible to common stock at $0.20 per share.
Triplets: [('OWC Pharmaceutical Research Corp preferred stock', 'convertible to', 'OWC Pharmaceutical Research Corp common stock'), (OWC Pharmaceutical Research Corp common stock, has price, $0.20 per share)]
Long Document
Distillation
Distill-SynthKG: Single-step Framework
Long Document
Smaller LLM
KG Construction
Graph Retrieval
Question Answering

Figure 1: Our SynthKG data synthesis method (left) generates high-coverage, ontology-free, document-level KGs. We distill this synthetic data into Distill-SynthKG (right), which is applied to multiple downstream applications. Long document refers to multi-paragraph documents in our pipeline.

present an overview of SynthKG in Figure 1. Additionally, we provide details of all LLM prompts utilized in this process in Appendix B.

3.1.1 Document Chunking and Decontextualization

Directly inputting long texts into an LLM has been shown to result in information loss (Edge et al., 2024). To mitigate this risks, we first split each input document into smaller, more manageable chunks before processing them with the LLM in subsequent steps. This chunking is done along sentence boundaries, without overlap, to preserve semantic coherence and avoid redundancy.

However, processing each chunk in isolation can lead to a loss of prior context. For example, if "John Doe" appears in one chunk and "John" in another, we might lose track of who "John" refers to. To prevent this, we apply a "decontextualization" step, where we prompt the LLM to rewrite each chunk, replacing all entity mentions with their most informative form based on the context of the preceding chunk. For example, if "John Doe" is introduced in a previous chunk, subsequent mentions of "John D." "John," or related pronouns are replaced with "John Doe." This not only preserves context but also prevents the same entity from being represented in different forms, which could lead to redundancy, discontinuous KG paths, and reduced accuracy at inference time. The first chunk of a document is not decontextualized, as chunking does not lead to context loss in this case. We provide an example of a decontextualized chunk in Figure 9.

To verify that the preceding chunk is sufficient for decontextualization, we calculate the average chunk distance for the same entity within each document in our generated dataset of 100K samples (details of this dataset are described in Section 6.1). Specifically, we measure the distance between the first occurrence of each entity and its subsequent mentions. The overall average chunk distance per entity is 0.9, indicating that, on average, entities are mentioned again within less than one chunk after their first mention. This suggests that using only the preceding chunk is sufficient and that no significant number of entities remain unresolved due to the chunk-based decontextualization process.

One potential downside of prompting the LLM to rewrite chunks is that the rewritten version may deviate from the original, potentially introducing information loss or hallucination. To mitigate this, we use ROUGE scores to compare the original and decontextualized chunks, filtering out those that exhibit significant deviations. Detailed experimental settings are provided in Section 6.1. To further assess the accuracy of our decontextualization process, we manually annotate 75 randomly selected decontextualized chunks. Three authors each annotate 25 chunks, with access to both the original chunk and the full document. They evaluate whether modifications are made, whether those modifications are correct, and whether any information is lost.

Among the 75 annotated chunks, we identify a total of 593 edits, with only six containing incorrect modifications and four showing information loss. These results indicate that the decontextualization process generally produces high-quality, self-contained text. Moreover, our annotations reveal that most modifications enhance specificity—for example, replacing a general term like "scientists" with "Darwinian scientists." This suggests that the rewritten chunks are typically self-contained and comprehensible on their own.

3.1.2 Entity and Relation Extraction

Similar to Edge et al. (2024) and Gutiérrez et al. (2024), we first prompt the LLM to extract all entities and their corresponding types from each text chunk, as shown in Step 3 of Figure 1. Then, we prompt the LLM again to generate all propositions and corresponding relation triplets based on the text chunk and previously extracted entities. Each relation is represented by quadruplets consisting of a **source** entity, **predicate**, **target** entity, and a **proposition** (see Figure 1 for examples). The proposition is a sentence that describes the semantic relation between the source and target entities, encapsulating all key details of that relation.

We extend traditional KG triples by adding a proposition component, which functions as an intermediate chain of thought (Wei et al., 2022) enabling the LLM to first articulate the relevant context coherently before extracting the corresponding triplets. This approach therefore better leverages contextual information. Additionally, the proposition acts as a fine-grained, self-contained retrieval unit, which facilitates the construction of KG-based retrieval indices. Beyond triplets and text chunks, our final KG incorporates these clear, independent propositions. For example, the proposition "*OWC Pharmaceutical Research Corp preferred stock is convertible to common stock at $0.20 per share.*" provides important contextual details, such as the "*conversion price $0.20 per share*," and also serves as a precise, indexable unit.

3.2 Distilling SynthKG

While the detailed, chunk-by-chunk approach in SynthKG enables the generation of high-quality KGs using LLMs, it introduces efficiency challenges. Each time we construct a KG from a document, multiple LLM calls are required, leading to high computational or API costs and limiting the scalability of KG construction. For example, processing a 1000-word document requires 12 LLM inference calls: the document is split into 4 chunks, and each chunk involves 3 calls for decontextualization, entity extraction, and relation extraction.

To mitigate this, we distill the entire multi-step SynthKG framework into a single-step framework for a smaller LLM, as shown in Figure 1, by leveraging the document–KG pairs generated during the original SynthKG process. Specifically, we fine-tune a smaller LLM so that it directly accepts the entire document as input, uses its smaller parameter size advantage, and produces the same knowledge graph (i.e., a set of quadruples) as SynthKG in one inference step. We hypothesize that the high-quality document-KG pairs generated by SynthKG can be effectively used to train smaller LLMs, helping to mitigate the information loss that commonly occurs when processing entire documents without such training. However, there is currently no large-scale dataset available for this type of training, making SynthKG essential for creating the data necessary to enable such model distillation.

4 KG Coverage Evaluation

Evaluating the quality of the extracted KG is essential, as it directly impacts its downstream applications. However, there is a lack of document-level data for KG evaluation. Although DocRED (Yao et al., 2019) is one existing dataset, it is limited to just 96 relations, making it less suitable for KGs used in retrieval tasks, which often rely on an open ontology and include diverse relations. To address this gap, we propose generating proxy ground truth relation triplets from multihop QA datasets and introduce metrics for evaluating the coverage of these proxy triplets in the extracted KG.

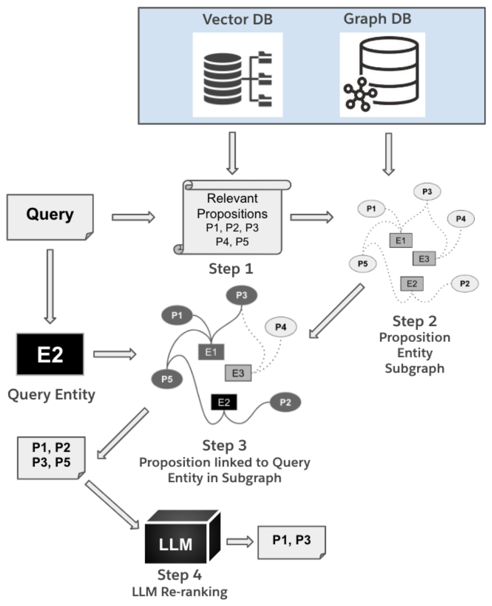
Proxy Triplets Generation We use GPT-4o to generate triplets from QA pairs. Given that multihop QA requires reasoning over multiple facts, we generate one triplet for each individual fact. In datasets where these facts are present as subquestion-answer pairs, we create triplets using these pairs while ensuring that the answer is used as the head, relation, or tail in the triplet. In cases where facts or subquestions are unavailable, we use GPT-4o to first generate the required subquestions before subsequently generating the corresponding triplets. The prompts used for generating the triplets and decomposed questions, along with relevant examples, are provided in Appendix C.1. In human evaluations, we found a high degree of validity in triplets extracted by GPT-4o with this approach; see Appendix C.2 for details.

