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 in domains that require knowledge-intensive reasoning. However, existing methods rely heavily on simple prompt-based approaches, which are inefficient when processing large-scale corpora. Additionally, such methods are prone to issues such as information loss and hallucination, especially when processing long documents, due to the lack of specialized design for KG construction. Additionally, there is a notable absence of evaluation datasets and methods for ontology-free KG construction. To address these challenges, we propose SynthKG, a multi-step document-level ontology-free KG generation workflow based on LLMs. By fine-tuning a smaller LLM on the generated document-KG pairs, we streamline the multi-step process into a single-step KG generation method called Distill-SynthKG, significantly reducing the number of LLM inference calls. Furthermore, we repurpose existing question-answering datasets to create KG evaluation datasets and introduce new evaluation metrics. Using KGs generated by Distill-SynthKG, we also develop a novel graph-based retrieval framework. Experimental results show that Distill-SynthKG not only surpasses all baseline models in KG quality—including models up to eight times larger—but also consistently outperforms them in retrieval and question-answering tasks. Our proposed graph retrieval framework also outperforms all baseline retrieval methods across multiple benchmark datasets. We make our SynthKG dataset and Distill-SynthKG model 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 or hallucination (Ji et al., 2023; Liu et al., 2024). This can result in KGs that are incomplete or contain inaccurate information. 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, enabling the 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 dis-

ambiguation 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 repurpose 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 main contributions are:

- We introduce SynthKG, a novel LLM-based workflow that generates high-quality, high-coverage document-level ontology-free KGs.
- We introduce Distill-SynthKG, which leverages SynthKG to synthesize training data and fine-tune a much smaller LLM. This approach simplifies the multi-step process into a single-step inference, significantly reducing the reliance on LLM calls.
- We propose new KG evaluation datasets by repurposing existing multi-hop QA datasets and introducing new evaluation metrics.
- We introduce a novel graph-based retrieval method that leverages KGs generated by Distill-SynthKG.
- 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 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) 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, advanced LLMs 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.

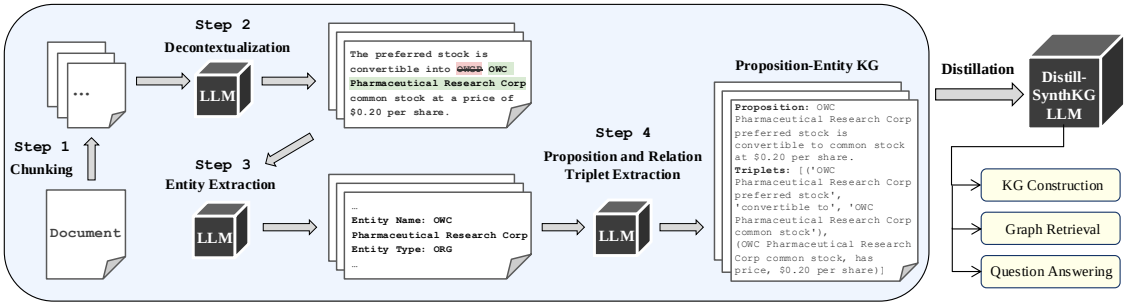

Figure 1: Our SynthKG data synthesis method (left side) is designed to generate high-coverage, ontology-free, document-level KGs. We distill this synthetic data into Distill-SynthKG (right side), which is applied to multiple downstream applications.

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 and relation extraction. These steps ensure high coverage of extracted entities and relations while minimizing information loss. We present an overview of SynthKG in Figure 1. Additionally, we provide details of all LLM prompts utilized in this process in Appendix D.

3.1.1 Document Chunking and Decontextualization

Directly inputting long texts into an LLM has been shown to result in information loss and potential hallucinations (Ji et al., 2023; Liu et al., 2024). To mitigate these 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 approach 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 during KG inference or retrieval. 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 decontextualized chunk in Figure 12.

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 address this, we use ROUGE scores to compare the original and decontextualized chunks and filter out those that show significant deviations due to the decontextualization process. Detailed experimental settings are provided in the Appendix B.1.

