Knowledge Graph Prompting for Multi-Document Question Answering

Anonymous Author(s)
Affiliation
Address
email

Abstract

The 'pre-train, prompt, predict' paradigm of large language models (LLMs) has achieved remarkable success in open-domain question answering (OD-QA). However, few works explore this paradigm in the scenario of multi-document question answering (MD-QA), a task demanding a thorough understanding of the logical associations among the contents and structures of different documents. To fill this crucial gap, we propose a Knowledge Graph Prompting (KGP) method to formulate the right context in prompting LLMs for MD-QA, which consists of a graph construction module and a graph traversal module. For graph construction, we create a knowledge graph (KG) over multiple documents with nodes symbolizing passages or document structures (e.g., pages/tables), and edges denoting the semantic/lexical similarity between passages or intra-document structural relations. For graph traversal, we design an LM-guided graph traverser that navigates across nodes and gathers supporting passages assisting LLMs in MD-QA. The constructed graph serves as the global ruler that regulates the transitional space among passages and reduces retrieval latency. Concurrently, the LM-guided traverser acts as a local navigator that gathers pertinent context to progressively approach the question and guarantee retrieval quality. Extensive experiments underscore the efficacy of KGP for MD-QA, signifying the potential of leveraging graphs in enhancing the prompt design for LLMs. Our code will be released upon publication.

1 Introduction

Due to the emergence of large language models (LLMs), the "pre-train, prompt, predict" paradigm has revolutionized natural language processing (NLP) in real-world applications, such as open-domain question answering (O-QA), fact-checking (FC), and arithmetic reasoning (AR) [1, 6, 2, 22, 38, 32]. However, no significant efforts have investigated this framework in the scenario of multi-documental question answering (MD-QA), which enjoys practical usage in academic research, customer support, and financial/legal inquiries that require analysis/insights derived from multiple documents [3, 37].

To investigate the capability of LLMs for MD-QA, we randomly sample multi-document questions from the development set of 2WikiMQA [14] and MuSiQue [41], and then prompt LLMs in four different strategies for the answer[1]. Successfully answering these questions requires knowledge of multiple Wikipedia documents. As shown in Figure 1, on 2WikiMQA and MuSiQue, directly prompting LLMs without providing any context, i.e., None, achieves only 25.07%/10.58% F1 and 18.60%/4.60% EM on 2WikiMQA and MuSiQue, which is far less than 59.69%/47.75% F1 and 40.20%/30.60% EM when prompting with supporting facts[2] provided

[1]Detailed experimental setting is presented in Section 5.

[2]Supporting facts: passages that are assumed to contain the answer to the question.

Submitted to 37th Conference on Neural Information Processing Systems (NeurIPS 2023). Do not distribute.

as contexts, i.e., the Golden one. This demonstrates the limitation of fulfilling MD-QA using solely the knowledge encoded in LLMs. One standard solution to overcome this limitation in conventional O-QA and single document question-answering (D-QA) [27, 45] is to retrieve grounding contexts and derive faithful answers from the contexts, i.e., retrieve-and-read [21, 56].

However, unlike O-QA and D-QA, the primary challenge of MD-QA roots in its demands for alternatively retrieving and reasoning knowledge across different documents [5, 31]. For example, successfully answering questions in Figure 2(a)-(b) requires reasoning over distinct passages from two different documents (in these two cases, Wikipedia pages). Moreover, each document is essentially a compilation of multi-modality structured data (e.g., pages, sections, paragraphs, tables, and figures) and some questions may specifically ask for the content in specific structures, which necessitates a comprehensive grasp of these complex document structures. For example, the question in Figure 2(c) asks about the difference between Page 1 and Table 2, which is unanswerable if leveraging heuristic methods like BM25 or deep-learning ones like DPR [22]. Building on previous challenges, the advent of LLMs introduces new complexities.

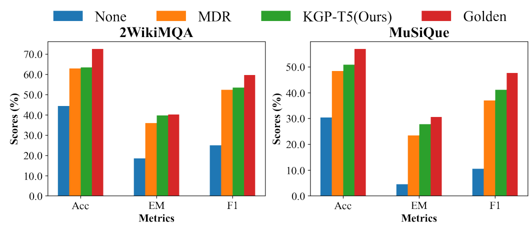


Figure 1: MD-QA performance when prompting ChatGPT with contexts retrieved in different ways.

For the challenge of alternatively retrieving and reasoning knowledge across different documents, although previous works train a multi-hop retriever [44, 52] to imitate such process by sequentially fetching the next passage based on the already-retrieved ones, none of them explore the potential of engaging LLMs into this process. More recent works design different prompting strategies such as Chain/Tree/Graph-of-thought [40, 42, 48, 49] to guide LLMs approaching answers progressively. However, prompting non-open-sourced LLMs back and forth incurs forbiddable latency as well as unaffordable consumption. In addition, how to integrate different document structures into the prompt design so that LLMs can understand them is still an open-ended question.

In view of the above challenges, we propose a knowledge graph prompting (KGP) method for enhancing LLMs in MD-QA. Specifically, we construct a knowledge graph (KG) over the given documents with nodes symbolizing passages or document structures and edges denoting their lexical/semantic similarity between passages or intra-document structural relations. Then for the first challenge of alternative retrieving and reasoning knowledge across different documents, we address it by alternatively prompting LMs to generate the next evidence to approach the question, i.e., reasoning, and selecting the most promising neighbor to visit next from the constructed KG based on the generated evidence, i.e., retrieval. Moreover, we apply the instruction fine-tuning strategy to augment the reasoning capability of our own LMs and hence refrain from repeatedly prompting non-open-sourced LLMs for evidence generation. For the multi-modality challenge, we add different types of nodes to the KG characterizing different document structures and hence enabling content retrieval within those specific structures. We highlight our contributions as follows:

- **Generally-applicable KG Construction.** We propose three KG construction methods over documents, with passages or document structures as nodes and their lexical/semantical similarity or structural relations as edges. Then we empirically evaluate the quality of the constructed KGs in MD-QA by checking the level of overlap between the neighborhood and the supporting facts for each question (Figure 4). We also provide a comprehensive summary of our proposed and existing KG construction methods in Table 5 in Supplementary.
- **Engaging KG for Prompt Formulation.** We design a Knowledge Graph Prompting (KGP) method, which retrieves the question-relevant contexts by traversing the constructed KG. Meanwhile, we fine-tune LMs that guide the graph traverser to adaptively navigate the most promising neighbors for approaching the question based on the already-visited nodes (retrieved passages).
- **Case Studies Verifying MD-QA Framework.** We provide insightful analysis, including comparing the quality of the constructed KGs in MD-QA and the performance of using different LMs to guide the graph traversal. We design a user interface and conduct case studies on visualizing MD-QA in Section A.7 in Supplementary.

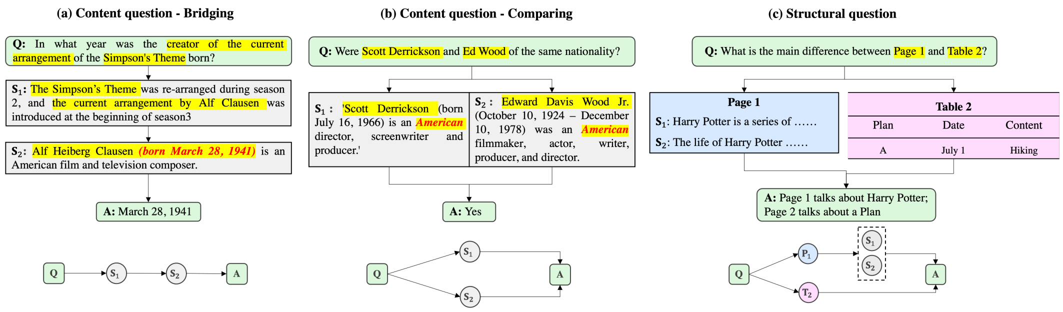


Figure 2: Questions requiring reasoning and retrieving over passages/pages/tables from multiple documents. **(a) Bridging questions** rely on sequential reasoning while **(b) Comparing questions** rely on parallel reasoning over different passages. **(c) Structural questions** rely on fetching contents in the corresponding document structures.

2 Related Work

Question answering Question Answering (QA) aims to provide answers to users' questions in natural language [30, 56], and most QA systems are composed of information retrieval (IR) and answer extraction (AE) [21, 26]. In IR, the system searches for query-relevant factual passages using heuristic methods (BM25) [36] or neural-ranking ones (DPR) [22]. In AE, the final answer is usually extracted as a textual span from related passages. Although this framework has been broadly applied in O-QA [26, 29] and D-QA [27, 45], no previous work focus on MD-QA, which demands alternatively reasoning and retrieving knowledge from multiple documents. To tackle this issue, we construct the KG to encode the logical associations among different passages across multiple documents and design an LM-guided traverser to alternatively generate the reason and visit the most matching passage node.

Pre-train, Prompt, and Predict with LLMs With the emergence of LLMs, the paradigm of 'pre-train, prompt, predict' has gained magnificent popularity in handling a wide spectrum of tasks [13, 24, 54]. This approach begins with pre-training LLMs by pretext tasks to encode world knowledge into tremendous parameters [43] followed by a prompting function to extract pertinent knowledge for downstream tasks [46]. Recent advancements explore different prompting strategies to enhance LLMs' reasoning capabilities [42, 48]. In contrast to that, our work offers a novel perspective by transforming the prompt formulation into the KG traversal.

3 Knowledge Graph Construction

Following [17], let $G = (\mathcal{V}, \mathcal{E})$ be a knowledge graph constructed from a set of documents $\mathcal{D}$, where the node set $\mathcal{V} = \{v_i\}_{i=1}^n$ representing document structures (e.g., passages/pages/tables, etc.) and the edge set $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$ representing the connections among different nodes (e.g., semantic/lexical similarity and belonging relations among document structures, etc.). Let $\mathcal{X} = \{\mathcal{X}_i\}_i^n$ be node features and $\mathcal{X}_i$ corresponds to the feature of node v_i, the form of which could be the text for the passage, the markdown for the table and the page number for the page.

Despite numerous well-established KGs [15, 23], they treat nodes/edges as entities/relations, which necessitates sophisticated relational extraction techniques and thereby limits their applicability in general domains [18]. Additionally, their primary focus on the Wikipedia domain also restricts their usage for answering non-Wikipedia questions such as ones over legal or financial documents. To remedy this issue, we propose generally-applicable KG construction methods.

We first analyze two representative questions in Figure 2(a)-(b) to motivate our KG construction. Answering these two questions necessitates the deduction of logical associations among different passages. These associations are encoded either through 1) lexical similarity: common keywords shared among different passages, e.g., 'Alf Clausen' bridges passage $\mathbf{S}_1$ and passage $\mathbf{S}_2$ in Figure 2(a), or 2) semantic similarity: syntactic elements that convey semantic relations, e.g., 'nationality' and 'American director' in Figure 2(b). This motivates us to construct the graph by modeling passages as nodes and their lexical/semantic similarity as edges. More specifically in Figure 3, we split each document into individual passages, and for each passage $\mathbf{S}_i$, we add a node v_i to the KG with its feature being the text of that passage $\mathcal{X}_i$. Then we add edges by checking the lexical/semantic similarity between pairs of passage nodes.

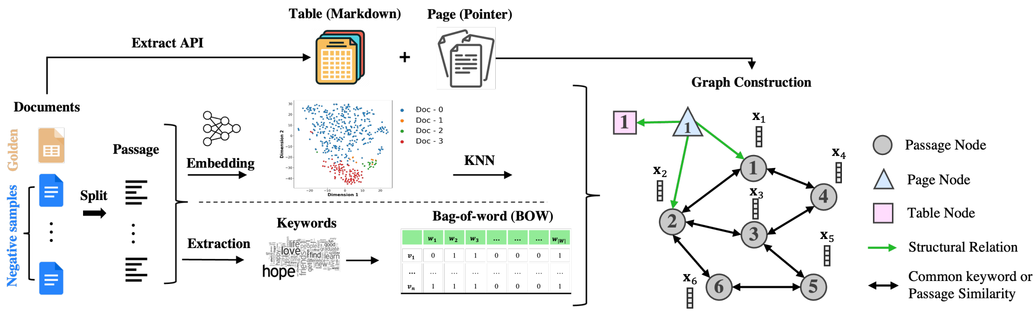


Figure 3: Knowledge Graph Construction. We split each document in the document collection into passages. For each passage, we either directly obtain their embeddings via pre-trained encoders or extract their keywords to build bag-of-word (BOW) features. Then we connect two passages based on their embedding similarity or whether they share common keywords. Additionally, we extract tables/pages via Extract-PDF API and add them as structural nodes to the KG. If pages include passages and tables, we add a directed edge to denote the belonging relations. The table nodes include the markdown formatted content of that table as Figure 8 in Supplementary has empirically shown that LLMs are able to understand tables in this format.