KG Coverage Evaluation Metrics Existing KG evaluation metrics typically depend on exact match or F1 score at the text level, given that relations are derived from a predefined set. However, this approach is inefficient for ontology-free KGs, where entities and relations are not constrained. To address this, we use semantic matching to align the extracted triplets with the ground truth triplets, and propose three complementary metrics: semantic scores, triplet coverage, and F1 scores. Note that the quality of an extracted KG is evaluated based on its coverage of a given set of ground truth triplets. Therefore, these three metrics are not aimed at measuring the graph's comprehensiveness but rather at verifying whether the important triplets—those critical for answering questions—are included in the graph. As a proxy for comprehensiveness, we additionally compare the number of extracted triplets. Our three proposed metrics are defined as follows:

- *Semantic score*: We calculate the cosine similarity between the vector representation of each ground truth triplet and the triplets in the knowledge graph, taking the highest similarity score as the semantic score for that ground truth triplet. A higher semantic score indicates a closer match between the ground truth and the extracted graph.
- *Triplet Coverage*: If the semantic score for a ground truth triplet exceeds a cutoff threshold, it is marked as covered (coverage = 1); otherwise, the triplet is not covered (coverage = 0).
- *F1 score*: We use the semantic score to identify the triplet from the knowledge graph that most closely matches the ground truth triplet. Then, we compute the F1 score by comparing the text of the extracted and ground truth triplets.

5 Proposition-Entity Graph Retriever

We introduce a novel retriever based on Proposition-Entity Graph (Figure 2), designed for queries requiring multi-hop reasoning. Given a question, we first retrieve the top-M most relevant propositions from the knowledge graph using embedding similarity, narrowing the search space to a smaller subset of relevant information. In step 2, we construct a sub-graph consisting of these propositions and their linked entities, capturing the relations among the retrieved propositions. In step 3, we traverse the sub-graph starting from the entities mentioned in the question, selecting only propositions within their N-hop neighborhood. This filters out semantically similar but irrelevant propositions, ensuring that only those logically connected to the question entities are retained. We then include text chunks corresponding to the selected propositions within N-hop distance



Vector DB
Graph DB
Query
Relevant Propositions P1, P2, P3 P4, P5
Step 1
P1
P2
P3
P4
P5
E1
E2
E3
Step 2 Proposition Entity Subgraph
E2
Query Entity
Step 3 Proposition linked to Query Entity in Subgraph
P1, P2 P3, P5
LLM
P1, P3
Step 4 LLM Re-ranking

Figure 2: Our Proposition-Entity Graph Retriever for multi-hop reasoning retrieves semantically similar propositions, uses graph traversal to select those connected though query entities, and then re-rank selected propositions using LLMs.

to question entities, ranked by their embedding similarity to the query, until the top-K chunks are selected. We call this approach **Graph** Retriever.

Additionally, as shown in step 4 of Figure 2, we prompt an LLM to identify the necessary propositions to answer the question from those retrieved in the Graph Retriever process, effectively using LLM reasoning capabilities to re-rank the selected propositions. Following this LLM-based re-ranking, we include the chunks corresponding to the LLM-identified propositions first, and then fall back to the Graph Retriever to select additional chunks until the top-K chunks are selected. We refer to this combined approach as **Graph+LLM** in Section 6.

6 Experiments

6.1 KG Synthesis and Distillation Settings

SynthKG Dataset and Model: We use *Llama-3.1-70b-Instruct* (AI@Meta, 2024) on 100K documents from *IndustryCorpus* (by BAAI) for synthesizing KGs. We sample an equal number of documents from the following categories: politics, news, medicine, literature, finance, film & TV, computer science, automotive, technology, and education. We use the SentenceSplitter from the Llama-Index (Liu, 2022) framework to split documents into chunks, setting the chunk size to 256 tokens and chunk overlap to 0 tokens. We apply a

KG Source	MuSiQue				2wiki				HotpotQA			
	Triplets	Semantic	Coverage	F1	Triplets	Semantic	Coverage	F1	Triplets	Semantic	Coverage	F1
Llama-3-8b	93855	0.8111	32.09	0.51	41384	0.8281	43.39	0.56	76906	0.8343	41.79	0.58
SynthKG-8b	125197	0.8341	38.84	0.55	56178	0.8275	44.56	0.54	108031	0.8448	47.72	0.60
Llama-3-70b	102119	0.8346	40.34	0.56	46100	0.8475	54.10	0.58	82948	0.8440	47.20	0.61
SynthKG-70b	**140527**	**0.8559**	**47.18**	**0.59**	**71305**	**0.8778**	**63.30**	**0.61**	**124460**	0.8633	54.54	0.63
D-SynthKG-8b	139376	0.8546	46.90	**0.59**	68800	0.8693	58.27	0.59	123458	**0.8693**	**55.26**	**0.64**

Table 1: KG coverage performance. The best scores are **bolded**, and the second-best scores are underlined.

filtering criterion based on the ROUGE-1 F1 score (Lin, 2004), setting a threshold of 0.70 to minimize the risk of hallucinations from decontextualization. We perform the KG synthesis using VLLM (Kwon et al., 2023) on 160 Intel® Gaudi 2 AI accelerators in the Intel® Tiber™ AI Cloud. Our 100K generated document-KG pairs will be publicly released.

SynthKG Distillation: We train *Meta-Llama-3-8b-Instruct* (AI@Meta, 2024) on 30K synthesized documents to directly generate corresponding KGs for entire input document using 8 Intel® Gaudi 2 AI accelerators in the Intel® Tiber™ AI Cloud. We employ a learning rate of 5e-5, a batch size of 32, and train for one epoch. We name our model Distill-SynthKG and refer to it subsequently as **D-SynthKG-8b** .

6.2 Evaluation Settings

Datasets We evaluate KGs extracted by D-SynthKG-8b on KG coverage, text chunk retrieval and QA tasks using 3 multi-hop reasoning datasets: MuSiQue, 2WikiMultiHopQA (2wiki) and HotpotQA. We follow the settings of HippoRAG (Gutiérrez et al., 2024) and use the same 1000 questions and candidate passages, including both supporting and distractor passages. For KG coverage evaluation, we generate proxy ground-truth triplets using GPT-4o.

Baselines Across all tasks, we compare KGs extracted by our D-SynthKG-8b against those extracted by two baseline models: Llama-3-8b and Llama-3-70b[1]. Additionally, we run the full multi-step SynthKG pipeline with both Llama-3-8b (SynthKG-8b) and Llama-3-70b (SynthKG-70b). For the retrieval and multihop QA tasks, we include the performance of the most widely used dense vector retrieval method, a dense retriever combined with an LLM-based re-ranking approach, as well as retrieval using GPT-4o-based KGs. Lastly, for the multihop QA task, we also include results from GraphRAG and HippoRAG, both using KGs extracted by GPT-4o, as well as a non-RAG system where the LLM relies solely on its internal parametric knowledge to answer questions[2]. We provide full experimental details in Appendix E.

Multihop QA Frameworks We evaluate our D-SynthKG-8b model using three distinct indexing/ retrieval frameworks. All use LlamaIndex's *TreeSummarize* response builder with GPT-4o to generate answers from retrieved context. We only modify the query engine's prompt by adding specific instructions to produce concise answers tailored to the requirements of the test datasets.

- *LlamaIndex*: Uses LlamaIndex's KnowledgeGraphIndex to build a hybrid KG index combining keyword-based entity search and semantic similarity based triplet search. It retrieves the top-K relevant text chunks and associated subgraph for answer generation.
- *Chain-of-Triplet*: Decomposes a multi-hop question into sub-queries and retrieves the top 20 matching triplets per sub-query. Answers are generated using these triplets and associated propositions, allowing direct assessment of KG effectiveness (details in Appendix E.3.2).
- *Graph+LLM*: Retrieves the top 10 relevant chunks and 2-hop paths involving the question entity using our proposition-entity graph retriever to generate the final answer.