3.1.2 Entity and Relation Extraction

Following the paradigm of 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. One reason for this is that propositions can act as a chain-of-thought mechanism (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 a KG is constructed, we must repeatedly invoke an LLM, resulting in substantial computational or API costs, which hinders the scalability of KG construction. For example, processing a 1000-word document involves 12 LLM inference calls (4 chunks, with 3 calls per chunk for decontextualization, entity extraction, and relation extraction). To mitigate this, we distill SynthKG into a smaller LLM by leveraging the document-KG pairs produced in the original process. Specifically, we fine-tune a smaller LLM to take entire documents as input, enabling it to directly generate the same KGs as SynthKG (i.e., a series of quadruplets) in a single 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 and hallucination issues that often occur 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 introducing 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 A.1. In human evaluations, we found a high degree of validity in triplets extracted by GPT-4o with this approach; see Appendix A.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*: We use a cutoff threshold: if the semantic score for a ground truth triplet exceeds this 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 triplet with that of the ground truth triplet.

5 Proposition-Entity Graph Retriever

We introduce a novel retriever based on Proposition-Entity Graph, 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. We then construct a sub-graph consisting of these propositions and their linked entities, capturing the relations among the retrieved propositions. Next, we traverse the sub-graph starting from the entities mentioned in the question, and select only propositions within their N-hop neighborhood. This process 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 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, 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*[1] on 100K documents from *BAAI/IndustryCorpus*[2] for synthesizing KGs. We sample an equal number of documents from each of the following industry categories: politics, news, medicine, literature, finance, film & TV, computer science, automotive, technology, and education. Additional details on SynthKG are provided in Appendix B.1. We perform the KG synthesis using VLLM (Kwon et al., 2023) on 160 Intel® Gaudi 2 AI accelerators in the Intel® Tiber™ AI Cloud. We will publicly release our 100K generated document-KG pairs.

SynthKG Distillation: We train the *Meta-Llama-3-8B-Instruct* model[3] on 30K synthesized documents to directly generate corresponding KGs for entire input document. We use Optimum-Habana[4] and the Transformers (Wolf et al., 2020) Python library to train this distilled model on 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-8b** $_{\textbf{Llama-3}}$.

6.2 Evaluation Settings

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

Baselines Across all tasks, we compare KGs extracted by our Distill-SynthKG-8b $_{\text{Llama-3}}$ against KGs extracted by two baseline models: Llama-3-8b and Llama-3-70b[5]. For the retrieval task, we also include the performance of the most widely used dense vector retrieval method. We provide full experimental details in Appendix C.

Multihop QA Frameworks We evaluate our Distill-SynthKG model using three distinct indexing/ retrieval frameworks. In all frameworks, we employ LlamaIndex's *TreeSummarize* response builder, powered by GPT-4o, to generate answers from the retrieved context. The only modification

[1] https://huggingface.co/meta-llama/Llama-3.1-70B-Instruct

[2] https://huggingface.co/datasets/BAAI/IndustryCorpus

[3] https://huggingface.co/meta-llama/Meta-Llama-3-8B-Instruct

[4] https://github.com/huggingface/optimum-habana

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

Dataset	KG Source	Triplets	Semantic Score	Triplet Coverage	F1
MuSiQue	Llama-3-8b	93855	0.8111	0.3209	0.51
	Llama-3-70b	102119	0.8346	0.4034	0.56
	Distill-SynthKG-8b $_{\text{Llama-3}}$	**139376**	**0.8546**	**0.469**	**0.59**
2wiki	Llama-3-8b	41384	0.8281	0.4339	0.56
	Llama-3-70b	46100	0.8475	0.5041	0.58
	Distill-SynthKG-8b $_{\text{Llama-3}}$	**68800**	**0.8693**	**0.5827**	**0.59**
HotpotQA	Llama-3-8b	76906	0.8343	0.4179	0.58
	Llama-3-70b	82948	0.844	0.472	0.61
	Distill-SynthKG-8b $_{\text{Llama-3}}$	**123458**	**0.8693**	**0.5526**	**0.64**

Table 1: KG coverage evaluation. Distill-SynthKG-8b $_{\text{Llama-3}}$ outperforms both baselines in all metrics, showing significant improvements in triplet count, semantic score, and coverage.