3.1 TF-IDF KG Construction

For adding edges according to lexical similarity, we first apply TF-IDF keyword extraction [35] over each document to filter out meaningless words such as supporting verbs and articles, which reduces the dimension of BOW features, sparsifies the constructed graph and increases the efficiency of the graph traversal. In addition, we add the document title into the extracted keyword set since some questions focus on title entities. We collect the extracted keywords from all documents to form the keyword space $\mathcal{W}$ and then connect two passages if they share any common keyword in $\mathcal{W}$.

3.2 KNN-ST/MDR KG Construction

For adding edges according to semantic similarity, we can readily employ pre-existing models such as sentence transformers to generate passage embedding $\mathbf{X}_i$ for each node v_i and subsequently compute pairwise similarity matrix to construct the K-nearest neighbor (KNN) graph. However, these off-the-shelf models, typically trained on tasks not so-related to MD-QA, may not adequately encapsulate necessary logical associations in their embedding similarity demanded by the question. To overcome this problem, we follow the training strategy of MDR [44] and train a sentence encoder by predicting the subsequent supporting facts based on previously supporting facts, thereby endowing the encoder with reasoning capability. Consequently, the embedding similarity and the corresponding constructed KNN graph fundamentally encapsulate the necessary logical associations between different passages.

3.3 TAGME

Moreover, we employ TAGME [28] to extract Wikipedia entities from each passage and construct the graph based on whether two passage nodes share common Wikipedia entities.

In addition to passage nodes, we further add structural nodes into the graph by extracting document structures via Extract-PDF [3]. In this paper, we only consider adding pages and tables but the constructed KG can include more different types of document structures. The feature of table nodes is the markdown since LLMs can understand this as demonstrated in Figure 8 in Supplementary. The feature of page nodes is the page number and we add directed edges from it to sentence/table nodes in that page. *Note that we do not aim to propose a one-size-fits-all KG construction method. Instead, we seek to compare the merits and limitations of various methods in Table 5, offering guidance on which KGs are best suited for specific scenarios.*

To verify the constructed KGs indeed encode the necessary information for MD-QA, we randomly sample questions from HotpotQA and construct KGs over the set of documents for each of these questions using our proposed methods. We vary the hyperparameters to control the sparsity of

[3] `https://developer.adobe.com/document-services/docs/overview/pdf-extract-api/`

the constructed graph and measure how much percentage of the supporting facts are covered by neighbors of the seeding passages initialized by TF-IDF. Details about the construction methods and their hyperparameters are included in Section A.5 in Supplementary. As shown in Figure 4, as the constructed graph becomes denser, the chance that the neighboring node passages hit the supporting facts increases (i.e., SF-EM increases) although the redundant information also increases (i.e., the precision decreases). Given the common keywords shared between one passage to all other passages are typically far less than the total number of passages across all documents, the density of the constructed graph by TF-IDF would be upper-bounded, causing lower SF-EM (evidenced by SF-EM below 0.7 in Figure 4 for TF-IDF curve). For TAGME, we empirically find it identifies a larger quantity of entities mentioned in a single passage, which leads to a denser graph and causes the starting SF-EM of TAGME to be already around 0.95. In addition, since KNN-MDR is pre-trained by predicting the next supporting facts [44] on HotpotQA, it achieves better trade-off than KNN-ST where the embeddings are directly from the sentence transformer without dataset-specific pre-training.

To summarize, although high SF-EM indicates that the supporting facts for most questions are fully covered by the neighbors of seeding passages, low precision signifies that most of these neighboring passages are irrelevant to the question. Therefore, if we blindly perform graph traversal without any question-tailored adaptation, our retrieved contexts would include redundant passages and compromise the capability of LLMs in MD-QA (which is also verified by the low performance of KGP w/o LM in Table 3). To remedy this issue, in the next section, we introduce an LM-guided graph traverser to adaptively visit neighboring passages that are most conducive to answering the given question.

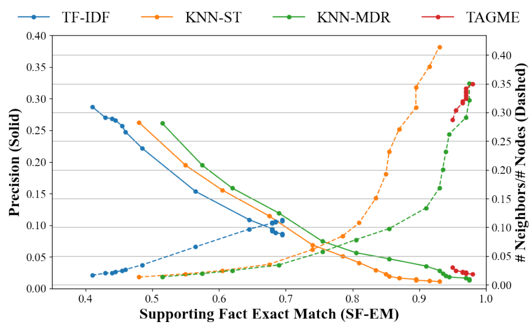


Figure 4: MD-QA performance when prompting ChatGPT with contexts retrieved in different ways.

4 LM-guided Graph Traverser

A natural solution to enable adaptive graph traversal is to rank the candidate nodes, i.e., the neighbors of the already-visited nodes in our case, thereby determining which ones to visit next. The most straightforward way is to apply heuristic-based fuzzy matching or embedding-based similarity ranking, which cannot capture the intrinsic logic relations between the already traversed paths and the nodes to visit. Instead, we fine-tune a language model (LM) to guide the graph traversal toward the next most promising passages in approaching the question based on the visited passages.

Given a question q asking about the document content, the LM-guided graph traverser reasons over previously visited nodes/retrieved passages $\{s_k\}_{k=0}^{j}$ and then generates the next passage s_{j+1} as follows:

$$s_{j+1} = \underset{v \in \mathcal{N}_j}{\arg\max}\, \phi(g(\mathcal{X}_v), f(||_{k=0}^{j}\mathcal{X}_k)), \tag{1}$$

where $||_{k=0}^{j}\mathcal{X}_k$ concatenates the textual information of previously retrieved passages/visited nodes. For the choice of f, one way is to employ encoder-only models like Roberta-base [2, 44, 52] and correspondingly g would be another encoder model with $\phi(\cdot)$ being the inner product measuring the embedding similarity. Another way is to employ encoder-decoder models such as T5 [4, 39] and correspondingly g would be an identity function with $\phi(\cdot)$ measuring the textual similarity. To mitigate the hallucination issue [19] and enhance the reasoning capability [42] of LMs, we further apply instruction fine-tuning to f [7] by predicting the next supporting facts based on previous supporting facts, thereby integrating commonsense knowledge encoded originally in their pre-trained parameters with the enhanced reasoning capability inherited from the instruction fine-tuning. After visiting the top-scoring nodes selected from the candidate neighbor queue by Eq (1), the candidate neighbor queue is updated by adding neighbors of these newly visited nodes. We iteratively apply this process until hitting the preset budget. Next, we illustrate the above process with an example in Figure 5 but leave the comprehensive traversal algorithm in Algorithm 1 in Supplementary.

In Figure 5, the content-based question asks ‘In what year was the creator of the current arrangement of Simpson’s Theme born?’. We use TF-IDF search to initialize our seeding passage Node 1, which

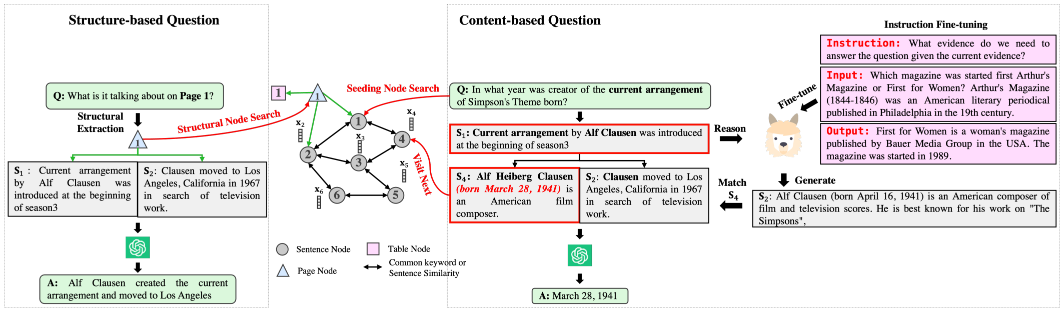


Figure 5: LM-guided graph traverser for context retrieval. For questions on document structures (left), we employ LM to extract structures and retrieve their corresponding contents (the content of pages are passages belonging to that page, and the content of tables is the markdown-formatted text). For questions on document content, we concatenate it with the currently retrieved context and prompt the LM to generate the next evidence to answer the question. By comparing the similarity between the candidate neighboring sentences and the generated passage, we determine the next passage node to traverse. Correspondingly, the candidate neighbors are updated for the next round of traversal.

reads: 'Alf Heiberg Clausen (born March 28, 1941) is an American film composer'. Subsequently, we prefix the currently retrieved-context (Node 1) with the question and prompt the LM to generate the next evidence required to approach the question closer. Because we augment the reasoning capability of the LM by instruction fine-tuning, it is expected to recognize the logical associations between the question and the currently retrieved context. Consequently, it can predict the subsequent passage that *maintains logical coherence, albeit may contain factual mistakes*, i.e., 'Alf Clausen (born April 16, 1941) is an American composer of film and television scores.' To rectify this potential factual mistake, we select nodes from the candidate neighbors that match the most with the LM generated passage, in this case, Node 4 'Alf Heiberg Clausen (born March 28, 1941) is an American film composer'. Since this passage sources directly from documents, it inherently ensures the validity of the information. Then we prompt LLMs along with the retrieved context Node 1 and 4 for the answer.

Additionally, for questions asking about document structures, we extract the document structure names and locate their corresponding structural nodes in the KG. For the table node, we retrieve its markdown formatted content while for the page node, we traverse its one-hop neighbor and obtain passages belonging to that page.

5 Experiment

In this section, we conduct experiments to verify the proposed knowledge graph prompting method (KGP) for MD-QA. In particular, we answer the following questions:

- **Q1 - Section 5.2**: How well does KGP perform MD-QA compared with existing baselines?
- **Q2 - Section 5.3-5.4**: How do the quality of the constructed KG and the LM-guided graph traverser impact the MD-QA performance?

Due to space limitations, we first briefly introduce our experimental setting in the following and leave comprehensive details in Supplementary A.1-A.2.

Table 1: Statistics of document collections and KGs constructed by TAGME average across questions.

Dataset	# Questions	# Passages	# Edges	Passage Avg. Length	KG Density
HotpotQA	500	715.22	70420.68	37.55	0.23
IIRC	477	1120.55	143136.17	37.24	0.20
WikiMHop	500	294.19	19235.15	37.24	0.27
MuSiQue	500	748.04	97931.28	38.56	0.29

5.1 Experimental Setting

5.1.1 Dataset

To explore the uncharted domain of MD-QA, we have created our own datasets to simulate real-world scenarios where users maintain folders containing various documents and pose questions to which the answers are only from certain parts of these documents. Specifically, we randomly sample questions from the development set of four existing datasets: HotpotQA [47], IIRC [10], 2WikiMQA [14], and MuSiQue [41]. For each question, we source documents from Wikipedia that encompass supporting facts pertaining to the question and combine them with randomly sampled negative documents to form the document collection. In addition to the content-based questions from these four existing datasets, we additionally incorporate the 'Comp' dataset, an internal company collection of real-world document-based questions. During its creation, humans were asked to read documents and pose questions according to document structures. We summarize the statistics of each dataset along with their KGs in Table 1 with more details in Supplementary.

5.1.2 Baselines

We compare KGP with retrieval baselines in three categories. The first category is the heuristic-based retriever including KNN with fuzzy search, TF-IDF [35], and BM25 [36]. The second category is the deep-learning-based retriever including DPR [22] and MDR [44]. The third category is the prompting-based retriever including IRCoT [40]. For KGP, we explore three variants based on their LM-guided graph traverser: KGP-T5, KGP-LLaMA, and KGP-MDR, using T5 (encoder-decoder), LLaMA (decoder only), and MDR (encoder only) respectively as f in Eq (1).

5.1.3 Evaluation Criteria

Following [53], we compute F1 and EM to compare the LLM's answer and the ground-truth one. As the predicted answer may not overlap with the ground-truth one, we additionally check the correctness of the answer following [9, 25, 55] by prompting the LLM. Moreover, for evaluating the quality of KGs in Figure 4, we adopt SF-EM (Supporting Fact Exact Matching) and precision from [44]. Given the subjective nature of the questions in Comp, we devise the metric, Structure Exact Matching (Struct-EM) to assess if retrieved contexts include the document structures mentioned in the question.

5.2 Performance Comparison on MD-QA

We compare the MD-QA performance of the proposed KGP-T5 and other baselines in Table 2. Firstly, the baseline 'None' and 'Golden' achieve the worst and the best performance because one provides no context and the other provides the golden context. All other baselines achieve the performance in-between because the retrieved context only covers the partial of the supporting facts. Our proposed methods KGP-T5 rank at the Top-1 except for the Golden baseline. The 2^{nd}-performing baseline MDR fine-tunes a RoBERTa-base encoder by predicting the next supporting fact based on the question and the already retrieved contexts [44]. This next-passage prediction pretext task equips the model with the reasoning capability of the knowledge across different passages and hence increases the quality of the retrieved contexts. The other deep-learning-based retriever DPR achieves much worse performance than MDR because it only fine-tunes the encoder by maximizing the similarity between the query and its supporting facts regardless of their sequential order, demonstrating the importance of understanding the logical order of different knowledge when solving MD-QA [44]. By comparing the MD-QA performance across different datasets, we find that all baselines perform better on HotpotQA than on IIRC. This is because questions in HotpotQA are generally simpler than in IIRC. Existing works [20] have shown that some questions can be easily answered by following shortcuts while questions in IIRC sometimes necessitate arithmetic skills to derive the numerical answers, e.g., 'How many years did the event last when Wingfield lost much of his fortune?'.