7 Results

7.1 KG Coverage Results

The multi-step SynthKG pipeline consistently generates more triplets and achieves higher coverage

[1]We use the Instruct variants of both models throughout

[2]To improve GraphRAG performance, we append the instruction: "Only provide the answer without any context. For yes/no questions, just mention yes or no. Do not cite data sources." at the end of each query. For GraphRAG, we report results using the local and drift modes, which yield the best performance; the global mode is excluded. For HippoRAG, we use GPT-4o for knowledge graph construction and apply our query synthesizer to the retrieved text chunks to generate the final answer, ensuring a fair comparison.

KG Source	Retriever	MuSiQue				2wiki				HotpotQA			
		Hits@2	Hits@10	MRR	MAP	Hits@2	Hits@10	MRR	MAP	Hits@2	Hits@10	MRR	MAP
None	Dense	41.32	64.19	79.89	40.17	62.22	74.72	97.86	55.73	66.55	89.45	91.98	60.68
None	Dense+LLM	47.60	67.02	84.44	44.26	72.63	76.70	97.77	58.65	**83.10**	92.10	**96.79**	**67.58**
Llama-3-8b	Graph + LLM	31.33	42.68	60.67	29.49	41.55	45.60	66.53	36.70	50.65	57.45	73.72	45.06
SynthKG-8b	Graph + LLM	50.62	65.17	86.65	45.43	65.25	69.65	95.54	54.79	76.55	86.35	92.69	63.44
Llama-3-70b	Graph + LLM	48.64	68.93	85.24	45.20	68.73	74.47	97.32	57.42	79.10	93.75	93.27	65.78
SynthKG-70b	Graph + LLM	53.70	72.23	88.81	48.32	73.23	78.80	98.80	60.09	81.90	94.40	94.62	66.93
D-SynthKG-8b	Graph + LLM	53.35	**72.78**	87.41	48.04	73.15	78.57	98.74	59.91	81.85	94.70	94.53	67.22
GPT-4o	Graph + LLM	**53.90**	70.38	**90.46**	**48.66**	**74.35**	**79.25**	**99.02**	**60.52**	82.90	**94.95**	93.98	67.15

Table 2: Retrieval performance. The best scores are **bolded**, and the second-best scores are underlined.

		MuSiQue		2wiki		HotpotQA		Average	
KG Source	Retrieval	EM	F1	EM	F1	EM	F1	EM	F1
None	None	0.100	0.220	0.190	0.340	0.290	0.440	0.193	0.333
None	Dense Retriever	0.237	0.376	0.380	0.497	0.471	0.641	0.363	0.505
None	Dense + LLM	0.260	0.398	0.414	0.531	0.509	0.678	0.394	0.536
GPT-4o	GraphRAG (local)	0.291	0.412	0.432	0.491	0.448	0.569	0.390	0.491
GPT-4o	GraphRAG (drift)	0.222	0.350	**0.497**	**0.629**	0.434	0.561	0.384	0.513
GPT-4o	HippoRAG	0.224	0.368	0.493	0.627	0.492	0.644	0.403	0.546
				Ours					
Llama-3-8b	LlamaIndex	0.155	0.259	0.366	0.461	0.405	0.555	0.308	0.425
Llama-3-70b	LlamaIndex	0.202	0.309	0.417	0.507	0.424	0.563	0.347	0.459
D-SynthKG-8b	LlamaIndex	0.217	0.320	0.435	0.528	0.451	0.608	0.367	0.485
Llama-3-8b	Chain-of-Triplet	0.131	0.244	0.305	0.381	0.278	0.469	0.238	0.365
Llama-3-70b	Chain-of-Triplet	0.188	0.299	0.351	0.428	0.370	0.517	0.303	0.415
D-SynthKG-8b	Chain-of-Triplet	0.243	0.383	0.410	0.507	0.400	0.579	0.354	0.490
Llama-3-8b	Graph + LLM	0.181	0.299	0.291	0.394	0.373	0.515	0.281	0.402
Llama-3-70b	Graph + LLM	0.297	0.437	0.400	0.501	**0.544**	0.705	0.413	0.548
D-SynthKG-8b	Graph + LLM	**0.320**	**0.459**	0.440	0.544	0.539	**0.706**	**0.433**	**0.569**

Table 3: Multi-hop QA evaluation (Exact Match and F1 score). The best scores for each framework are underlined.

than the commonly used single-step LLM prompting approach across all three datasets, for both LLaMA-3-8b and 70b models, highlighting the effectiveness of our SynthKG workflow (Table 1). Furthermore, our D-SynthKG-8b model outperforms the untrained Llama-3-8b, Llama-3-70b, and SynthKG-8b baselines, demonstrating the benefit of distilling the SynthKG pipeline using Llama-3-70b as the teacher. Remarkably, D-SynthKG-8b is also highly competitive with SynthKG-70b, despite being approximately $\sim 8\times$ smaller and relying on a single-step inference process. These results underscore the success of distilling SynthKG's capabilities into a smaller and efficient 8b model.

As previously discussed, our KG coverage metric emphasizes precision over recall, since manually annotating every generated triplet is highly challenging. To check for irrelevant triplets, we randomly sample 50 triplets (150 total) from D-SynthKG-8b's predictions on the three experimented datasets and manually verify each against the original text. Among these 150 triplets, only 4 are labeled incorrect and 5 meaningless, indicating that the generated triplets largely align with the source content.

7.2 Retrieval Results

Our D-SynthKG-8b model yields an average absolute improvement of 28.27 in Hits@2 over the pre-trained Llama-3-8b, 5.31 over SynthKG-8b, and 3.96 over the larger Llama-3-70b (Table 2). Notably, D-SynthKG-8b is highly competitive with the full SynthKG-70b pipeline—despite being significantly smaller and using single-step inference—further highlighting the effectiveness of distilling the multi-step SynthKG workflow into the smaller 8b model. It also performs comparably to GPT-4o (details in Appendix A.2). Additionally, our graph+LLM retriever achieves an average improvement of 12.75 in hits@2 over standard dense retrieval and 1.67 in hits@2 over the dense retriever with an LLM-based reranker.

7.3 Multi-hop QA Results

D-SynthKG-8b achieves the best overall performance across all three frameworks—LlamaIndex, Chain-of-Triplet, and Graph+LLM—outperforming both the Llama-

3-8b and the larger Llama-3-70b models. This demonstrates the general applicability and robustness of the KGs generated by our model. In the Graph + LLM framework, D-SynthKG-8b achieves the highest gain, with a +15.2% absolute improvement in EM accuracy over Llama-3-8b and a +2.0% gain over Llama-3-70b, leading to the best overall results. It also outperforms standard dense retrieval and dense+LLM reranking baselines. Notably, it surpasses GraphRAG and HippoRAG—two strong KG-based RAG systems built using KGs generated by the state-of-the-art GPT-4o model—highlighting the effectiveness and scalability of our approach despite relying on a significantly smaller LLM for KG construction.

7.4 Analysis

How effective is multi-step SynthKG in processing documents of increasing length? We compare our multi-step SynthKG framework with a single-step LLM prompting approach by examining the number of triples generated per 100 words for documents of varying lengths. We present the results in Figure 5 (Appendix A). For this analysis, we use a subset of the 1,000 documents employed to train the D-SynthKG-8b model. We observe that, for the single-step method, the proportion of extracted relations decreases as document length increases, with triple density dropping by about 60% when moving from 100-word documents to 1,200-word documents. In contrast, SynthKG's triple density remains nearly constant across all lengths, demonstrating its effectiveness in maintaining robust triple generation for both shorter and longer documents.