Dataset	Retriever/ KG Source	Hits@2	Hits@5	Hits@10	MRR	MAP
MuSiQue	Dense Retrieval	41.32	55.28	64.19	79.89	40.17
	Llama-3-8b	31.33	39.19	42.68	60.67	29.49
	Llama-3-70b	48.64	63.14	68.93	85.24	45.20
	Distill-SynthKG-8b $_{\text{Llama-3}}$	**53.35**	**67.93**	**72.78**	**87.41**	**48.04**
2wiki	Dense Retrieval	62.22	70.62	74.72	97.86	55.73
	Llama-3-8b	41.55	44.2	45.6	66.53	36.7
	Llama-3-70b	68.73	73.05	74.47	97.32	57.42
	Distill-SynthKG-8b $_{\text{Llama-3}}$	**73.15**	**77.5**	**78.57**	**98.74**	**59.91**
HotpotQA	Dense Retrieval	66.55	81.80	89.45	91.98	60.68
	Llama-3-8b	50.65	55.5	57.45	73.72	45.06
	Llama-3-70b	79.10	90.35	93.75	93.27	65.78
	Distill-SynthKG-8b $_{\text{Llama-3}}$	**81.85**	**92.35**	**94.70**	**94.53**	**67.22**

Table 2: Retrieval evaluation. Distill-SynthKG-8b $_{\text{Llama-3}}$ outperforms all baselines on all metrics: Hits@2, Hits@10, and Mean Reciprocal Rank (MRR) and Mean Average Precision (MAP) scores.

is adding specific instructions to the query engine's original prompt, directing GPT-4o to produce concise answers tailored to the requirements of the test datasets.

- *LlamaIndex*: We use the KnowledgeGraphIndex module from LlamaIndex to build a hybrid KG index that combines keyword-based entity search with semantic similarity-based triplet search. This approach retrieves the top-K relevant text chunks along with the associated subgraph. Both the retrieved text chunks and the subgraph are then used to answer questions.
- *Chain-of-Triplet*: We decompose a multi-hop question into chains of triplet sub-queries and retrieve the top 20 matching triplets from the knowledge graph for each sub-query. We then generate answers based on these triplets and the associated propositions. This approach allows us to directly assess the effectiveness of the extracted KGs in answering multi-hop questions. Refer to Appendix C.3.2 for more details of this method.
- *Graph+LLM*: We use our proposition-entity graph retriever to retrieve the top 10 relevant chunks and 2-hop paths from the KG that include the question entity to generate the final answer.

7 Results

7.1 KG Coverage Results

Table 1 presents the results of the coverage evaluation of KGs generated by our Distill-SynthKG-8b $_{\text{Llama-3}}$ model, along with two baseline models. We observe a consistent pattern across all three test datasets: the KGs generated by our model outperform both the untrained Llama-3-8b model of the same size and the much larger Llama-3-70b model across all metrics and datasets. Particularly noteworthy is that, compared to the KGs generated by Llama-3-70b, our model produces 45% more triplets on average across the three datasets and achieves a 16% higher triplet coverage. These results highlight the high quality of the KGs synthesized by SynthKG and demonstrate that this multi-step process can be effectively distilled into a much smaller 8b model through fine-tuning.

KG Source	Retrieval	MuSiQue		2wiki		HotpotQA		Average	
		EM	F1	EM	F1	EM	F1	EM	F1
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
Distill-SynthKG-8b $_{\text{Llama-3}}$	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
Distill-SynthKG-8b $_{\text{Llama-3}}$	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
Distill-SynthKG-8b $_{\text{Llama-3}}$	Graph + LLM	**0.320**	**0.459**	**0.440**	**0.544**	0.539	**0.706**	**0.433**	**0.569**

Table 3: Multi-hop question answering evaluation. Distill-SynthKG-8b $_{\text{Llama-3}}$ outperforms both baselines under all three retrieval frameworks.

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

7.2 Retrieval Results

We present the retrieval results in Table 2. The higher KG coverage of the Distill-SynthKG-8b $_{\text{Llama-3}}$ model leads to consistently superior retrieval performance across all datasets and metrics, outperforming both the unfinetuned 8b-Instruct model of the same size and the much larger 70b-Instruct model. Specifically, the graph+LLM retriever using the KGs generated by our model achieves an average absolute improvement of 28.27 in hits@2 across all datasets compared to the KGs produced by the unfinetuned Llama-3-8b model. Additionally, the graph+LLM retriever with our model delivers an average improvement of 12.75 in hits@2 over standard dense retrieval methods.