Moreover, without any particular design for document structures, no existing baselines can handle structural questions in Comp, e.g. 'What is the difference between Page 1 and Page 2' or 'In Table 3, which station has the highest average flow rate?'. Fortunately, with the constructed KG incorporating the structural nodes and our designed traversal algorithm retrieving structural contexts, our proposed method achieves 67% Struct-EM.

Table 2: MD-QA performances of different baselines. The best (runner-up) are in **bold** (underlined).

Method	Metric	None	KNN	TF-IDF	BM25	DPR	MDR	IRCoT	KGP-MDR	KGP-T5	KGP-LLaMA	Golden
HotpotQA	Acc	41.80	71.57	**76.64**	71.95	73.43	75.30	74.36	75.72	76.53	75.66	82.19
	EM	19.00	40.73	45.97	41.46	43.61	45.55	45.29	46.09	**46.51**	46.22	50.20
	F1	30.50	57.97	64.64	59.73	62.11	65.16	64.12	65.77	**66.77**	66.31	71.06
IIRC	Acc	19.50	43.82	47.47	41.93	48.11	**50.84**	49.78	49.58	48.28	49.57	62.68
	EM	8.60	25.15	27.22	23.48	26.89	27.52	**27.73**	29.32	26.94	28.09	35.64
	F1	13.17	37.24	40.80	35.55	41.85	**43.47**	41.65	43.21	41.54	42.56	54.76
2WikiMQA	Acc	44.40	52.40	58.40	55.80	62.40	63.00	61.81	60.94	**63.50**	62.45	72.60
	EM	18.60	31.20	34.60	30.80	35.60	36.00	37.75	37.22	**39.80**	37.55	40.20
	F1	25.07	42.13	44.50	40.55	51.10	52.44	50.17	51.29	**53.50**	52.45	59.69
MuSiQue	Acc	30.40	44.70	44.40	44.47	44.27	48.39	45.14	51.22	**50.92**	50.81	57.00
	EM	4.60	18.86	21.59	21.11	20.32	23.49	22.46	27.76	**27.90**	26.72	30.60
	F1	10.58	30.04	32.50	31.15	31.64	37.03	34.21	41.11	**41.19**	40.01	47.75
Comp	Acc	0.00	–	–	–	–	–	–	67.00			100.00
Rank	w Comp	10.54	9.00	6.69	8.92	7.23	4.54	5.61	3.23	3.69	3.69	1.00
	w/o Comp	11.00	9.33	6.83	9.25	7.42	4.50	5.66	3.33	3.83	3.83	1.00

None: no passages but only the question is provided. Golden: supporting facts are provided along with the question.

5.3 Impact of the LM-guided Graph Traverser

Here we study the influence of using different LMs in guiding graph traversers over TAGME-constructed KG on MD-QA performance. Specifically, we compare the guidance by no LM (w/o LM), LLaMA, T5, and MDR in Table 3. Because TAGME w/o LM only blindly traverses in the KG without any guidance from LM, it unavoidably collects irrelevant passages and hence achieves the worst performance than others with LM guidance. This aligns with our previous observation on the generally low precision in Figure 4 and further demonstrates the necessity of using LMs to guide the graph traversal. Interestingly, we find that KGP-T5 performs better than LLaMA even though the parameters of LLaMA (7B) are more than the ones with T5 (0.7B). We hypothesize this is because models with larger amounts of parameters require more training data to avoid over-fitting.

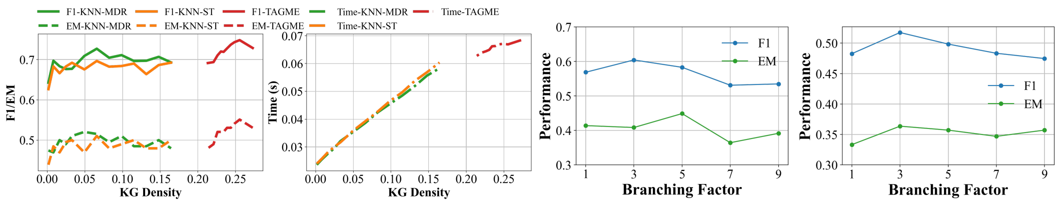
Table 3: Statistics of document collections and KGs by TAGME average across all questions.

Dataset		HotpotQA			IIRC			2WikiMQA			MuSiQue		
Metric		Acc	EM	F1	Acc	EM	F1	Acc	EM	F1	Acc	EM	F1
TAGME	w/o LM	73.52	43.79	63.14	46.30	27.70	41.43	58.12	35.07	45.95	44.67	21.93	32.90
	LLaMA	75.66	46.22	66.31	49.57	28.09	42.56	62.45	37.55	52.45	50.81	26.72	40.01
	T5	**76.53**	**46.51**	66.77	48.28	26.94	41.54	**63.50**	**39.80**	**53.50**	50.92	**27.90**	**41.19**
	MDR	75.72	46.09	65.77	**49.58**	**29.32**	**43.21**	60.94	37.22	51.29	**51.22**	27.76	41.11

5.4 Impact of the Constructed Graph and Branching Factor in Graph Traversal

Here we construct KGs with varying densities by changing the hyperparameters of TF-IDF/KNN-ST/KNN-MDR/TAGME and studying its impact on the performance and the neighbor matching time of MD-QA using KGP-T5. Since the LM-guided graph traverser selects the next node to visit from neighbors of already visited nodes, the chance that it hits the supporting facts increases as the number of neighbors increases. In contrast, the neighborhood matching efficiency decreases as the candidate pool, i.e., $\mathcal{N}_j$ in Eq (1), becomes larger. As evidenced in Figure 6(a), we observe a similar trend, i.e., as the KG density increases, the F1/EM increases and then stays stable while the latency for selecting the most promising neighbors to visit next also increases. KNN-MDR performs better than KNN-ST when the density of the two constructed KGs is the same. This is because the encoder in KNN-ST is pre-trained on wide-spectrum datasets while the encoder in MDR is specifically pre-trained on the HotpotQA dataset by the pretext task of predicting the next supporting facts. Therefore, the embedding similarity and the corresponding neighbor relations better reflect the logical associations among different passages, which aligns with the better constructed KG by KNN-MDR than the KG by KNN-ST in Figure 4. Compared with KNN-MDR/ST, TAGME delivers superior performance at the cost of increasing latency since the generated KG by TAGME is denser than KGs by KNN-ST/MDR.

Furthermore, we perform the sensitivity analysis of the branching factor (the number of nodes selected from candidate neighbors to visit next). In Figure 6(b) the performance first increases as the branching factor increases because more passage nodes selected from the candidate neighbors lead to more reasoning paths to reach the final answer. However, as we fix the context budget to ensure fair comparison (i.e., the total number of passages we are allowed to retrieve for each question is the same across all baselines), the performance declines as the branching factor increases because the number of initial seeding nodes diminishes, leading to reduced coverage of the KG.



(a) Impact of Constructed KG on HotpotQA

(b) Impact of Branching Factor on 2WikiMQA (Left) and MuSiQue (Right)

Figure 6: **(a)**: The performance/latency increases as the KG density increases. The results are averaged across 100 randomly sampled questions on HotpotQA. **(b)**: The performance first increases and then decreases as the branching factor increases. The results are averaged across 100 sampled questions on 2WikiMQA and MuSiQue.

5.5 Visualizing the Reasoning-and-Retrieving Process of LM-guided Graph Traverser

In this section, we visualize the KG-LLaMA's reasoning-and-retrieving process in retrieving relevant context for MD-QA. Due to space limitation, for each question, we visualize the top-3 sentence nodes visited at 1-hop along with their generated evidence from LLaMA that required further to approach the answer. Based on the generated evidence, we retrieve the top-2 sentence nodes from the candidate neighbor queue. For each retrieved sentence node, we also visualize its ranking score given by TF-IDF. We can see that the first retrieved evidence suggests the academy (USMMA) where Joseph D. Stewart was appointed Superintendent. Based on that, the LLM can then rationalize correctly and suggest the next passage should include information indicating the location of USMMA, which is used further to retrieve the ground-truth passage including that information.

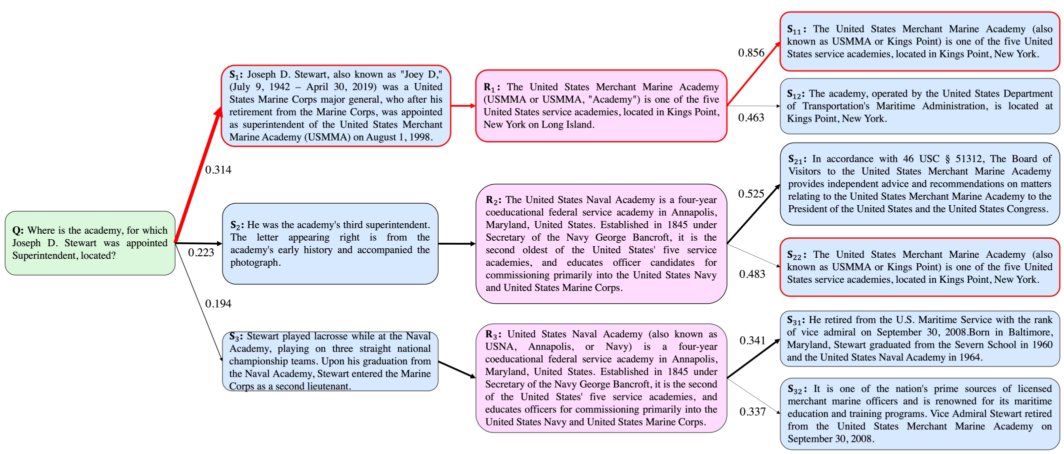


Figure 7: Visualizing the graph traversal over MD-QA.

6 Conclusion

Answering multi-document questions demands knowledge reasoning and retrieving from different documents across various modalities, presenting challenges for applying the paradigm of 'pre-train, prompt and predict' with LLMs. Recognizing that the logical associations among passages and structural relations within the documents can be unified into a graphical representation, we propose a Knowledge Graph Prompting method (KGP) for aiding LLMs in MD-QA. The KGP constructs KGs from documents with nodes depicting sentences or document structures and edges denoting their lexical/semantic similarity or structural relations. Since the constructed KGs may contain irrelevant neighbor information, we further design an LM-guided graph traverser that selectively visits the most promising node in approaching the question. In the future, we plan to investigate the capability of LLMs in understanding graph topology and explore the potential of fine-tuning/prompting LLMs to encode complex topological signals hidden in the graph.

References

[1] Rami Aly, Zhijiang Guo, Michael Schlichtkrull, James Thorne, Andreas Vlachos, Christos Christodoulopoulos, Oana Cocarascu, and Arpit Mittal. Feverous: Fact extraction and verification over unstructured and structured information. *arXiv preprint arXiv:2106.05707*, 2021.

[2] Akari Asai, Kazuma Hashimoto, Hannaneh Hajishirzi, Richard Socher, and Caiming Xiong. Learning to retrieve reasoning paths over wikipedia graph for question answering. *arXiv preprint arXiv:1911.10470*, 2019.

[3] Mark Bolino, David Long, and William Turnley. Impression management in organizations: Critical questions, answers, and areas for future research. *Annual Review of Organizational Psychology and Organizational Behavior*, 3:377–406, 2016.

[4] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[5] Avi Caciularu, Matthew E Peters, Jacob Goldberger, Ido Dagan, and Arman Cohan. Peek across: Improving multi-document modeling via cross-document question-answering. *arXiv preprint arXiv:2305.15387*, 2023.

[6] Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. Reading wikipedia to answer open-domain questions. *arXiv preprint arXiv:1704.00051*, 2017.

[7] Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416*, 2022.

[8] Junnan Dong, Qinggang Zhang, Xiao Huang, Keyu Duan, Qiaoyu Tan, and Zhimeng Jiang. Hierarchy-aware multi-hop question answering over knowledge graphs. In *Proceedings of the ACM Web Conference 2023*, pages 2519–2527, 2023.

[9] Yann Dubois, Xuechen Li, Rohan Taori, Tianyi Zhang, Ishaan Gulrajani, Jimmy Ba, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. Alpacafarm: A simulation framework for methods that learn from human feedback. *arXiv preprint arXiv:2305.14387*, 2023.

[10] James Ferguson, Matt Gardner, Hannaneh Hajishirzi, Tushar Khot, and Pradeep Dasigi. Iirc: A dataset of incomplete information reading comprehension questions. *arXiv preprint arXiv:2011.07127*, 2020.