What is the optimal retrieval source? Our method constructs KGs which support multiple retrieval strategies. Given a question, we can retrieve text chunks using triples, propositions, or leverage the entire graph by integrating entities and propositions to rank text chunks. Optionally, all three approaches can be combined with an LLM for re-ranking. In Figure 3, we compare these three KG-based retrieval methods, along with their LLM-enhanced counterparts. We observe that proposition retrieval outperforms triples (+0.89 Hits@10) due to richer context, while graph-based retrieval achieves the best performance (+2.50 over propositions). LLM re-ranking further boosts all strategies, with graph-based retrieval seeing the largest gain of +3.59 in Hits@10.

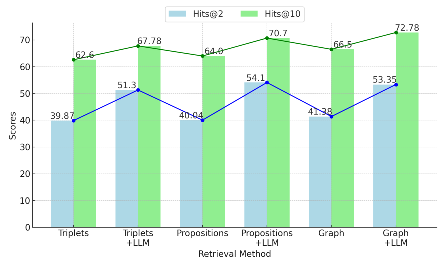


Figure 3: Performance comparison of different KG-based retrieval methods on multi-hop QA.

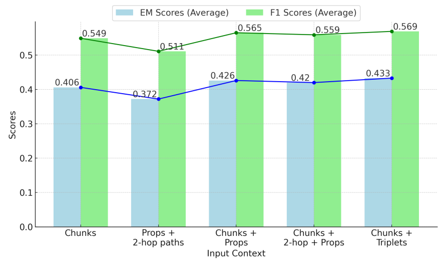


Figure 4: Ablation study results on different combinations of input context for multi-hop QA.

How can our KG improve RAG beyond retrieval? Our KG can enhance RAG beyond retrieval by enriching the retrieved context with structured signals—namely triplets, multi-hop relation paths, and propositions. In our ablation study (Figure 4), we combine retrieved text chunks with each of these signals to evaluate their impact on multi-hop QA performance. We find that adding related propositions (+2 EM) or 2-hop paths (+2.7 EM) improves accuracy, with 2-hop paths offering slightly better gains—likely due to their ability to capture more complex relationships across entities. Interestingly, combining both triplets and propositions does not yield additional improvements, suggesting overlapping information between the two.

8 Conclusion

In this work, we introduced SynthKG, a novel approach for synthesizing high-coverage KG training data using LLMs. Leveraging this synthesized data, we proposed Distill-SynthKG, an efficient model that distills the multi-step KG construction process into a single inference step. Our experiments demonstrated that Distill-SynthKG significantly improves KG coverage, retrieval accuracy, and QA performance across multiple datasets, outperforming a model nearly eight times its size. These results validate the effectiveness of Distill-SynthKG, highlighting its potential for scalable, ontology-free KG construction and its application in RAG tasks.

9 Limitations

The documents used for synthesizing KGs in this study were limited to the English language, which has been widely studied in existing NLP research. Additional work is needed to investigate the construction of KGs from documents in other languages. While we investigated synthetic data generation using two strong foundational LLMs (Llama-3-70b and GPT-4o), the use of other LLMs with SynthKG may yield different results than those reported in our study. Similarly, our distillation experiments were limited to two popular LLMs (Llama-3-8b and Mistral-7b) which we believe are representative of the capabilities of LLMs of similar size. Finally, our proposed benchmarks for evaluating KG coverage rely on automated question decomposition and triplet extraction using GPT-4o, which introduces the possibility of errors or omissions (see Appendix C.2 for human evaluation results).

References

AI@Meta. 2024. Llama 3 model card.

Zhen Bi, Jing Chen, Yinuo Jiang, Feiyu Xiong, Wei Guo, Huajun Chen, and Ningyu Zhang. 2024. Codekgc: Code language model for generative knowledge graph construction. *Preprint*, arXiv:2304.09048.

Harrison Chase. 2022. LangChain.

Bohan Chen and Andrea L. Bertozzi. 2023. Autokg: Efficient automated knowledge graph generation for language models. *Preprint*, arXiv:2311.14740.

Yew Ken Chia, Lidong Bing, Soujanya Poria, and Luo Si. 2022. Relationprompt: Leveraging prompts to generate synthetic data for zero-shot relation triplet extraction. *Preprint*, arXiv:2203.09101.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. Qlora: Efficient finetuning of quantized llms. *arXiv preprint arXiv:2305.14314*.

Linyi Ding, Sizhe Zhou, Jinfeng Xiao, and Jiawei Han. 2024. Automated construction of theme-specific knowledge graphs. *Preprint*, arXiv:2404.19146.

Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, and Jonathan Larson. 2024. From local to global: A graph rag approach to query-focused summarization.

Bernal Jiménez Gutiérrez, Yiheng Shu, Yu Gu, Michihiro Yasunaga, and Yu Su. 2024. Hipporag: Neurobiologically inspired long-term memory for large language models. *arXiv preprint arXiv:2405.14831*.

Jiuzhou Han, Nigel Collier, Wray Buntine, and Ehsan Shareghi. 2024. Pive: Prompting with iterative verification improving graph-based generative capability of llms. *Preprint*, arXiv:2305.12392.

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

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. *Preprint*, arXiv:2310.06825.

Yizhu Jiao, Ming Zhong, Sha Li, Ruining Zhao, Siru Ouyang, Heng Ji, and Jiawei Han. 2023. Instruct and extract: Instruction tuning for on-demand information extraction. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 10030–10051, Singapore. Association for Computational Linguistics.

Yu Su Kai Zhang, Bernal Jiménez Gutiérrez. 2023. Aligning instruction tasks unlocks large language models as zero-shot relation extractors. In *Findings of ACL*.

Hanieh Khorashadizadeh, Nandana Mihindukulasooriya, Sanju Tiwari, Jinghua Groppe, and Sven Groppe. 2023. Exploring in-context learning capabilities of foundation models for generating knowledge graphs from text. *Preprint*, arXiv:2305.08804.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.

Shilong Li, Yancheng He, Hangyu Guo, Xingyuan Bu, Ge Bai, Jie Liu, Jiaheng Liu, Xingwei Qu, Yangguang Li, Wanli Ouyang, Wenbo Su, and Bo Zheng. 2024. Graphreader: Building graph-based agent to enhance long-context abilities of large language models. *Preprint*, arXiv:2406.14550.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Jerry Liu. 2022. LlamaIndex.

Nandana Mihindukulasooriya, Sanju Tiwari, Carlos F Enguix, and Kusum Lata. 2023. Text2kgbench: A benchmark for ontology-driven knowledge graph generation from text. In *International Semantic Web Conference*, pages 247–265. Springer.

Anmol Nayak and Hari Prasad Timmapathini. 2023. Llm2kb: Constructing knowledge bases using instruction tuned context aware large language models. *Preprint*, arXiv:2308.13207.

OpenAI. 2024. Hello gpt-4o! `https://openai.com/index/hello-gpt-4o/`.

Diego Sanmartin. 2024. Kg-rag: Bridging the gap between knowledge and creativity. *Preprint*, arXiv:2405.12035.

Qi Sun, Kun Huang, Xiaocui Yang, Rong Tong, Kun Zhang, and Soujanya Poria. 2024. Consistency guided knowledge retrieval and denoising in llms for zero-shot document-level relation triplet extraction. In *Proceedings of the ACM on Web Conference 2024*, WWW '24, page 4407–4416, New York, NY, USA. Association for Computing Machinery.

Milena Trajanoska, Riste Stojanov, and Dimitar Trajanov. 2023. Enhancing knowledge graph construction using large language models. *Preprint*, arXiv:2305.04676.

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

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

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*, SIGIR 2024. ACM.

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, Brussels, Belgium. Association for Computational Linguistics.

Liang Yao, Jiazhen Peng, Chengsheng Mao, and Yuan Luo. 2024. Exploring large language models for knowledge graph completion. *Preprint*, arXiv:2308.13916.

