Group 10 - Project Report
RAGs for Open Domain Complex QA

Roberto Gheda
5863503
Isidoro Tamassia
6054765
Athanasios Georgoutsos
6015883
Rodrigo Alvarez Lucendo
4997727

Abstract

As question answering (QA) systems increasingly rely on large language models (LLMs), integrating retrieval mechanisms to provide context is essential for handling complex queries, which may require information missing from its training data. This work explores the impact of different contexts on Retrieval-Augmented Generation (RAG) systems for complex QA, through comparing the performance of several LLMs with relevant, negative and random contexts. Our results indicate that injecting negative contexts and using different prompting techniques can impact QA performance differently depending on the LLM.

GitHub: github.com/r-gheda/rag-complex-qa

1 Introduction

Question Answering (QA) systems have significant applications in various domains, such as healthcare, law, and technical support, where precise information retrieval is crucial. Traditional QA models often struggle with complex questions that require compositional reasoning, evidence gathering from multiple sources, or numerical calculations. With the advent of Large Language Models (LLMs), the approach to QA has evolved, and significant progress has been achieved in the specific case of Open-domain QA, where the model has to answer based on large and heterogeneous collections such as Wikipedia. LLMs are usually given a short prompt and must reply with an answer based on their general knowledge acquired during training, which is usually conducted on massive large-scale corpora of information. Because of their In-Context Learning (ICL) capabilities (Xie et al., 2022), these models can also be prompted with a few question-answer examples to improve the quality of the generated answer, a technique called few-shot prompting.

Despite their strengths, LLMs face challenges in providing accurate answers to questions needing dynamic or recent knowledge which may not be present in the training data, sometimes leading to hallucinations (Ji et al., 2023), particularly in the case of zero-shot prompting. Retrieval-Augmented Generation (RAG) has emerged as a promising solution to this problem. In RAG, LLMs are usually supplemented with relevant contexts retrieved from external sources, ensuring that the generated answers are based on relevant and up-to-date information. However, recent studies conjecture that only providing relevant contexts could be a sub-optimal choice, and obtain surprisingly better results by injecting the prompt with randomly sampled contexts, which seem to act as beneficial noise (Cuconasu et al., 2024).

Motivated by these insights, we investigate the influence of various types of contexts on the performance of RAG systems, specifically focusing on compositional questions from the 2WikiMultiHopQA dataset (Ho et al., 2020), which require the model to perform effective multi-hop reasoning across multiple contexts to answer a given question. By repeating the experiments in several different settings and providing quantitative and qualitative analysis of the results, we aim to address the following research questions:

- **RQ1**: How do negative contexts impact downstream answer generation performance?
- **RQ2**: Are negative contexts more important for answer generation than related contexts?
- **RQ3**: Does providing only gold contexts deteriorate the performance compared to mixing with other negative or related contexts?
- **RQ4**: Does few-shot prompting help in mitigating the performance decay caused by inserting hard negatives in the prompt?

Our results indicate that LLMs perform better when provided with the oracle contexts rather than with Top-k retrieved ones. We find that injecting negative documents into the oracle contexts decreases their performance, but adding them to the Top-k retrieved contexts benefits one of the tested models, Flan-T5. Basic few-shot prompting rarely mitigates the performance decay introduced by hard negatives, while elicitive prompting (Wei et al., 2023) diminishes the issue.

2 Related Work

2.1 Open-Domain Question Answering

Open-Domain Question Answering (OpenQA) (Roy and Anand, 2022) is the task of creating systems capable of generating accurate and contextually relevant answers to a broad range of questions posed in natural language without limitations to specific domains or predefined datasets. Complexity in QA may arise from compositionality (Trivedi et al., 2022) or requirement to gather evidence from multiple sources (Chen et al., 2021) or numerical reasoning (Ling et al., 2017).

2.2 Retrieval Augmented Generation (RAG)

Retrieval Augmented Generation (RAG) (Mao et al., 2020; Gao et al., 2023) represents a cutting-edge approach in the domain of open-domain question answering, blending the strengths of retrieval-based and generative models. RAG consists in augmenting a query through text generation of heuristically discovered relevant contexts without external resources. The RAG framework typically involves two main components: a retriever and a generator (Gao et al., 2023). The retriever searches a corpus to identify pertinent documents or passages related to the query, while the generator uses this retrieved context to produce a more accurate answer. This dual approach enables RAG models to generate responses that are both factually grounded and coherent. Notable implementations of RAG have demonstrated significant improvements in performance metrics across various QA benchmarks (Gao et al., 2023), showcasing their ability to effectively handle diverse and complex queries.

2.3 Zero and Few-Shot Prompting

Zero-shot prompting involves giving the LLM a task with no examples or prior context provided. Therefore, the model must rely entirely on its pre-trained knowledge to understand and execute the task. Few-shot prompting involves providing the LLM with a few examples of the task before asking it to perform the task on new input. This method helps the model recognize the pattern or context of the task, often leading to better performance than zero-shot prompting (Brown et al., 2020).