[11] Paolo Ferragina and Ugo Scaiella. Tagme: on-the-fly annotation of short text fragments (by wikipedia entities). In *Proceedings of the 19th ACM international conference on Information and knowledge management*, pages 1625–1628, 2010.

[12] Aristides Gionis, Piotr Indyk, Rajeev Motwani, et al. Similarity search in high dimensions via hashing. In *Vldb*, volume 99, pages 518–529, 1999.

[13] Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A Smith. Don't stop pretraining: Adapt language models to domains and tasks. *arXiv preprint arXiv:2004.10964*, 2020.

[14] Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. Constructing a multi-hop qa dataset for comprehensive evaluation of reasoning steps. *arXiv preprint arXiv:2011.01060*, 2020.

[15] Johannes Hoffart, Fabian M Suchanek, Klaus Berberich, and Gerhard Weikum. Yago2: A spatially and temporally enhanced knowledge base from wikipedia. *Artificial intelligence*, 194:28–61, 2013.

[16] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

[17] Ziniu Hu, Yuxiao Dong, Kuansan Wang, and Yizhou Sun. Heterogeneous graph transformer. In *Proceedings of the web conference 2020*, pages 2704–2710, 2020.

[18] Xiaofeng Huang, Jixin Zhang, Zisang Xu, Lu Ou, and Jianbin Tong. A knowledge graph based question answering method for medical domain. *PeerJ Computer Science*, 7:e667, 2021.

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

[20] Yichen Jiang and Mohit Bansal. Avoiding reasoning shortcuts: Adversarial evaluation, training, and model development for multi-hop qa. *arXiv preprint arXiv:1906.07132*, 2019.

[21] Mingxuan Ju, Wenhao Yu, Tong Zhao, Chuxu Zhang, and Yanfang Ye. Grape: Knowledge graph enhanced passage reader for open-domain question answering. *arXiv preprint arXiv:2210.02933*, 2022.

[22] Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. *arXiv preprint arXiv:2004.04906*, 2020.

[23] Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, et al. Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic web*, 6(2):167–195, 2015.

[24] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys*, 55(9):1–35, 2023.

[25] Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. Gpteval: Nlg evaluation using gpt-4 with better human alignment. *arXiv preprint arXiv:2303.16634*, 2023.

[26] Yuning Mao, Pengcheng He, Xiaodong Liu, Yelong Shen, Jianfeng Gao, Jiawei Han, and Weizhu Chen. Rider: Reader-guided passage reranking for open-domain question answering. *arXiv preprint arXiv:2101.00294*, 2021.

[27] Minesh Mathew, Dimosthenis Karatzas, and CV Jawahar. Docvqa: A dataset for vqa on document images. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 2200–2209, 2021.

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

[29] Kyosuke Nishida, Itsumi Saito, Atsushi Otsuka, Hisako Asano, and Junji Tomita. Retrieve-and-read: Multi-task learning of information retrieval and reading comprehension. In *Proceedings of the 27th ACM international conference on information and knowledge management*, pages 647–656, 2018.

[30] Hariom A Pandya and Brijesh S Bhatt. Question answering survey: Directions, challenges, datasets, evaluation matrices. *arXiv preprint arXiv:2112.03572*, 2021.

[31] Jayr Pereira, Robson Fidalgo, Roberto Lotufo, and Rodrigo Nogueira. Visconde: Multi-document qa with gpt-3 and neural reranking. In *European Conference on Information Retrieval*, pages 534–543. Springer, 2023.

[32] Chengwei Qin, Aston Zhang, Zhuosheng Zhang, Jiaao Chen, Michihiro Yasunaga, and Diyi Yang. Is chatgpt a general-purpose natural language processing task solver? *arXiv preprint arXiv:2302.06476*, 2023.

[33] Ao Qu, Yu Wang, Yue Hu, Yanbing Wang, and Hiba Baroud. A data-integration analysis on road emissions and traffic patterns. In *Driving Scientific and Engineering Discoveries Through the Convergence of HPC, Big Data and AI: 17th Smoky Mountains Computational Sciences and Engineering Conference, SMC 2020, Oak Ridge, TN, USA, August 26-28, 2020, Revised Selected Papers 17*, pages 503–517. Springer, 2020.

[34] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551, 2020.

[35] Juan Ramos et al. Using tf-idf to determine word relevance in document queries. In *Proceedings of the first instructional conference on machine learning*, volume 242, pages 29–48. Citeseer, 2003.

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

[37] Girolamo Tessuto. Legal problem question answer genre across jurisdictions and cultures. *English for Specific Purposes*, 30(4):298–309, 2011.

[38] James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. Fever: a large-scale dataset for fact extraction and verification. *arXiv preprint arXiv:1803.05355*, 2018.

[39] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[40] Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. *arXiv preprint arXiv:2212.10509*, 2022.

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

[42] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35:24824–24837, 2022.

[43] Xuansheng Wu, Kaixiong Zhou, Mingchen Sun, Xin Wang, and Ninghao Liu. A survey of graph prompting methods: techniques, applications, and challenges. *arXiv preprint arXiv:2303.07275*, 2023.

[44] Wenhan Xiong, Xiang Lorraine Li, Srini Iyer, Jingfei Du, Patrick Lewis, William Yang Wang, Yashar Mehdad, Wen-tau Yih, Sebastian Riedel, Douwe Kiela, et al. Answering complex open-domain questions with multi-hop dense retrieval. *arXiv preprint arXiv:2009.12756*, 2020.

[45] Yang Xu, Yiheng Xu, Tengchao Lv, Lei Cui, Furu Wei, Guoxin Wang, Yijuan Lu, Dinei Florencio, Cha Zhang, Wanxiang Che, et al. Layoutlmv2: Multi-modal pre-training for visually-rich document understanding. *arXiv preprint arXiv:2012.14740*, 2020.

[46] Jingfeng Yang, Hongye Jin, Ruixiang Tang, Xiaotian Han, Qizhang Feng, Haoming Jiang, Bing Yin, and Xia Hu. Harnessing the power of llms in practice: A survey on chatgpt and beyond. *arXiv preprint arXiv:2304.13712*, 2023.

[47] Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W Cohen, Ruslan Salakhutdinov, and Christopher D Manning. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. *arXiv preprint arXiv:1809.09600*, 2018.

[48] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *arXiv preprint arXiv:2305.10601*, 2023.

[49] Yao Yao, Zuchao Li, and Hai Zhao. Beyond chain-of-thought, effective graph-of-thought reasoning in large language models. *arXiv preprint arXiv:2305.16582*, 2023.

[50] Michihiro Yasunaga, Antoine Bosselut, Hongyu Ren, Xikun Zhang, Christopher D Manning, Percy S Liang, and Jure Leskovec. Deep bidirectional language-knowledge graph pretraining. *Advances in Neural Information Processing Systems*, 35:37309–37323, 2022.

[51] Michihiro Yasunaga, Hongyu Ren, Antoine Bosselut, Percy Liang, and Jure Leskovec. Qa-gnn: Reasoning with language models and knowledge graphs for question answering. *arXiv preprint arXiv:2104.06378*, 2021.

[52] Semih Yavuz, Kazuma Hashimoto, Yingbo Zhou, Nitish Shirish Keskar, and Caiming Xiong. Modeling multi-hop question answering as single sequence prediction. *arXiv preprint arXiv:2205.09226*, 2022.

[53] Wenhao Yu, Dan Iter, Shuohang Wang, Yichong Xu, Mingxuan Ju, Soumya Sanyal, Chenguang Zhu, Michael Zeng, and Meng Jiang. Generate rather than retrieve: Large language models are strong context generators. *arXiv preprint arXiv:2209.10063*, 2022.

[54] Yue Yu, Yuchen Zhuang, Jieyu Zhang, Yu Meng, Alexander Ratner, Ranjay Krishna, Jiaming Shen, and Chao Zhang. Large language model as attributed training data generator: A tale of diversity and bias. *arXiv preprint arXiv:2306.15895*, 2023.

[55] Yanzhe Zhang, Ruiyi Zhang, Jiuxiang Gu, Yufan Zhou, Nedim Lipka, Diyi Yang, and Tong Sun. Llavar: Enhanced visual instruction tuning for text-rich image understanding. *arXiv preprint arXiv:2306.17107*, 2023.

[56] Fengbin Zhu, Wenqiang Lei, Chao Wang, Jianming Zheng, Soujanya Poria, and Tat-Seng Chua. Retrieving and reading: A comprehensive survey on open-domain question answering. *arXiv preprint arXiv:2101.00774*, 2021.

A Supplementary

A.1 Dataset Collection

This section introduces the collection of datasets used for the experiments conducted in this paper.

A.1.1 Document Set Collection and Procession

As no previous works focus on MD-QA, we create our own datasets to simulate real-world scenarios where users maintain folders containing various documents and pose questions to which the answers are only from certain parts of these documents. To imitate this scenario, we randomly sample questions from the development set of existing datasets: HotpotQA/IIRC/2WikiMQA/MuSiQue, and then for each specific question, we fetch documents from Wikipedia that encompass supporting facts pertaining to the question [4] and term these documents as golden documents. Then we randomly sample negative documents from Wikipedia and pair them with golden documents to constitute the document collection. For each document in the collected document set, we split it into multiple passages with the default passage length being the sentence length. As questions from these existing datasets are only focused on document contents, we additionally incorporate the 'Comp' dataset, an internal company collection of real-world questions focusing on document structures.

A.1.2 Knowledge Graph Construction

We construct a knowledge graph for each question and its corresponding collection of documents. For datasets where the questions are from Wikipedia: HotpotQA, IIRC, WikiMHop, and Musique, we only have passage nodes since answering questions in these datasets does not require information about document structures. For the Comp dataset, in addition to passage nodes, we apply ExtractAPI to obtain the page and table information so that the constructed KG also has pages/tables as nodes. For all of these datasets, we add edges following Section 3. Table 4 summarizes the average statistics of the document collections across all questions with their corresponding KGs. Except for Comp, we plan to release the code for collecting the documents and constructing the KGs upon publication.

Table 4: Statistics of document collections and their corresponding knowledge graph used in Table 2 and 3 average across all questions.

Dataset	#Docs	#Questions	#Passages	#Edges	Passage Avg. Length	KG Density
HotpotQA	12	500	715.22	70420.68	37.55	0.23
IIRC	12	477	1120.55	143136.17	37.24	0.20
2WikiMQA	12	500	294.19	19235.15	37.24	0.27
MuSiQue	12	500	748.04	97931.28	38.55	0.29

For Comp, due to privacy concerns, we omit the data statistics but only provide some question examples, e.g., 'How many more classical students in Table 2 had the mixed teaching style versus the classical teaching style?' or 'Can you give me a simple summary about page 5?'.

A.1.3 Sequential Data Collection

Training MDR [44] requires rearranging supporting facts into the sequential order that progressively approaches the answer. To fulfill this requirement, we directly follow MDR and use the pre-processed HotpotQA data from the GitHub Repository[5] to train the encoder and apply it to other datasets that do not provide the sequential order of supporting facts. For instruction fine-tuning LLaMA, we still use the above HotpotQA data and rearrange it into the instruction-input-output format and use the instruction 'What evidence do we need to answer the question given the current evidence'. We present one example in Listing 1. For T5-large, we use the same input-output but prefix the reasoning instruction to the input following the original T5 input format [34].

[4]The HotpotQA/IIRC/2WikiMQA/Musique datasets already have the supporting facts for each question.

[5]https://github.com/facebookresearch/multihop_dense_retrieval/tree/main

A.2 Experiment Details

A.2.1 Training DPR and MDR

For training DPR [22], we pair each question with its supporting facts as its positive passages, and some randomly sampled negative passages as its negative passages. For training MDR [44], as each question in HotpotQA only requires 2 supporting facts to derive the answer, we set the first supporting fact as the positive pair for each question. Further, we concatenate this question and the first supporting fact to form a new question and for this newly-formed question, we set the second supporting fact as its positive pair. For both the original question and the concatenated one, we randomly sample other passages as the negative pair. Following [44, 22], we use RoBERTa-base as the default encoder and search hyperparameters for them as follows: hidden dimension 768, max context length $\{128, 256, 350\}$, batch size $\{128, 256, 512\}$, epoch 50, warmup steps 300, learning rate $2e-5$, gradient clipping range 2.

A.2.2 Instruction Fine-tuning LLaMA[6] and T5-Large[7]

We fine-tune LLaMA using instruction data in Listing 1. Due to the computational limitation, we choose LLaMA-7B and use LoRA [16]. For fine-tuning T5-Large, we use the same instruction data except that we remove the instruction but only prefix the reasoning instruction to the input [34]. We use the default hyperparameters from their original GitHub repository to fine-tune these two LLMs.