Yuan Yao, Deming Ye, Peng Li, Xu Han, Yankai Lin, Zhenghao Liu, Zhiyuan Liu, Lixin Huang, Jie Zhou, and Maosong Sun. 2019. DocRED: A large-scale document-level relation extraction dataset. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 764–777, Florence, Italy. Association for Computational Linguistics.

Yunzhi Yao, Shengyu Mao, Ningyu Zhang, Xiang Chen, Shumin Deng, Xi Chen, and Huajun Chen. 2023. Schema-aware reference as prompt improves data-efficient knowledge graph construction. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '23, page 911–921, New York, NY, USA. Association for Computing Machinery.

Yuqi Zhu, Xiaohan Wang, Jing Chen, Shuofei Qiao, Yixin Ou, Yunzhi Yao, Shumin Deng, Huajun Chen, and Ningyu Zhang. 2024. Llms for knowledge graph construction and reasoning: Recent capabilities and future opportunities. *Preprint*, arXiv:2305.13168.

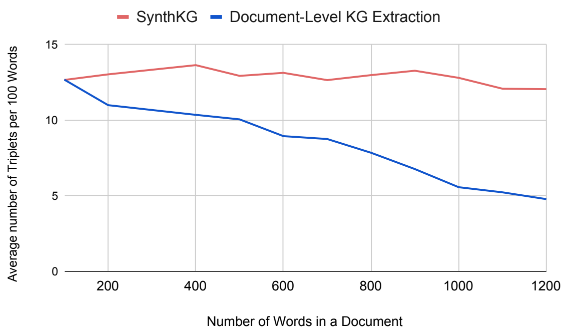


Figure 5: SynthKG maintains the triplet density consistently across documents of different lengths.

A Additional Analyses

A.1 Efficiency and Generalizability of Distill-SynthKG

In Table 4, we study three key important questions for developing Distill-SynthKG: 1. the efficiency of training Distill-SynthKG, 2. the effectiveness of various powerful LLMs for synthesizing training data, and 3. the generalizability of fine-tuning other smaller LLMs on synthesized data. To address these questions, we employ QLoRA (Dettmers et al., 2023) fine-tuning on approximately 1,000 synthetic Document-KG pairs, generated using GPT-4o and Llama-3.1-70b-Instruct. We provide fine-tuning configurations in Appendix D.1. Additionally, to answer the third question, we fine-tune another well-known base LLM, Mistral-7b-Instruct-v0.3 (Jiang et al., 2023).

We observe that both QLoRA fine-tuned models perform slightly below the fully fine-tuned model on retrieval and multi-hop QA tasks. However, the performance gap is minimal, demonstrating that QLoRA fine-tuning, even on a small dataset, remains competitive while requiring significantly fewer compute resources. The model fine-tuned on GPT-4o synthesized KGs shows slightly lower performance, which we attribute to more abstractive and atomic propositions in the synthesized data.

A.2 Comparison of Retrieval Performance with Proprietary Foundation Models

We include additional comparisons between our distilled model D-SynthKG-8b and a state-of-the-art proprietary foundation model, GPT-4o from OpenAI. We evaluate retrieval performance on three multihop QA benchmarks: 2Wiki, HotpotQA, and MuSiQue. We present the results in Table 5. The results show that D-SynthKG-8b achieves retrieval performance that is highly competitive with GPT-4o across all datasets. Notably, D-SynthKG-8b yields a slightly higher average Hits@10 (82.02 vs. 81.52), despite being significantly more cost-efficient. GPT-4o incurs \$2.50 per million input tokens and \$10 per million output tokens, while D-SynthKG-8b operates at only \$0.20 per million tokens (input or output), representing roughly **3%** of the inference cost. These findings highlight the practical advantages of our approach in cost-sensitive deployment scenarios.

A.3 Entity Distribution Analysis

When analyzing entity references across document chunks, we find that the overall average chunk distance per entity is 0.9. This metric represents the average number of chunks between an entity mention and its most recent previous mention. Further analysis reveals that 80.03% of entities have their most recent mention within a single paragraph, indicating that the majority of entity references are relatively localized within the document.

These findings support our design decision to only reference the immediately preceding chunk during decontextualization, as this approach effectively balances computational efficiency with adequate contextual information.

B LLM Prompts for SynthKG

We use prompts Figure 6, Figure 7 and Figure 8 for decontextualization, entity extraction and relation extraction respectively within SynthKG. We also provide an example of decontextualized chunk in Figure 9.

C KG Coverage Evaluation

C.1 Prompts and examples

Figure 10 shows the prompt that we used to generate the triplets, and Figure 11 the prompts that we used to instruct the model to generate the decomposed questions. Table 6 shows a question from HotpotQA dataset, and the generated decomposed questions and the triplet for the question answer pair.

C.2 Human evaluation of extracted triplets

To evaluate the quality of the GPT-4-generated KG coverage evaluation data, three authors of this work reviewed and validated both the decomposed questions and the proxy triplets. A random sample of 50 instances from each dataset was selected for human

KG Source	Retrieval Evaluation Hits@2	Hits@5	Hits@10	MRR	MAP	QA Evaluation EM	F1
D-SynthKG-7b$^{\text{GPT}}_{\text{Mistral}}$	0.680	0.776	0.809	0.932	0.575	0.417	0.556
D-SynthKG-7b$^{\text{Llama-3}}_{\text{Mistral}}$	0.685	0.780	0.811	0.931	0.578	**0.433**	0.565
D-SynthKG-8b	**0.695**	**0.792**	**0.820**	**0.936**	**0.584**	**0.433**	**0.569**

Table 4: Efficiency and Generalizability results for Distill-SynthKG. The results show average performance across MuSiQue, 2wiki, and HotpotQA datasets. D-SynthKG-7b$^{\text{GPT}}_{\text{Mistral}}$ and D-SynthKG-7b$^{\text{Llama-3}}_{\text{Mistral}}$ are Mistral-7b-Instruct-v0.3 models fine-tuned using QLoRA on 1000 document-KG pairs annotated by GPT-4o and Llama-3.1-70b-Instruct (respectively).

Previous paragraph from Document:
Gualala, the isolated Mendocino Coast town with a name that leaves most visitors tongue-tied, is on a new list of the 50 best places to live in the United States. Men's Journal magazine describes Gualala as an öutpost of adventure lifestyleïn its latest edition, which goes on sale today. The magazine describes Gualala (pronounced wa-LA-la by locals) as one of the b̈elow-the-radar places to a make a move on before the word gets out.T̈here were five such cities. The others were Homer, Alaska; Newport, Vt.; Logan, Utah; and Walla Walla, Wash. Rolling Stone magazine's Jann Wenner publishes Men's Journal, which has a paid circulation of about 620,000. Gualala joined three other California communities on the magazine's list: Santa Cruz, Mammoth Lakes and Bishop. Ẅe were looking for places that combined affordability, proximity to outdoor adventure and a generally undiscovered quality of life,s̈aid Erica Kestenbaum, a spokeswoman for Men's Journal.
Instruction:
Rewrite the below paragraph by resolving all entity coreferences with the preceding paragraph from document.
- Resolve all inter-sentence pronoun references.
- Make sure that all pronouns in a sentence refers to some named entity with in the same sentence.
- Explicitly mention entity names wherever necessary to remove ambiguity from a sentence. Remember to make each sentence clear and unambiguous.
- For each entity, use only the one most informative name.
- Do not generate anything except the rewritten paragraph.
Paragraph:
She said isolation played a factor. Ïn Northern California, it's particularly difficult to find a beautiful coastal setting that isn't entirely overrun,s̈he said. Gualala residents Monday were largely unaware of the magazine listing or the attention it could bring to the old logging town turned tourist center. A few coastal residents chuckled about any notion of affordability, given an influx of newcomers who've driven the median housing price to $580,000 compared to the median family income of $47,778. Others recalled an era when the Gualala region was better known for the logging of ancient redwoods, marijuana growing and boisterous beer drinking at the historic Gualala Hotel. Still there was a certain pride to the magazine's designation. Yvette White, a 25-year resident who works at the Gualala Sport; Tackle shop, said she's proud her town made it on the list.
Output:
Erica Kestenbaum said isolation played a factor. Ïn Northern California, it's particularly difficult to find a beautiful coastal setting that isn't entirely overrun,Ërica Kestenbaum said. Gualala residents Monday were largely unaware of the Men's Journal magazine listing or the attention it could bring to the old logging town turned tourist center. A few coastal residents of Gualala chuckled about any notion of affordability, given an influx of newcomers who've driven the Gualala's median housing price to $580,000 compared to the median family income of $47,778. Other Gualala residents recalled an era when the Gualala region was better known for the logging of ancient redwoods, marijuana growing and boisterous beer drinking at the historic Gualala Hotel. Still there was a certain pride to the Men's Journal magazine's designation. Yvette White, a 25-year Gualala resident who works at the Gualala Sport; Tackle shop, said she's proud her town made it on the list.
Previous paragraph from Document: [previous paragraph]
Instruction:
Rewrite the below paragraph by resolving all entity coreferences with the preceding paragraph from document.
- Resolve all inter-sentence pronoun references.
- Make sure that all pronouns in a sentence refers to some named entity with in the same sentence.
- Explicitly mention entity names wherever necessary to remove ambiguity from a sentence. Remember to make each sentence clear and unambiguous.
- For each entity, use only the one most informative name.
- Do not generate anything except the rewritten paragraph.
Paragraph: [paragraph]
Output:

Figure 6: The prompt for chunk decontextualization.

Extract all named entities from the document. Also generate the type for each entity.
Instructions
- Generate only the most informative name for each named entity. Example: if John P., Parker, John Parker are coreferential, only generate John Parker.
- Use your best understanding best on the domain of paragraph to decide appropriate entity types.
- Respond using json format provided below.

```
{
    "n1":{"name": "entity_name", "type": "entity_type_label"},
    "n2":{},
}
```

Below is an example for reference.
Paragraph: Tucked into Eli Lilly's year-end earnings report, the company revealed positive results from Synergy-NASH—its phase 2 study of tirzepatide in adults in nonalcoholic steatohepatitis (NASH), also known as metabolic dysfunction-associated steatohepatitis (MASH).
Output:

```
{
    "n1": {"name": "Eli Lilly", "type": "Organization"},
    "n2": {"name": "Synergy-NASH", "type": "Clinical Trial"},
    "n4": {"name": "tirzepatide", "type": "Drug"},
    "n5": {"name": "nonalcoholic steatohepatitis", "type": "Disease"},
    "n6": {"name": "metabolic dysfunction-associated steatohepatitis", "type": "Disease"},
    "n7": {"name": "year-end earnings report", "type": "Document"}
}
```

Figure 7: The prompt for graph node extraction

assessment, where evaluators rated the validity of the generated outputs. For decomposed questions from the HotpotQA dataset, the validity rate was found to be 85%, while the generated triplets for

Extract all facts from the document. For each fact, also generate all semantic triplets.
Instructions
- Consistently use the most informative name for each named entity in all facts and triplets.
- Avoid pronouns or ambiguous references in facts and triplets. Instead, directly include all relevant named entities in facts.
- Ensure that each semantic triplet contains head entity, predicate, and tail entity.
- Ensure that at least one (preferably both) entity in each semantic triplet is present in the given entities list.
- Respond using json format provided below:

```
{
    "f1":{
        "fact": "A factual statement describing important information (preferably about some entities) from the paragraph",
        "triplets: [["entity 1", "predicate", "entity 2"], ["entity 1", "predicate", "entity 3"]]
    },
    "f2":{},
}
```

Below is an example for reference.
Paragraph: Locked in a heated battle with Novo Nordisk's semaglutide franchise, Eli Lilly's tirzepatide is beginning to come into its own—both with regards to sales and amid attempts to show the dual GIP/GLP-1 agonist can strike out beyond diabetes and obesity. As Mounjaro, tirzepatide won its first FDA nod in Type 2 diabetes back in May 2022. An obesity approval followed last November, with that formulation of tirzepatide adopting the commercial moniker Zepbound. In 2023's fourth quarter, Mounjaro generated a whopping $2.2 billion in sales, a nearly eight-fold increase over the $279 million it pulled down during the same stretch in 2022. Year-to-date, the drug brought home around $5.2 billion in revenues, Lilly said in an earnings release Tuesday. Zepbound, for its part, generated $175.8 million during its first quarter on the market. Overall, Lilly reeled in around $9.4 billion in fourth-quarter sales, growing 28% over the $7.3 billion it made for the quarter in 2022.
Entities: Eli Lilly, Novo Nordisk, Tirzepatide, Semaglutide, GLP-1, GIP, FDA, Mounjaro, Zepbound
Output:

```
{
    "f1": {
        "fact": "Eli Lilly's tirzepatide is competing with Novo Nordisk's semaglutide franchise.",
        "triplets": [["Eli Lilly", "competing with", "Novo Nordisk"], ["Tirzepatide", "is competing with", "Semaglutide"]]
    },
    "f2": {
        "fact": "Eli Lilly is trying to show tirzepatide, the dual GIP/GLP-1 agonist, can strike out beyond diabetes and obesity.",
        "triplets": [["Eli Lilly", "is trying to show", "Tirzepatide"], ["Tirzepatide", "is a", "dual GIP/GLP-1 agonist"],
                     ["Tirzepatide", "can treat beyond", "Diabetes"], ["Tirzepatide", "can treat beyond", "Obesity"]]
    },
    "f3": {
        "fact": "Tirzepatide, under the brand name Mounjaro, received its first FDA approval for Type 2 diabetes in May 2022.",
        "triplets": [["Tirzepatide", "branded as", "Mounjaro"], ["Mounjaro", "won", "FDA approval"],
                     ["FDA approval", "for",  "Type 2 diabetes"], ["FDA approval", "was in", "May 2022"]]
    },
    "f4": {
        "fact": "Tirzepatide, under the brand name Zepbound, received an obesity approval in November 2022.",
        "triplets": [["Tirzepatide", "was branded as", "Zepbound"], ["Zepbound", "received", "Obesity approval"],
                     ["Obesity approval", "was in", "November 2022"]]
    },
    "f5": {
      "fact": "Mounjaro generated $2.2 billion in sales in the fourth quarter of 2023, an eight-fold increase from the $279 million during the same period
        in 2022.",
       "triplets": [["Mounjaro", "2023's fourth quarter sales", "$2.2 billion sales"], ["Mounjaro", "2022's fourth quarter sales", "$279 million"]]
    },
    "f6": {
        "fact": "Mounjaro brought in around $5.2 billion in revenues year-to-date in 2023, Lilly said in an earnings release Tuesday",
        "triplets": [["Mounjaro", "2023 sales year-to-date", "$5.2 billion revenues"]]
    },
    "f7": {
        "fact": "Zepbound generated $175.8 million in sales in its first quarter on the market.",
        "triplets": [["Zepbound", "first quarter sales", "$175.8 million"]]
    },
    "f8": {
        "fact": "Eli Lilly's fourth-quarter sales were around $9.4 billion, a 28% increase over the $7.3 billion during the same period in 2022.",
        "triplets": [["Eli Lilly", "2023 fourth-quarter sales", "$9.4 billion,"], ["Eli Lilly", "2022 fourth-quarter sales", "$7.3 billion,"]]
    }
}
```

Figure 8: The prompt for relation extraction

The Supreme Court ~~(SC)~~ on July 18 directed the Union of India to come up with proper guidelines within the time frame of two weeks to blacklist those builders, contractors and architects who are found to have constructed buildings against the sanctioned plan.\nThe ~~SC~~ Supreme Court said that the sealing and demolition of unauthorised constructions in the national capital will continue.\nThe ~~court~~ Supreme Court has passed the verdict after the Centre said that it had not asked the authorities in Delhi to stop the sealing drive against illegal constructions.\nA ~~SC~~ Supreme Court division bench, headed by Justice Madan Bhimrao Lokur told the Centre that hereafter, owners of illegal or encroached constructions would only be given 48 hours show cause notice as to why the building should not be sealed or demolished.\nThe ~~apex court~~ Supreme Court was informed today that Municipal Councillor, Mukesh Suryan, Chairman of Najafgarh wards committee, had allegedly prevented officers from carrying out sealing drives in the area, to which the top court sought ~~his~~ Mukesh Suryan's personal presence.\nThe bench also asked the Najafgarh authority to file an affidavit on the allegation that Deputy Commissioner Vishwendra Singh was transferred at the behest of Municipal Councillor Mukesh Suryan.