7.3 Multi-hop QA Results

We present the results of the multi-hop QA task in Table 3, evaluating the general applicability of the KGs generated by our Distill-SynthKG-8b $_{\text{Llama-3}}$ model across different retrieval and QA frameworks. A good model, along with its extracted KGs, should ideally demonstrate robustness across various retrieval and question-answering methodologies. The results indicate that under three distinct QA frameworks (LlamaIndex, Chain-of-Triplet, and Graph + LLM), our model's generated KGs consistently outperform its unfinetuned 8B counterpart as well as the larger 70B model. In the Chain-of-Triplet framework, which relies exclusively on the KG for answering questions, our distilled model shows an absolute gain of +11.6% in EM accuracy compared to the Llama-3-8b model, while the improvement is +5.9% in the LlamaIndex framework. This further underscores the superior coverage of the KG extracted by our distilled model. In the Graph + LLM framework, our distilled model achieves the highest gain, with a +15.2% absolute increase in EM accuracy over the Llama-3-8b model, resulting in the best overall performance. This highlights the importance of using our entity-proposition graph retriever in conjunction with the distilled KG model to achieve optimal results.

7.4 Ablations

Efficiency and Generalizability of Distill-SynthKG In Table 4, we study three key important questions for developing Distill-SynthKG: 1.

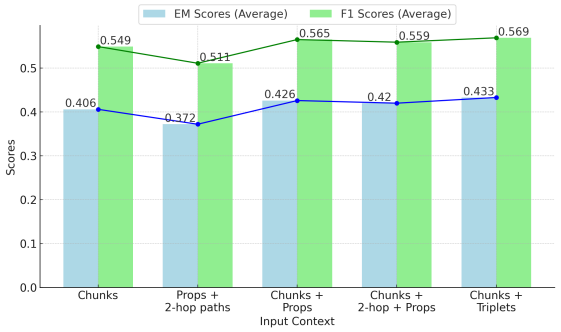

Figure 2: Ablation study results on different combinations of input context for multi-hop QA.

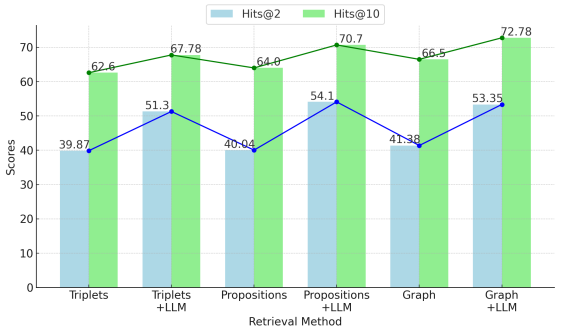

Figure 3: Performance comparison of different KG-based retrieval methods on multi-hop QA.

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 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 B.2. Additionally, to answer the third question, we fine-tune another well-known base LLM, Mistral-7B-Instruct-v0.3.

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.

Benefits of KG in multi-hop QA Along with text chunks, our KG provides both triplets (or relation paths between entities) and propositions, which can all be leveraged as context to answer questions. To determine the optimal combination of these inputs for multi-hop QA, we conduct an ablation study, as shown in Figure 2. Our findings indicate that adding either related propositions (+2 EM) or 2-hop paths (+2.7 EM) from the KG to the top-K chunks improves performance on multi-hop QA task. Notably, 2-hop paths perform slightly better, likely due to their ability to capture more complex relationships between entities by connecting multiple relations across two propositions. Lastly, combining both triplets and propositions did not result in further improvement, which may be due to redundant information being captured by both triplets and propositions.

KG-based Retrieval Methods Our KG supports multiple retrieval methods: (1) retrieval based on the similarity between triplets and questions, (2) retrieval based on the similarity between propositions and questions, and (3) graph-based retrieval, which ranks chunks by integrating entities and propositions. Optionally, all methods can incorporate an LLM for reranking. In Figure 3, we compare three retrieval methods utilizing KGs. Our findings indicate that propositions outperform triplets, achieving an increase of +0.89 in Hits@10, which is expected, as propositions offer more detailed information than triplets. The graph retrieval method demonstrates even greater performance by integrating both entities and propositions, yielding a +2.50 higher Hits@10 score compared to propositions alone. Secondly, for all three retrieval types, incorporating an LLM to re-rank propositions and triplets significantly enhances retrieval performance, with an increase of +3.59 in Hits@10 for the best performing graph method.