A.2.3 Prompting LLMs for MD-QA - Table 2 and 3

Following [40], we randomly select questions from the development set for reporting the performance. To ensure a fair comparison, we set the number of retrieved passages to 30 across all baselines and use ChatGPT as the downstream LLM for reading the retrieved passages and generating the answer. We summarize the key implementation details for each baseline as follows:

- **KNN**: We employ the sentence-transformer variant 'multi-qa-MiniLM-L6-cos-v1' to obtain passage embeddings as it has been trained on 215M (question, answer) pairs from diverse sources. Then we select the top-15 passages according to the embedding similarity and the top-15 passages according to the fuzzy matching[8].
- **MDR**: We use beam search with the inner product as the scoring function to rank passages. We limit the search depth to 2 as answering questions in HotpotQA requires at most 2-hop reasoning steps [44]. We set the number of passages to be 15 in the first-hop retrieval and for each of these passages, we further retrieve 3 more passages in the second round, which in total generates 45 passage pairs. Then we rank these 45 passage pairs by the product of the scores between the first-hop and the second-hop retrieval and select the top 30 ones as the final context.
- **IRCoT**: Instead of directly employing the original IRCoT code [40], we modify it based on our problem setting. The first reason is that passages to be retrieved in IRCoT [40] are the pre-processed Wikipedia Corpus and do not cover the whole contents of Wikipedia documents, which thereby is not aligned with our MD-QA setting. The second reason is that the question-answering reader employed in IRCoT requires running on A100-80G GPU, which is not affordable on our side. Therefore, we modify the IRCoT by replacing the question reader with the ChatGPT and using our pre-processed Wikipedia document collections as introduced in Section A.1. For the prompt used in the reasoning step, we select 2 examples from 'gold_with_2_distractors_context' for the demonstration purpose. We iteratively select top-5 passages based on the generated reason from LLM along with their document titles and add them to the retrieved context until hitting the prefix budget. For the prompt used in the reading step, we use exactly the same prompt as other baselines as we find it empirically leads to better performance than the original one used in IRCoT [40].
- **KGP-T5/LLaMA/MDR**: We use T5-large/LLaMA-7B/MDR as the LM to guide the graph traversal respectively. For content-based questions, similar to MDR, we perform a 2-hop retrieval but for each hop, we only search the node to visit next from neighbor candidates. In the 1$^{\text{st}}$-hop retrieval, we select 10 passages and in 2$^{\text{nd}}$-hop retrieval, we select 3 passages, which totally forms 30 reasoning paths. Note that passages in the 1$^{\text{st}}$-hop retrieval are allowed to overlap with the ones in the 2$^{\text{nd}}$-hop retrieval. For structural-based questions, we first use ChatGPT to extract page/table

[6] `https://github.com/Lightning-AI/lit-llama`

[7] `https://shivanandroy.com/fine-tune-t5-transformer-with-pytorch/`

[8] We use Levenshtein-distance to measure the lexical distance between two passages.

structures and then fetch relevant contents in those structures. Future work could explore how to pre-train a structural extraction model to obtain document structures.

- **KGP w/o LM**: We remove the LM-guided graph traversal but select passages nodes based on their TF-IDF similarity to the given question.

Note that we put the prompt template for running all the above baselines in Section A.9.

A.3 Algorithm and Complexity for KGP

Here we present the algorithm for our proposed knowledge graph prompting (KGP) method for MD-QA. Given a question, we first apply LLM to classify whether the question is asking about the document structure or document content. If the question focuses on the document structure, we extract the structural keywords such as Page or Table, and retrieve the content in the corresponding structural nodes in KG. If the question focuses on the document content, we follow the step according to Algorithm 1. Specifically, we first initialize seeding passages $\mathcal{V}^s$ and the reasoning path queue $\mathcal{P}$ by TF-IDF search. Then for each seeding passage $v_i \in \mathcal{V}^s$, we add its neighboring passage nodes $\mathcal{N}_i$ into the candidate neighbor queue $\mathcal{C}$. (lines 1-4) After that, we iteratively pop out the leftmost reasoning path/candidate neighborhood $\mathcal{P}_i/\mathcal{C}_i$ from $\mathcal{P}/\mathcal{C}$ and employ the fine-tuned LM-guided graph traverser to rank the popped out neighbors in $\mathcal{C}_i$ by Eq. (1) (lines 5-7). Last, we select top-k passage nodes $\mathcal{V}_i'$ from $\mathcal{C}_i$ to visit next based on their rank and correspondingly update the candidate neighbor queue/reasoning path queue (lines 8-13). The above process terminates when either the candidate neighbor queue becomes empty or the prefixed budget K for the retrieved passages is met.

Algorithm 1: Knowledge Graph Prompting Method for Questions on Document Contents

---

Input: A question q over a set of documents $\mathcal{D}$, the constructed knowledge Graph $G = \{\mathcal{V}, \mathcal{E}, \mathcal{X}\}$ over $\mathcal{D}$, the fine-tuned LLM-guided graph traversal f_{GT}, the preset context budget K, the initial TF-IDF search function g.

1 Initialize seed passages $\mathcal{V}^s = g(\mathcal{V}, \mathcal{X}, q)$
2 Initialize the retrieved passage queue $\mathcal{P} = [\{v_i\} | v_i \in \mathcal{V}^s]$
3 Initialize the candidate neighbor queue $\mathcal{C} = [\mathcal{N}_i | v_i \in \mathcal{V}^s]$
4 Initialize the retrieved passage counter $k = \sum_{\mathcal{P}_i \in \mathcal{P}} |\mathcal{P}_i|$
5 **while** *queue $\mathcal{P}$ and queue $\mathcal{C}$ are not empty* **do**
6     $\mathcal{P}_i \leftarrow \mathcal{P}$.dequeue(), $\mathcal{C}_i \leftarrow \mathcal{C}$.dequeue()
7     $\mathcal{V}_i' =$ Graph Traversal$(\{q\} \cup \mathcal{P}_i, \mathcal{C}_i, k)$ by Eq (1)
8   **for** $v \in \mathcal{V}_i'$ **do**
9         $\mathcal{P}$.enqueue$(\mathcal{P}_i \cup \{v\})$
10         $\mathcal{C}$.enqueue$(\mathcal{N}_v)$
11         $k \leftarrow k + 1$
12   **if** $k > K$ **then**
13   **Terminate**
14 **return** *Retrieved Passage Queue $\mathcal{P}$*

Since our algorithm can be essentially deemed as the combination of the neighborhood ranking by Eq. (1) and the breadth-first-search. The time complexity would be the multiplication between the time of bread-first-search $\mathcal{O}(|\mathcal{V}| + |\mathcal{E}|)$ and the time of neighborhood ranking $\mathcal{O}(|\mathcal{N}|\gamma) = \mathcal{O}(\hat{d}\gamma)$ where γ is the time for computing the embedding similarity between a specific neighbor passage and the retrieved reasoning path and $\hat{d}$ is the average degree of the KG. Therefore the final time complexity would be $\mathcal{O}((|\mathcal{V}| + |\mathcal{E}|)\hat{d}\gamma)$, which is in-between the linear and quadratic to the size of the graph. As users typically maintain 10-100 documents, correspondingly the number of nodes in the constructed KG would be around 1,000-10,000 (according to Table 4, a collection of 12 documents have roughly 200-1000 passage nodes), which is affordable even with the quadratic time complexity. Moreover, we can apply advanced techniques to further reduce the time complexity for neighborhood ranking, such as KD-tree [33] and LSH [12].

For space complexity, it takes $\mathcal{O}(|\mathcal{V}|(\alpha + \beta))$ to maintain the constructed KG on the fly where α is the average space for saving the passage embedding vector while β is the average space for saving the textual information of that passage. Although our constructed KG treats passages as nodes, which cannot scale very well when the graph is extremely large, the total number of documents a user maintains in a folder is typically around 10-100, which is still affordable.

A.4 Markdown-Formatted Table

Figure 8 demonstrates that by sending Tables in the markdown format, ChatGPT can successfully understand their content and perform information retrieval based on the given questions. However, we do observe that such a markdown-formatted solution is not feasible for the long table due to the input token limitation of ChatGPT, we plan to explore the solution using SQL as the prompt content or modeling the Table as the grid graph to solve the issue in the future.

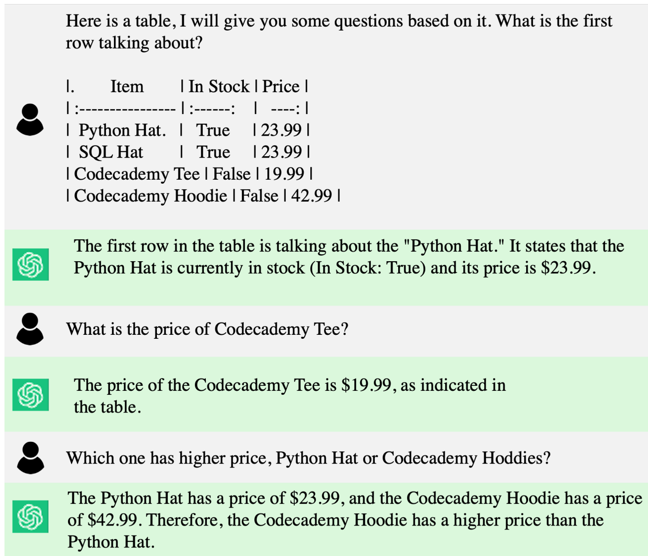


Figure 8: An example demonstrating that ChatGPT can understand table in the markdown format.

A.5 Knowledge Graph Construction Comparison

Table 5 compares different knowledge graph construction methods and their pros and cons.

Table 5: Systematically Comparison among existing and our proposed Knowledge Graphs.

KG	Node	Edge	Domain	Constructor	Scalability	Hyperparameters	Advantage	Disadvantage
TAGME	Passage	Common Wikipedia Entity	Wikipedia	/	No	Prior Threshold	Effectively Identify Wikipedia Entities	Low efficiency for Entity Identification Narrow Domain Application
TF-IDF	Passage	Common Keyword	General	/	No	# Keywords	No Domain Limitation	Common keywords irrelevant to question
KNN-ST	Passage	Semantic Similarity	General	Sentence Transformer	No	# Neighbors	No Domain Limitation	Semantic Similarity irrelevant to question
KNN-MDR	Passage	Semantic Similarity	General	MDR	No	# Neighbors	Encoding the logical association for QA	Require logically ordered supporting facts to pre-train the model
Knowledge Base	Entity	Relationship	Specific	Human	Yes	/	Powerful in encoding factual information	Relation Extraction is non-trivial Domain Specific

- **TAGME**: TAGME [11] is very effective in extracting Wikipedia Entities from a passage despite the low efficiency. In our graph construction, it usually takes more than 8 hours to extract entities of all passages for even just 12 Wikipedia documents. Even after we apply parallel processing, it still takes more than 2 hours. In addition, it can only handle entities mentioned in the existing Wikipedia system and hence cannot generalize to documents from other domains.
- **TF-IDF and KNN-ST**: Although there is no domain limitation, it is hard to guarantee the extracted keywords or the embedding semantic similarity can precisely encode the relationships that are desired for answering the given question between any two passages. We empirically find TF-IDF is more likely to extract meaningless keywords even after removing supporting verbs and articles.

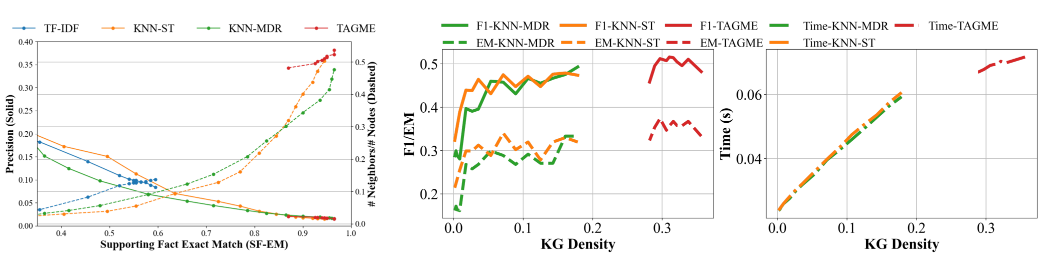
- **KNN-MDR**: Since KNN-MDR pre-trains the sentence encoder by predicting the next supporting passage given already-retrieved passages, the embedding similarity between two passages is more likely to encode necessary logical associations required for MD-QA. However, the main bottleneck here is how to obtain the logically ordered supporting facts that can progressively reach the answer. Obtaining these sequential data is non-trivial and usually requires a large number of human resources for well-curated annotation.
- **Existing Knowledge Base**: One common approach in the literature is to use existing knowledge bases or extract subgraphs from them for specific tasks [8, 50, 51]. Because the factual information is characterized as a triplet consisting of two entity nodes and their relationship, it is very powerful in encoding factual information/commonsense knowledge and also avoids the scalability issue (since two different passages might share the same entity). Despite its potency and ease of use, constructing this type of KGs demands meticulously designed relation extractors, which is still deemed a challenging task in the literature. Recent research has explored using LLMs for relation extraction. However, with increasing document numbers, using non-open-sourced LLMs can become prohibitively expensive. A potential solution is fine-tuning an open-sourced LLM specifically for relation extraction. Detailed discussion on this is beyond the scope of this study and is thus omitted.