Figure 9: An example of decontextualization chunk.

```
You are given a question answer pair, please generate a relation triplet to represent the relationship.
Generate output in the format described below. “‘ head || relation || tail “‘ Note: - Must include relation, head entity and tail entity. Ensure that head entity is a subject of relation, and tail
entity is a direct object of relation. - You must use the given answer as the head or tail entity. - Specific entity is more preferable than generic entity. - Do NOT generate pronouns or
references in head and tail entities. —- Example 1:
Question: To whom was Messi’s goal in the first leg of the Copa del Rey compared? Answer: Diego Maradona
Output: Messi’s goal || was compared to || Diego Maradona
Example 2:
Question: The father of Chiang Hsiao-wen is whom? Answer: Chiang Ching-kuo
Output: Chiang Ching-kuo || the father of || Chiang Hsiao-wen
Question: {question}
Answer: {answer}
Output:
```

Figure 10: The prompt for generating triplet given a question and the answer.

```
You are given a multihop question, some facts that are used to reach the correct answer, and the correct answer. Your goal is to decompose the question into sub-question, and the
corresponding answer to each sub question.
Example 1
Question: What relationship does Fred Gehrke have to the 23rd overall pick in the 2010 Major League Baseball Draft?
Facts: He is the great-grandfather of Miami Marlin Christian Yelich Yelich was drafted out of high school by the Marlins in the 1st round (23rd overall) of the 2010 Major League
Baseball Draft.
Answer: great-grandfather
Decompose question answer pairs:
Who was the 23rd overall pick in the 2010 Major League Baseball Draft? Christian Yelich
What relationship does Fred Gehrke have to Christian Yelich? Great-grandfather
Question: {question}
Facts: {facts}
Answer: {answer}
Decompose question answer pairs:
```

Figure 11: The prompt for decomposing question into sub-questions and answers.

Dataset	Model	Hits@2	Hits@10	MAP	MRR
2Wiki	GPT-4o (OpenAI)	74.35	79.25	60.52	99.02
	D-SynthKG-8b	73.15	78.57	59.91	98.74
HotpotQA	GPT-4o (OpenAI)	82.90	94.95	67.15	93.98
	D-SynthKG-8b	81.85	94.70	67.22	94.53
MuSiQue	GPT-4o (OpenAI)	53.90	70.38	48.66	90.46
	D-SynthKG-8b	53.35	72.78	48.04	87.41

Table 5: Retrieval performance comparison between D-SynthKG-8b and GPT-4o across three multihop QA datasets.

both the Musique and HotpotQA datasets showed a validity rate of 86%. Annotators also provided reasons for any invalid ratings. Common issues with decomposed questions included the presence of previously unseen entities in the first sub-question or a poorly structured second sub-question. For triplets, the most frequent problem was the omission of minor details, such as dates, which did not necessarily make the triplet incorrect but affected its completeness. Only 4% of the cases involved an incorrect relation being extracted.

D Experimental Settings

D.1 QLoRa fine-tuning setup

In our experiments detailed in Appendix A.1, we employ the QLoRA fine-tuning. The training configuration used is as follows: we train models for 3 epochs, with an alpha value of 256 and a rank of 128. The learning rate, warmup steps and batch size are set to 0.00003, 50 and 8 respectively.

E Task-Specific Evaluation Settings

E.1 KG Coverage Task

We use the *‘all-MiniLM-L6-v2’* checkpoint to embed the triplets for semantic matching. For the coverage, we use threshold 0.88 as we manually check that this threshold representing a desirable semantic match.

E.2 Retrieval Task

We use *‘text-embedding-3-small’* for both dense retrieval and embedding propositions in KG-based retrievers. For both the graph and graph + LLM retrievers, we first construct the sub-graph by selecting the 200 propositions (M = 200) most similar to the question based on embedding similarity. Within the sub-graph, we traverse the KG, starting from the question entity, and select propositions within a 5-hop neighborhood (N = 5). For re-ranking the propositions in the LLM-based retriever and also for re-ranking chunks in Dense+LLM retriever, we use the GPT-4o model. The Dense+LLM retriever uses LlamaIndex’s implementation of the *LLMRerank* post-processor.

We evaluate retrieval performance at the passage level, following the setup used in HippoRAG . For each query, we evaluate whether the retrieved passages contain all ground-truth information required to answer the question. Retrieval metrics include Hits@2 and Mean Reciprocal Rank (MRR), computed based on the rank positions of the passages associated with ground-truth triplets. Specifically, a passage is considered relevant if it contains all the facts (triplets or propositions) needed for correct multi-hop reasoning. The evaluation code is directly adopted from HippoRAG.

To ensure comparability, we use the same benchmark datasets and experimental setup as HippoRAG, including 1,000 questions and a fixed set of candidate passages. The number of ground-truth passages per question is consistent with the original annotations. Our experiments are conducted on three multi-hop QA datasets: MuSiQue (11,656 passages), 2WikiMultiHopQA (6,119 passages), and HotpotQA (9,221 passages).

E.3 Multi-hop Question Answering Task

E.3.1 LlamaIndex Configuration

Table 7 presents the complete configuration of our LlamaIndex query engine setup.

E.3.2 Chain-of-Triplet

We design a triplet retrieval method that first breaks down the question into sub-queries in a triplet format. These triplet queries are then used to retrieve the most semantically matching triplet facts from the extracted knowledge graph. Specifically, it includes three steps to generate the final answer.

Step 1: Generate the Chain of Triplet Queries: given a question, we convert it into a series of triplet queries. Specifically, since our downstream task involves multi-hop QA, instead of generating a single triplet, we prompt the model to generate a chain of triplets. The generated triplets may contain placeholders that represent unknown entities. The prompt is shown in Figure 12.

Step 2: Triplet Retrieval: once the chain of triplet queries is generated, we retrieve the top 20 triplets for each query. During retrieval, if any of the triplets contain placeholders for uncertain entities, we attempt to resolve those entities by filling them with entities or relations from the previously retrieved triplets. For subsequent triplet queries in the chain, placeholders are updated with these resolved entities, thus refining the triplet queries progressively.

Step 3: Question Answering: with the question, the chain of triplet queries, and the retrieved triplets, we prompt the model to generate the answer. If the graph extraction method also retrieves associated propositions alongside the triplets, these propositions are provided to the model to further enhance the answer generation. The prompt is shown in Figure 13.

Graph + LLM : We use the same graph + LLM retriever hyper-parameters as in appendix E.2.

F Data Release and Training/ Inference Cost Considerations

We will make our annotated 100K data samples publicly available to support future research. With the rapid advancements in LLMs, researchers may choose to resynthesize data to better align with their specific applications. In such cases, we recommend using our cost-efficient approach, detailed in Appendix A.1, which provides a practical balance between performance and computational cost.

Below, we present the detailed training and inference costs, highlighting the efficiency of our SynthKG and DistilSynthKG methods.