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 nine 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 A.2 for human evaluation results).

References

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.

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.

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.

Ziwei Ji, Tiezheng Yu, Yan Xu, Nayeon Lee, Etsuko Ishii, and Pascale Fung. 2023. Towards mitigating LLM hallucination via self reflection. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 1827–1843, Singapore. Association for Computational Linguistics.

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.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173.

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: Multi-hop 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.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

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.

A KG Coverage Evaluation

A.1 Prompts and examples

Figure 4 shows the prompt that we used to generate the triplets, and Figure 5 the prompts that we used to instruct the model to generate the decomposed questions. Table 5 shows a question from HotpotQA dataset, and the generated decomposed questions and the triplet for the question answer pair.

A.2 Human evaluation of extracted triplets

To assess the quality of the GPT-4-generated data, three authors of this work reviewed and validated both the decomposed questions and the proxy triplets. Specifically, we randomly selected 50 samples from each dataset and asked human evaluators to rate the generated outputs as either good or bad. For the decomposed questions generated for the HotpotQA dataset, the reported good ratio was 85%, while for the triplets generated for the Musique and HotpotQA datasets, the good ratios were both 86%. Additionally, annotators were asked to provide reasons for why certain generated outputs were rated as poor. For low-quality decomposed questions, the main issues reported were that the first sub-question often contained unseen entities, or the second sub-question was not well-formed. In the case of low-quality triplets, the most common issue was missing minor details (e.g., dates), which, while incomplete, did not necessarily invalidate the triplet. In only 4% of cases, the problem was identified as an incorrect relation being extracted.

B Experimental Settings

B.1 SynthKG Setup

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 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.

In Figure 6, we present the number of triplets generated per 100 words in documents of varying lengths. We observe that the ratio of extracted relations decreases as document length increases, with triplet density dropping by approximately 60% from 100-word documents to those of 1200 words. This highlights the need for SynthKG in data synthesis to address this gap.

B.2 QLoRa fine-tuning setup

In our experiments detailed in Section 7.4, 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.

C Task-Specific Evaluation Settings

C.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.

C.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 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. For re-ranking the propositions in the LLM-based retriever, we use the GPT-4o model.

C.3 Multi-hop Question Answering Task

C.3.1 LlamaIndex Configuration

Table 6 presents the complete configuration of our LlamaIndex query engine setup.

C.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 question answering, instead of generating a single triplet, we prompt the model to generate a chain of triplets. The generated triplets

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 5: 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 6: LlamaIndex query engine parameter settings.

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 4: 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 5: The prompt for decomposing question into sub-questions and answers.

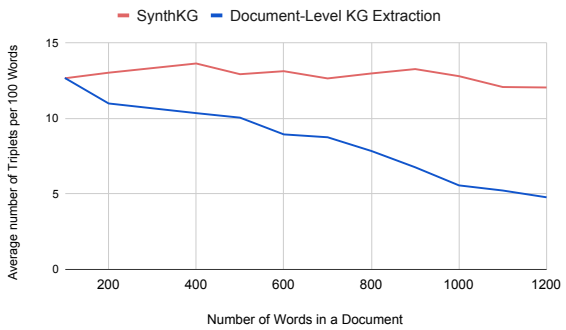

Figure 6: Average number of triplets per 100 words for documents of different lengths.

may contain placeholders that represent unknown entities. The prompt is shown in Figure 7.

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 facts alongside the triplets, these facts are provided to the model to further enhance the answer generation. The prompt is shown in Figure 8.

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

D LLM Prompts for SynthKG

We use prompts Figure 9, Figure 10 and Figure 11 for decontextualization, entity extraction and relation extraction respectively within SynthKG.

E 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

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 7: 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 8: The prompt for generating the final answer given the original question, chain of triplet query, retrieved triplets and the facts.

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 comparedtothemedianfamilyincomeof$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 amp; 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 comparedtothemedianfamilyincomeof$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 amp; 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 9: The prompt for chunk decontextualization.

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.

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 10: The prompt for graph node extraction

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 11: 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 12: An example of decontextualization chunk.