To put it in a nutshell, there's no one-size-fits-all method for KG construction. Our paper offers an in-depth analysis of the proposed KG construction methods alongside other existing ones. The best approach often depends on the specific use case. For broad domains containing general factual information, tools like 'TAGME' or 'Knowledge Base' might be apt. However, for more niche or sensitive areas, methods like TF-IDF/KNN-ST are more appropriate. In certain situations, gathering domain-specific data and pre-training encoders is the most effective way to build the KG.

A.6 Additional Results and Discussions

A.6.1 Quality of KG on MuSiQue

Similar to the setting used for Figure 4, we change the hyperparameters to construct KGs for each question in MuSiQue with varying levels of sparsity and measure how much percentage of the supporting facts are covered by neighbors of the seeding passages that are initially retrieved by TF-IDF. The general trend in Figure 9(a) is similar to the one in Figure 4, i.e., as the graph becomes denser, the precision decreases while the SF-EM increases. However, on MuSiQue, KNN-MDR achieves the worst trade-off between Precision and SF-EM compared with KNN-ST and TF-IDF. This is because our KNN-MDR is pre-trained on HotpotQA and due to the distribution shift from HotpotQA to MuSiQue, it is expected for the graph constructed with KNN-MDR to have less quality. Note that although here KNN-ST leads to a better KG than KNN-MDR, it does not mean the KNN baseline in Table 2 should perform better than MDR because the baseline name only refers to the retrieval method while the name in this figure refers to the KG construction method.



(a) Impact of Constructed KG on MuSiQue **(b) Impact of Constructed KG on MuSiQue**

Figure 9: **(a)**: Quality of constructed KGs with different methods on MuSiQue. **TF-IDF**: lexical similarity based on common keywords extracted by TF-IDF. **KNN-ST**: KNN graph constructed based semantic similarity of embeddings from sentence-transformer; **KNN-MDR**: KNN graph constructed based on semantic similarity of embeddings from the pre-trained MDR [44]; **TAGME**: graph constructed based on whether two passages share common Wikipedia entity mentions. **(b)**: The performance/latency increases as the KG density increases. The results are averaged across 100 randomly sampled questions on MuSiQue.

A.6.2 The impact of KG on MuSiQue

Similar to the setting used for Figure 6, we compare the MD-QA performance for KGP-T5 using TAGME-based KG with different levels of density. Similar to Figure 6, here we also observe that as the KG becomes denser, the MD-QA performance increases while the time for the next node search increases. However, on MuSiQue, in most cases, KNN-ST achieves better F1/EM than KNN-MDR, which exactly aligns with the constructed KG quality observed in Figure 9, i.e., KNN-ST achieves better Precision/SF-EM trade-off than KNN-MDR on MuSiQue.

A.7 Case study on Structural/Content Questions

In this section, we conduct six MD-QA case studies using our self-designed user interface coupled with the proposed method on the backend. Examples include two table-based QA (Figure 10-11), one page-based QA (Figure 12), one single-document content-based QA (Figure 13) and two multi-document content-based QA (Figure 14-17). In our designed interface, we can upload documents we are interested in reading and the model on the backend will split each of them into multiple passages. In addition, on the left side, we can ask questions related to the currently uploaded documents. By clicking the button 'SUBMIT', the question would be sent to the model on the backend and it retrieves relevant context and arranges them as the prompt to get the answer from ChatGPT. In the figures below, we can see our system can understand the Table/Page questions and also questions requiring knowledge across multiple documents.

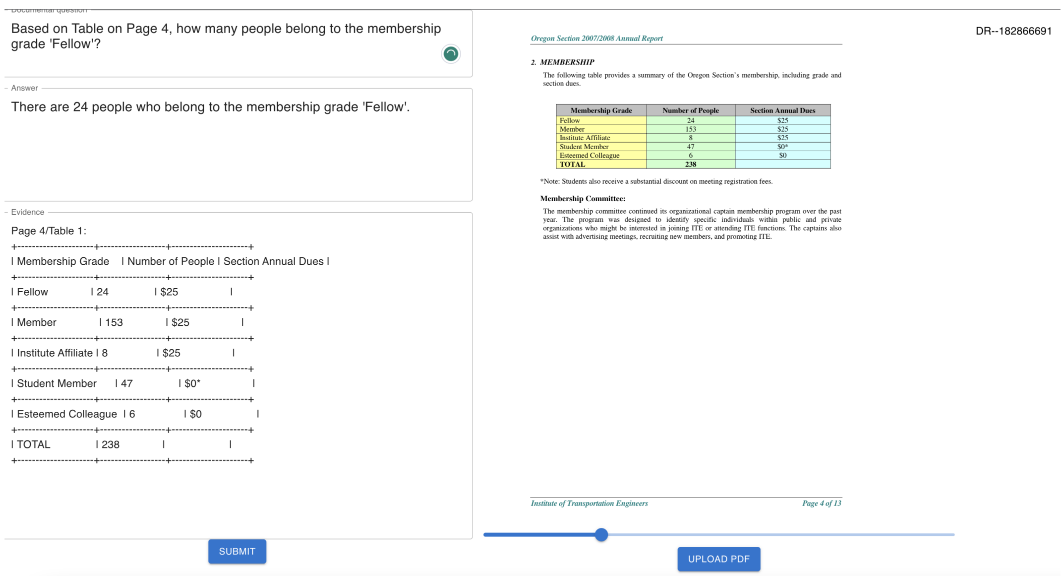


Figure 10: Table QA asking for the number of people belonging to the membership grade 'Fellow'. It is shown that ChatGPT can understand table structure in the format of markdown and successfully fetch the number of people belonging to membership 'Fellow'.

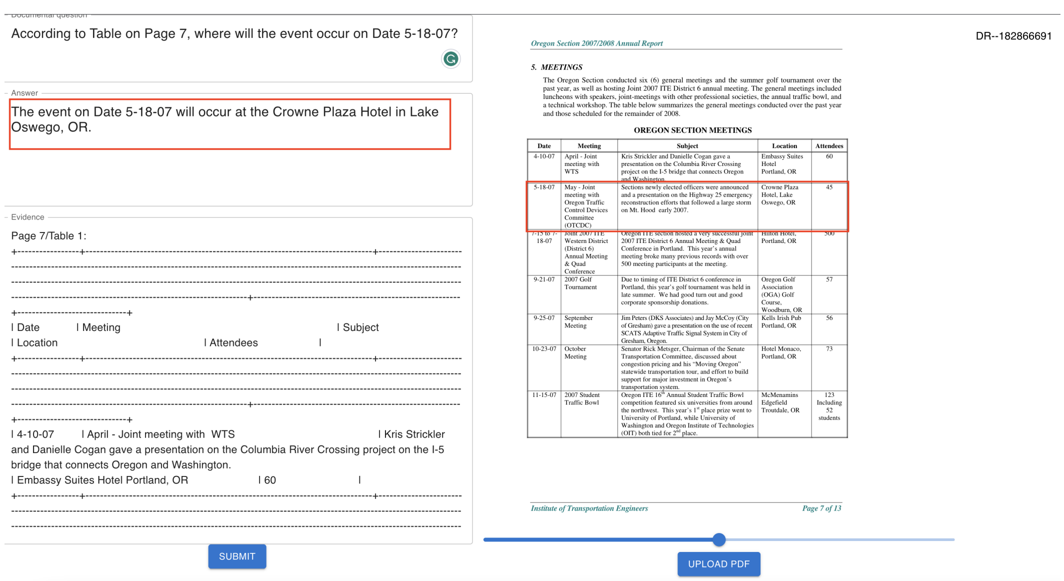


Figure 11: Table QA asking for the place where the event on Date 5-18-07 will occur.

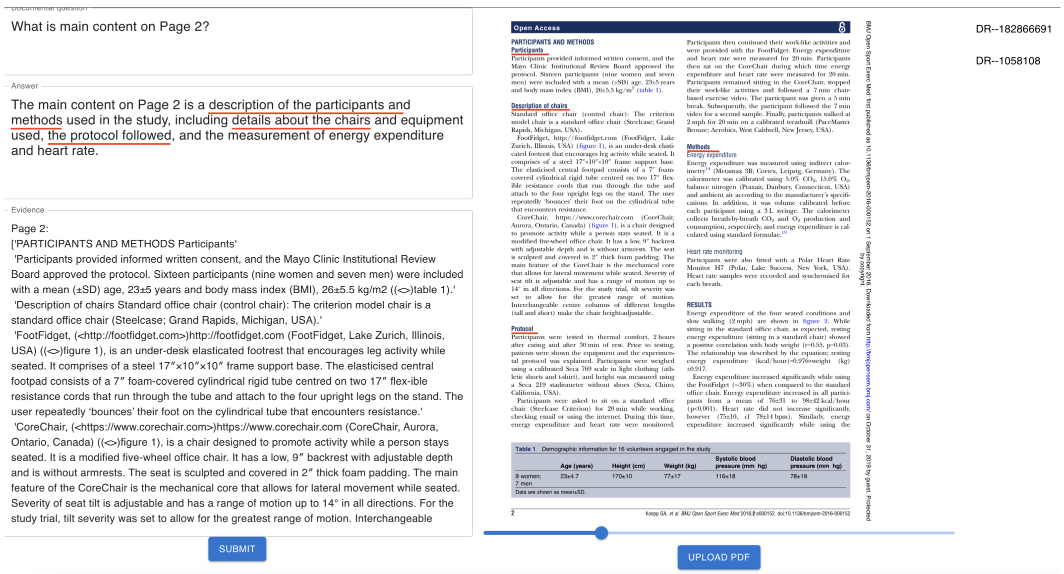


Figure 12: Page QA asking the main content on Page 2. The answer provides a high-level summarization of Page 2, covering the title of each section.

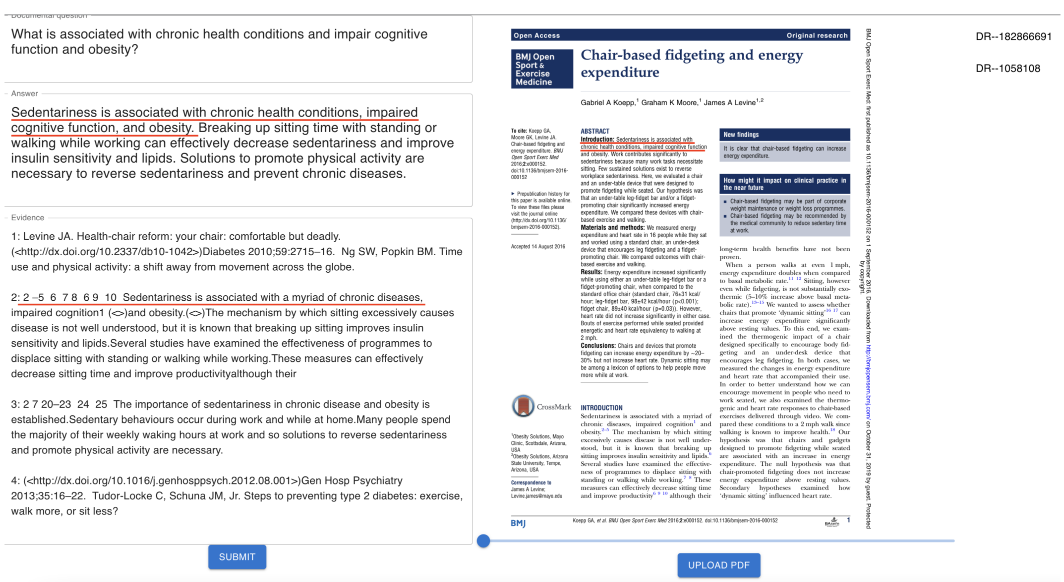


Figure 13: Single Document Content QA asking Sedentariness. The 2-**nd** retrieved sentence includes the answer and corresponds to the first sentence in the abstract of the paper.