Cost of Data Synthesis: With the Llama-3.1-70b-Instruct model, running the entire SynthKG pipeline on a single document requires processing an average of 11,849 input tokens. This results in a total of 2,675 average output tokens, distributed across intermediate steps such as decontextualization and the final entities, relations, and proposition generation. At a cost of $0.90 per million tokens[3], the total annotation cost per document is $0.0131. This translates to $13.08 for synthesizing training data for the D-SynthKG-7b$^{\text{Llama-3}}_{\text{Mistral}}$model and $392.28 for the D-SynthKG-8b model.

Cost of Model Training: After consolidating the data synthesized by SynthKG, each document contains an average of 1,723 input tokens (including prompts) and 1,248 output tokens, totaling 2,971 tokens per document. For a dataset of 30,000 documents, the total training token count is 89.13 million tokens. Fine-tuning Llama-3.1-8b-Instruct for one epoch on this dataset to obtain D-SynthKG-8b would cost $36.65. Additionally, fine-tuning Mistral-7b-Instruct-v0.2 for 3 epochs to obtain the D-SynthKG-7b$^{\text{Llama-3}}_{\text{Mistral}}$model would cost $3.67.

Combining data synthesis and fine-tuning costs, training D-SynthKG-8b would cost $428.93, while training D-SynthKG-7b$^{\text{Llama-3}}_{\text{Mistral}}$would cost $16.75.

[3]https://www.together.ai/pricing

```
There is a knowledge graph (entities and relations). Now, you are given a question, your task is to decompose this question into a chain of triplets used for searching fact from the graph.
The triplet should be in this format: head || relation || tail
Note: - Ensure that head entity is a subject of relation, and tail entity is a direct object of relation. - Do NOT generate pronouns or references in head and tail entities. - Do NOT generate entities that are not appeared in the question. - If an triplet includes an intermediate answer or the final answer, you can use # followed by an digit for reference. - The triplets order should be the same order for retrieving the facts from a knowledge graph.
Example 1:
Question: Who is older, Hampton Del Ruth or Ted Kotcheff?
Decompose Triplets:
Hampton Del Ruth || was born on || #1 Ted Kotcheff || was born on || #2
Example 2:
Question: In what town is Suffolk county hamlet that was served by the Suffolk Traction Company?
Decompose Triplets:
Suffolk Traction Company || served || #1 #1 || is located in || #2
Now please generate the decompose Triplets for the question: {question}
Decompose Triplets:
```

Figure 12: The prompt for generating chain of triplet query given a question, which are then used for triplet retrieval.

```
You are given a natural language question, triplets chain for this question, a set of retrieved tripletes, and a set of facts, please answer the question.
Question: {question}
Question Triplets Chain: {question triplets}
Retrieved Triplets: {retrieved triplets}
Retrieved Facts: {retrieved facts}
Short Answer no more than 3 words:
```

Figure 13: The prompt for generating the final answer given the original question, chain of triplet query, retrieved triplets and the facts.

Inference Cost: As mentioned in the cost of data synthesis, processing a single document using the SynthKG pipeline requires an average of 11,849 input tokens and 2,675 output tokens, totaling 14,524 tokens per document. At a cost of \$0.90 per million tokens, this amounts to \$0.031 per document. In contrast, with D-SynthKG-8b and D-SynthKG-7b $^{\text{Llama-3}}_{\text{Mistral}}$, each document requires 1,723 input tokens and 1,248 output tokens, totaling 2,971 tokens, with a cost of \$0.00267 per document. This is only 8.6% of the cost of SynthKG, demonstrating the significant efficiency gains from fine-tuning a distilled model.

G License information

We respect the license and intended use of all models and datasets employed in this study. Detailed license information is provided below.

Models. The Llama-3 models utilized in our study are licensed under the Meta Llama 3 Community License Agreement. The Llama-3.1 models utilized in our study are licensed under the Llama 3.1 Community License Agreement. The Mistral-7b-v0.3 model is licensed under the Apache 2.0 license.

Datasets. The BAAI/IndustryCorpus dataset used for extracting our synthetic training data is available under the Apache 2.0 license. The 2WikiMultihopQA dataset used in our evaluations is available under the Apache 2.0 license. The Musique dataset used in our evaluations is available under the Creative Commons Attribution 4.0 International license. The HotpotQA dataset used in our evaluations is available under the Apache 2.0 license. We will make our synthetic dataset publicly available under the MIT license, subject to terms and conditions of the Llama 3.1 Community License Agreement related to the use of Llama-3.1 outputs.

H Discussion on the Segmentation Strategy and Document Structure Preservation

While our segmentation and de-semanticization approach may appear simple, our design choice is guided by both empirical findings and practical considerations. First, our analysis (see Figure 5) shows that model performance consistently degrades as document length increases. Preserving structural associations in segmentation might result in longer input spans, which, based on our experiments, would still harm performance. Second, while we agree that capturing structural dependencies across multiple pages is a meaningful goal, it remains an open research challenge—particularly in open-ontology settings. Even state-of-the-art models like GPT-4o lack robust, generalizable pipelines for reliably preserving cross-page structure in a way that could be distilled into a smaller model. Given these limitations, we prioritize practicality and scalability by adopting a fixed-size (1K token) chunking approach. This method aligns with the constraints of retrieval-augmented generation (RAG) and enables

Question	Decomposed Question and Answer	Triplet (head ‖ relation ‖ tail)
The birthplace of George McCall Theal is a port city of what bay?	Where was George McCall Theal born? Saint John, New Brunswick	George McCall Theal ‖ was born in ‖ Saint John, New Brunswick
	Saint John is a port city of what bay? Bay of Fundy	Saint John ‖ is a port city of ‖ Bay of Fundy

Table 6: Example from HotpotQA dataset: the generated decomposed question answer pair and the triplet.

Parameter	Value
QA Prompt	You are an expert Q&A system that is trusted around the world. Always answer the query using the provided context information, and not prior knowledge. Some rules to follow: 1. Never directly reference the given context in your answer. 2. Avoid statements like 'Based on the context, ...' or 'The context information ...' or anything along those lines. 3. Provide only the essential information. Answer as briefly as possible, using keywords, phrases, or dates. Avoid full sentences or unnecessary details.
include_text	True
response_mode	tree_summarize
retriever_model	hybrid
num_chunks_per_query	10
similarity_top_k	2
graph_store_query_depth	2

Table 7: LlamaIndex query engine parameter settings.

effective, document-level KG construction at scale. We believe our approach provides a strong balance between empirical performance and real-world applicability under current technological constraints.

I Discussion on Manual Hyperparameter Selection

We manually tune two key hyperparameters in our framework: the semantic match score threshold for triplet coverage evaluation and the ROUGE score threshold for decontextualization. For the semantic match score, we intentionally select a higher threshold to ensure accuracy when determining whether two triplets are semantically equivalent. It is important to note that this threshold is only used for evaluation purposes and does not affect model training, inference, or RAG evaluation, thus having no impact on the model's learning or predictions. While the threshold is manually adjusted, downstream task performance reflects the reliability of our model.

For the ROUGE score threshold used in decontextualization, we conduct careful manual analysis to ensure its effectiveness. For both Llama-3.1-70b and GPT-4o, we examine a small subset of chunks with low ROUGE scores and find that most rewritings are accurate. We identify 593 edits in this subset, with only six containing incorrect modifications and four showing some loss of information. These results suggest that the decontextualization process generally yields high-quality, self-contained text.

The low ROUGE scores typically result from document footers or metadata, which are removed during the LLM's decontextualization process, and not from factual errors or information loss. Our analysis shows that approximately 72% of the chunks achieve a ROUGE score of 90 or higher, reflecting strong alignment with the original content. Chunks with scores below 0.70 make up less than 3%, and these are filtered out during the process to avoid any omission or extreme paraphrasing.