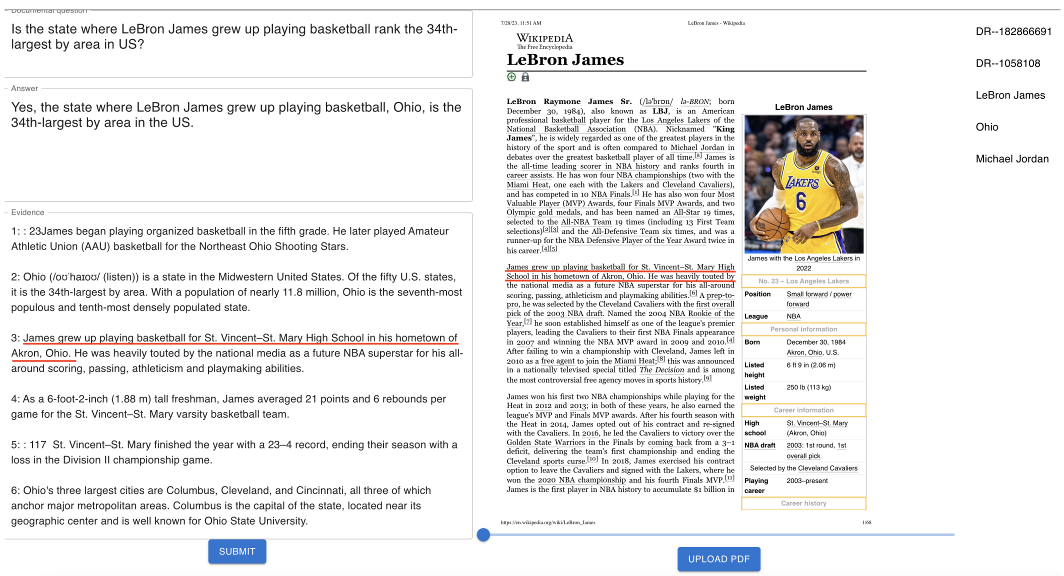


Figure 14: Multi-document Bridging Question asking the information about Lebron James and State Ohio. It requires to first retrieve the sentence stating the state where Lebron James grew up playing basketball.

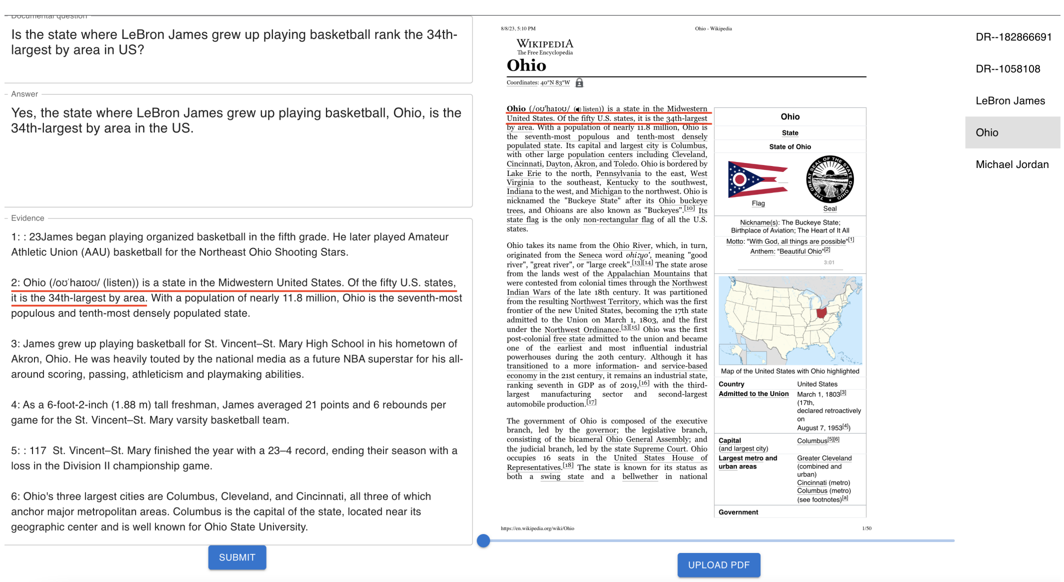


Figure 15: Multi-document Bridging Question asking the information about Lebron James and State Ohio. Then it requires to judge whether the State Ohio ranks the 34th-largest by area in the US.

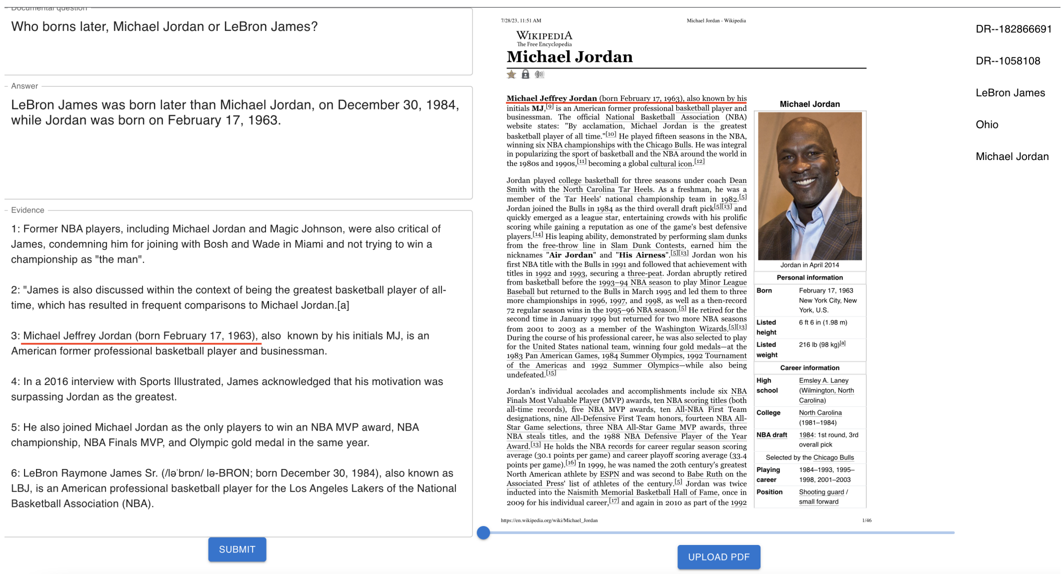


Figure 16: Multi-document Question comparing Lebron James and Michael Jordan. It requires the birthday information of Lebron and Jordan.

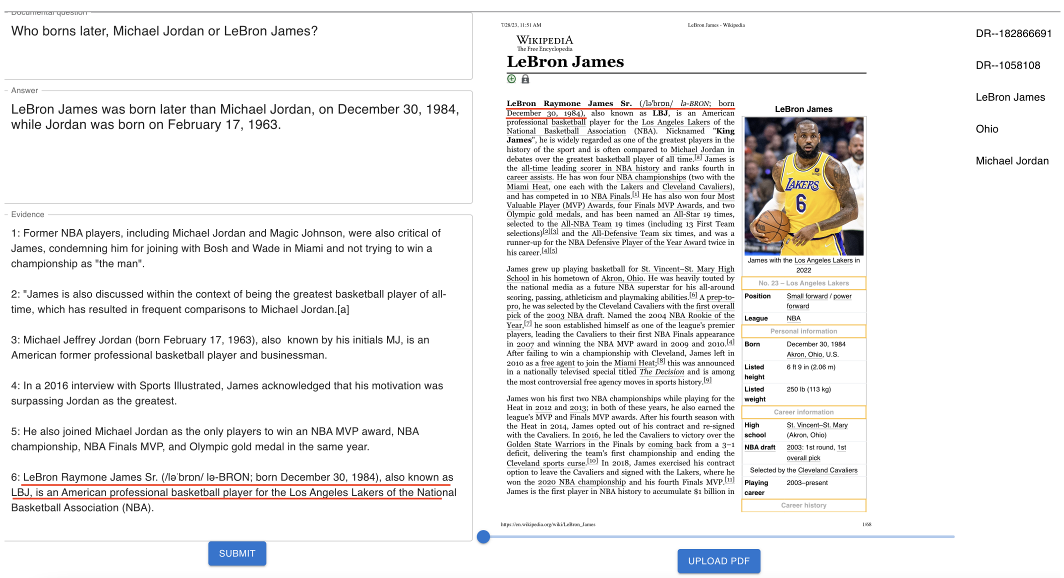


Figure 17: Multi-document Question comparing Lebron James and Michael Jordan. It requires the birthday information of Lebron and Jordan.

A.8 Visualizing the Reasoning-and-Retrieving Process of LM-guided Graph Traverser

In this section, we visualize the KG-LLaMA's reasoning-and-retrieving process in retrieving relevant context for MD-QA. Due to space limitation, for each question, we visualize the top-3 sentence nodes visited at 1-hop along with their generated evidence from LLaMA that required further to approach the answer. Based on the generated evidence, we retrieve the top-2 sentence nodes from the candidate neighbor queue. For each retrieved sentence node, we also visualize its ranking score given by TF-IDF. We can clearly see our designed LM-guided graph traversal could find the right evidence path to answer the given question.

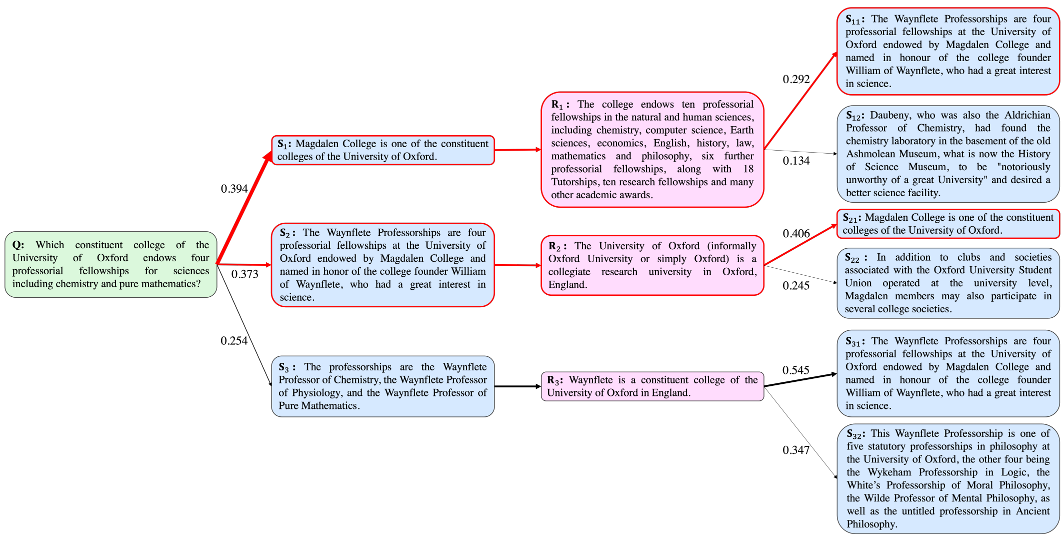


Figure 18: Visualizing the graph traversal over MD-QA-Example 1.

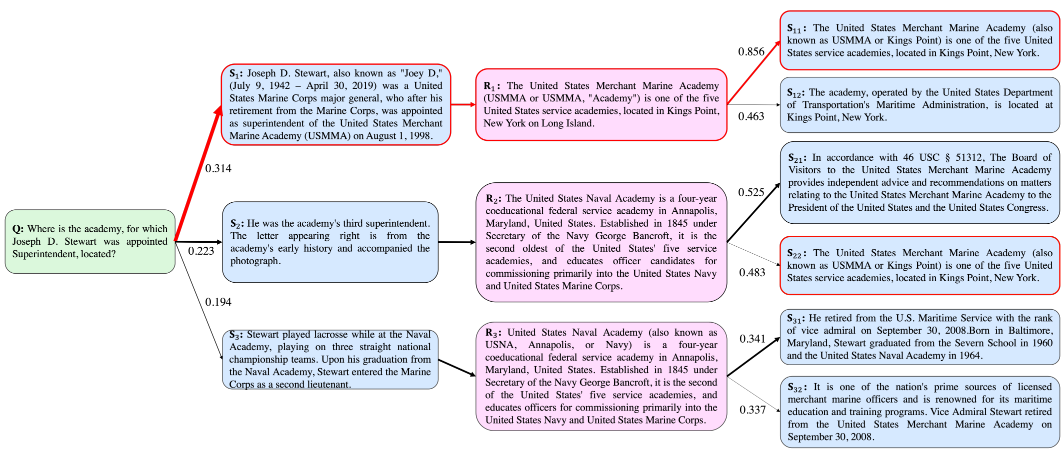


Figure 19: Visualizing the graph traversal over MD-QA-Example 2.

A.9 Prompt template used throughout this work

Listing 1: Examples of the Instruction Data for Fine-tuning LLaMA.

```
Question: Which magazine was started first Arthur's Magazine or First for Women?
Answer: Arthur's Magazine
Supporting Facts:
(1) Arthur's Magazine (1844−1846) was an American literary periodical published in Philadelphia in the 19th century.
(2) First for Women is a woman's magazine published by Bauer Media Group in the USA. The magazine was started in 1989.

Instruction: What evidence do we need to answer the question given the current evidence?
Input: Which magazine was started first Arthur's Magazine or First for Women? Arthur's Magazine (1844−1846) was an American literary periodical published in Philadelphia in the 19th century.
Output: First for Women is a woman's magazine published by Bauer Media Group in the USA. The magazine was started in 1989.
================================================================================================================

Question: In what year was the creator of the current arrangement of Simpson's Theme born?
Answer: March 28, 1941
Supporting Facts:
(1) The theme was re−arranged during season 2, and the current arrangement by Alf Clausen was introduced at the beginning of season 3.
(2) Alf Heiberg Clausen (born March 28, 1941) is an American film and television composer.

Instruction: What evidence do we need to answer the question given the current evidence?
Input: In what year was the creator of the current arrangement of Simpson's Theme born? The theme was re−arranged during season 2, and the current arrangement by Alf Clausen was introduced at beginning of season 3.
Output: Alf Heiberg Clausen (born March 28, 1941) is an American film and television composer.
```

Listing 2: Example of the Prompt for QA without Retrieved Contexts.

```
Given the following question, create a final answer to the question.
=========
QUESTION: What is the birthday of this Anglo−Irish actress, courtesan, and mistress, who was the mother to the illegitimate daughter of King William IV?
=========
ANSWER: Please answer in less than 6 words.
```

Listing 3: Example of the Prompt for QA with Retrieved Contexts.

```
Given the following question and contexts, create a final answer to the question.
=========
QUESTION: During which years was the model of car, featured on the cover of Earth's "Pentastar: In the Style of Demons" manufactured?
=========
CONTEXT:
1: Pentastar: In the Style of Demons is the third full−length studio album by the drone doom band Earth.
2: In 1957, he published The Interpersonal Diagnosis of Personality, which the Annual Review of Psychology called the "most important book on psychotherapy of the year".
3: During the evanescent heyday of the cyberdelic counterculture, he served as a consultant to Billy Idol in the production of the 1993 album Cyberpunk.
4: During the development of the Barracuda, one of the worst−kept secrets was Ford's plan to introduce a new sporty compact car based on the inexpensive Falcon chassis and running gear (which was eventually released as the Mustang in mid−model year 1964); the extent of the other changes was not known.
5: "Peace in Mississippi" is a cover of the Jimi Hendrix song. The original vinyl release of the album has an alternative take of "Peace in Mississippi".
6: A 1975 Barracuda had been planned before the end of the 1970−74 model cycle.
7: In the spring of 2021, when the third wave of the coronavirus epidemic arrived, VaÌĄradi called their airline one of the "rare rays of hope" for investors.
8: During this time the first U.S. Federal auto safety standards were phased in, and Chrysler's response a requirement for side−marker lights distinguishes each model year of the second−generation Barracuda: As the pony−car class became established and competition increased, Plymouth began to revise the Barracuda's engine options.
```

9: The Barracuda sold for a base price of US$2,512 ($24,000 today).The 1964 model year was the first for the Barracuda and also the last year for push−button control of the optional Torqueflite automatic transmission.

10: In the words of symbolist poet SteÌĄphane MallarmeÌĄ:Languages are imperfect because multiple; the supreme language is missing...no one can utter words which would bear the miraculous stamp of Truth Herself Incarnate...how impossible it is for language to express things...in the Poet's hands...by the consistent virtue and necessity of an art which lives on fiction, it achieves its full efficacy.

11: In France, the heart of the Decadent movement was during the 1880s and 1890s, the time of fin de sieÌĂcle, or end−of−the−century gloom.

12: Pentastar: In the Style of Demons is the third full−length studio album by the drone doom band Earth, released in 1996. It has a more rock−oriented sound than their earlier drone doom work, although in a very minimalist style.

13: The game was a rematch of the previous year's Russell Athletic Bowl, which Clemson won 40âĂŞ6.The two participants for the game were two of the semifinalists which were the Clemson Tigers and Oklahoma Sooners.

14: The effect of the war on Ernst was devastating; in his autobiography, he wrote of his time in the army thus: "On the first of August 1914 M[ax].E[rnst]. died. He was resurrected on the eleventh of November 1918".

15: Plymouth's executives had wanted to name the new model Panda, an idea unpopular with its designers. In the end, John Samsen's suggestion of Barracuda prevailed. Based on Chrysler's A−body, the Barracuda debuted in fastback form on April 1, 1964.

16: The Scapigliati (literally meaning "unkempt" or "disheveled") were a group of writers and poets who shared a sentiment of intolerance for the suffocating intellectual atmosphere between the late Risorgimento (1860s) and the early years of unified Italy (1870s).

17: Recurrent themes in his literary works include the supremacy of the individual, the cult of beauty, exaggerated sophistication, the glorification of machines, the fusion of man with nature, and the exalted vitality coexisting with the triumph of death.

18: Disc brakes and factory−installed air conditioning became available after the start of the 1965 model year. For the 1966 model year, the Barracuda received new taillamps, new front sheet metal, and a new instrument panel.

19: "Perhaps the worst failing of the book is the omission of any kind of proof for the validity and reliability of the diagnostic system," Eysenck wrote.

20: Based on stretched underpinnings of the rear−drive Alfa Romeo Giulia, it was rumored to be powered by a turbocharged V6 and arrive within the 2019 model year.

21: Their investments are in fleet development and the construction of airports, the first of which will be opened in Brasov.

22: He broke the hill record and this innovation was widely copied in the years to come.[citation needed]Mays made his mark on the track in such events as the 1935 German Grand Prix (scene of a famous victory of Tazio Nuvolari), sharing his ERA with Ernst von Delius.

23: There is still a question about the truth of the disclosure. In the 1968 Dragnet episode "The Big Prophet", Liam Sullivan played Brother William Bentley, leader of the Temple of the Expanded Mind, a thinly fictionalized Leary.

24: The Belgian FeÌĄlicien Rops was instrumental in the development of this early stage of the Decadent movement. A friend of Baudelaire, he was a frequent illustrator of Baudelaire's writing, at the request of the author himself.

25: After taking responsibility for the controlled substance, Leary was convicted of possession under the Marihuana Tax Act of 1937 on March 11, 1966, sentenced to 30 years in prison, fined $30,000, and ordered to undergo psychiatric treatment.

26: The general court delegation from Sullivan County is made up of all of the members of the New Hampshire House of Representatives from the county. In total, there are 13 members from 11 different districts.

27: Both teams then exchanged field goals, which brought the score to 16−10 in favor of Clemson. With 2:17 remaining, Oklahoma drove down the length of the field to score a touchdown, which gave the Sooners a one−point lead.

28: The average household size was 2.41 and the average family size was 2.88.23.90% of the population were under the age of 18, 6.40% from 18 to 24, 28.00% from 25 to 44, 25.90% from 45 to 64, and 15.80% who were 65 years of age or older.

29: The band announced the release of a deluxe version of the album "How It Feels To Be Lost", which came out on August 21, 2020. On June 2, 2021, the band released the single "Bloody Knuckles" from their upcoming album.

30: The 82nd Orange Bowl was a College Football Playoff semifinal with the winner of the game competing against the winner of the 2015 Cotton Bowl: Alabama Crimson Tide football in the 2016 College Football Playoff National Championship, which took place at the University of Phoenix Stadium in Glendale, Arizona.

```
=========
QUESTION: During which years was the model of car, featured on the cover of Earth's "Pentastar: In the Style of Demons" manufactured?
=========
ANSWER: Please answer in less than 6 words.
```

Listing 4: Example of the Prompt for QA with Retrieved Contexts for MDR, KGP-T5, KGP-LLaMA and KGP-MDR.

```
Given the following question and contexts, create a final answer to the question.
=========
QUESTION: Anthony Avent played basketball for a High School that is located in a city approximately 8 mi west of where?
=========
CONTEXT:
1: Newark is the second largest city in the New York metropolitan area, located approximately 8 mi west of lower Manhattan.\n Prior to Seton Hall, Avent played at Malcolm X Shabazz High School in Newark, New Jersey.

2: Newark is the second largest city in the New York metropolitan area, located approximately 8 mi west of lower Manhattan.\n The United States District Court for the District of New Jersey is also located in the city.

3: Newark is the second largest city in the New York metropolitan area, located approximately 8 mi west of lower Manhattan.\n Near Market Street and includes a dormitory for boarding students; and Saint Vincent Academy which is an all−girls Roman Catholic high school founded and sponsored by the Sisters of Charity of Saint Elizabeth and operated continuously since 1869.Link Community School is a non−denominational coeducational day school that serves approximately 128 students in seventh and eighth grades.

4: Prior to Seton Hall, Avent played at Malcolm X Shabazz High School in Newark, New Jersey.\n Newark is the second largest city in the New York metropolitan area, located approximately 8 mi west of lower Manhattan.

5: Prior to Seton Hall, Avent played at Malcolm X Shabazz High School in Newark, New Jersey.\n The United States District Court for the District of New Jersey is also located in the city.

6: Prior to Seton Hall, Avent played at Malcolm X Shabazz High School in Newark, New Jersey.\n On Newark Bay, it is run by the Port Authority of New York and New Jersey and serves as the principal container ship facility for goods entering and leaving the New York metropolitan area and the northeastern quadrant of North America.

7: He played collegiately at Seton Hall University where he played in the 1989 NCAA championship game. Prior to Seton Hall, Avent played at Malcolm X Shabazz High School in Newark, New Jersey.\n Prior to Seton Hall, Avent played at Malcolm X Shabazz High School in Newark, New Jersey.

8: He played collegiately at Seton Hall University where he played in the 1989 NCAA championship game. Prior to Seton Hall, Avent played at Malcolm X Shabazz High School in Newark, New Jersey.\n The United States District Court for the District of New Jersey is also located in the city.

9: He played collegiately at Seton Hall University where he played in the 1989 NCAA championship game. Prior to Seton Hall, Avent played at Malcolm X Shabazz High School in Newark, New Jersey.\n As of the 2020âĂŞ21 school year, the district, comprises 65 schools, had an enrollment of 40,423 students and 2,886.5 classroom teachers (on an FTE basis), for a studentâĂŞteacher ratio of 14.0:1.Science Park High School, which was the 69th−ranked public high school in New Jersey out of 322 schools statewide, in New Jersey Monthly magazine's September 2010 cover story on the state's "Top Public High Schools", after being ranked 50th in 2008 out of 316 schools.

10: Anthony Avent (born October 18, 1969) is an American former professional basketball player who was selected by the Atlanta Hawks in the first round (15th pick overall) of the 1991 NBA draft.\n Newark is the second largest city in the New York metropolitan area, located approximately 8 mi west of lower Manhattan.
```

```
11: Anthony Avent (born October 18, 1969) is an American former professional basketball player who was selected by the Atlanta Hawks in the first round (15th pick overall) of the 1991 NBA draft.\n The United States District Court for the District of New Jersey is also located in the city.

12: Anthony Avent (born October 18, 1969) is an American former professional basketball player who was selected by the Atlanta Hawks in the first round (15th pick overall) of the 1991 NBA draft.\n Atlanta United 1, New York Red Bulls 2 The first game in Atlanta United history was played before a sellout crowd of 55,297.

13: Anthony Avent (born October 18, 1969) is a retired American professional basketball player who was selected by the Atlanta Hawks in the first round (15th pick overall) of the 1991 NBA Draft.\n The total school enrollment in Newark was 77,097 in the 2013âĂŞ2017 ACS, with nursery and preschool enrollment of 7,432, elementary/high school (KâĂŞ12) enrollment of 49,532, and total college/graduate school enrollment of 20,133. The Newark Public Schools, a state−operated school district, is the largest school system in New Jersey.

14: Anthony Avent (born October 18, 1969) is a retired American professional basketball player who was selected by the Atlanta Hawks in the first round (15th pick overall) of the 1991 NBA Draft.\n As of the 2020âĂŞ21 school year, the district, comprises 65 schools, had an enrollment of 40,423 students and 2,886.5 classroom teachers (on an FTE basis), for a studentâĂŞteacher ratio of 14.0:1.Science Park High School, which was the 69th−ranked public high school in New Jersey out of 322 schools statewide, in New Jersey Monthly magazine's September 2010 cover story on the state's "Top Public High Schools", after being ranked 50th in 2008 out of 316 schools.

15: Anthony Avent (born October 18, 1969) is a retired American professional basketball player who was selected by the Atlanta Hawks in the first round (15th pick overall) of the 1991 NBA Draft.\n In the 2013−−2017 American Community Survey, 13.6% of Newark residents ages 25 and over had never attended high school and 12.5% didn't graduate from high school, while 74.1% had graduated from high school, including the 14.4% who had earned a bachelor's degree or higher.
=========
QUESTION: Anthony Avent played basketball for a High School that is located in a city approximately 8 mi west of where?
=========
ANSWER: Please answer in less than 6 words.
```

Listing 5: Example of the Prompt for Grading QA.

```
You are an expert professor specialized in grading whether the prediction to the question is correct or not according to the real answer.
==================
For example:
==================
Question: What company owns the property of Marvel Comics?
Answer: The Walt Disney Company
Prediction: The Walt Disney Company
Return: 1
==================
Question: Which constituent college of the University of Oxford endows four professorial fellowships for sciences including chemistry and pure mathematics?
Answer: Magdalen College
Prediction: Magdalen College.
Return: 1
==================
Question: Which year was Marvel started?
Answer: 1939
Prediction: 1200
Return: 0
==================
You are grading the following question:
Question: Anthony Avent played basketball for a High School that is located in a city approximately 8 mi west of where?
Answer: lower Manhattan
Prediction: Newark
If the prediction is correct according to the answer, return 1. Otherwise, return 0.
Return: your reply can only be one number '0' or '1'
```