3 Methodology

3.1 Dataset

The 2WikiMultiHopQA dataset (Ho et al., 2020) is a large-scale, open-domain question answering dataset designed to test multi-hop reasoning. This means that it requires models to gather and synthesize information from multiple sources in order to answer a question, unlike single-hop questions. As a result, it constitutes an effective resource for testing advanced models with a deeper knowledge understanding, capable to solve more complex QA tasks.

The dataset consists of text descriptions from Wikipedia and statements from Wikidata. There are four types of questions within the dataset:

- **Comparison questions** aim to compare two entities based on some given aspect *("Who died first, Fleetwood Sheppard or George William Whitaker?").*

- **Inference questions** aim to uncover a relationship between two entities, by processing through text passages the relationships of these entities with a third common entity *("Who is the maternal grandfather of Antiochus X Eusebes?").*

- **Compositional questions** are similar to inference ones, however there is no relationship to be uncovered *("What is the award that the director of film The Last Night Of The Barbary Coast got?").*

- **Bridge-Comparison questions** require the models to find the bridge entities and, then, make comparisons between them to obtain the answer *("Which film has the director who died later, The Fatal Mistake or The Devil'S Hairpin?").*

3.2 Context Categories

We feed our LLMs different types of contexts to see how the performance changes. These contexts have different characteristics:

- **Oracle contexts** refers to annotated documents that are highly relevant to the respective query, always providing pertinent information for answering it. Ideally, a LLM would be able to extract the useful information from these documents, process and combine it effectively and, subsequently, answer the query.

- **Hard negatives** are documents that are related and close to the query in the vector space but do not help answer the question. As such, they could be considered as misleading for the LLMs that attempt to answer the query. (Cuconasu et al., 2024) observed how the presence of these samples can reduce the RAG system prediction quality, which motivates our particular focus in this project on their role and influence.

- **Random documents** have no direct relation to the query and are selected arbitrarily. Their main role is to act as noise to the LLMs, helping us assess their capabilities of distinguishing between informative documents and irrelevant ones.

3.3 Context Retrieval

Document retrieval is the task of finding relevant documents in a large collection to answer specific queries. This is important for answering open-domain questions. In our work, we employ different kind of retrievers to analyze how their capalibities later influence the quality of the answers generated by the LLMs.

3.3.1 BM25

BM25 is a bag-of-words retrieval function that ranks a corpus of documents based on the query terms appearing in each document, according to the formula:

$$\sum_{i \in q} \log \frac{N}{df_i} \frac{(k_1 + 1)tf_i}{k_1((1-b) + b\frac{dl}{avdl}) + tf_i},$$

where df_i is the frequency of term i in the document and tf_i is the term frequency in the query. k_1 and b are parameters. dl and $avdl$ are, respectively, document length and average document length.

3.3.2 Contriever

Transformer-based approach that allows learning beyond lexical similarities. It trains the network to retrieve a document among a corpus, given a synthetic query generated from it. This is done in an unsupervised manner using a contrastive loss, which encourages positive pairs (from the same document) to have high scores and negative pairs (from different documents) to have low scores.

3.3.3 ADORE

ADORE (Zhan et al., 2021) is a query-side training method for Dense Retrieval (DR) models. This method requires a DR encoder, which is used for the document corpus. The authors proposed a novel encoder, STAR, for this purpose. Then, they adopt it as a pre-trained query encoder and fine-tune it with dynamic hard negative samples. Therefore, ADORE effectively improves the encoder's ranking performance.

We fine-tuned ADORE on the 2WikiMultiHopQA training data on top of a pre-trained STAR encoder.

3.3.4 Retrieval Evaluation Metrics

We use a variety of different metrics to assess the efficiency of the context retrieval methods:

- **Mean Reciprocal Rank** is the inverse of the position of the first relevant result, averaged across all queries:

$$\text{MRR} = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{rank_i}$$

where $|Q|$ is the number of queries and $rank_i$ is the position of the first relevant result for the i-th query.

- **Discounted Cumulative Gain (DCG)** gives higher scores to relevant results appearing earlier, discounting by their position:

$$\text{DCG} = \sum_{i=1}^{p} \frac{rel_i}{\log_2(i+1)}$$

where rel_i is the relevance score at position i. We use a normalized version of DCG.

- **Recall - Precision**: Recall measures the proportion of relevant documents successfully retrieved, while Precision measures the proportion of retrieved documents that are relevant.

3.4 Large Language Models (LLMs)

Different LLMs produce different responses according to their training procedure, model architecture, and dataset used for training and fine-tuning. For this reason, we employ three different LLMs to ensure the consistency of our results and see if there are interesting variations. Specifically, we pick the small versions of the open-source LLMs from Meta AI and Mistral AI, both fine-tuned on conversation datasets, and the Flan-T5 model from Google.

3.4.1 Llama-3

`Meta-Llama-3-8B-Instruct` is one of the Llama instruction tuned models that are optimized for dialogue use cases, and outperform many of the available open source chat models on common industry benchmarks.

3.4.2 Mistral

`Mistral-7B-Instruct-v0.1` is a finetuned version of the base model `Mistral-7B-v0.1`, a pretrained generative text model with 7 billion parameters from Mistral AI. The model was finetuned with publicly available conversation datasets and does not include any moderation mechanisms.

3.4.3 Flan-T5

`flan-t5-base` is an extension of the T5 model, fine-tuned on more than 1000 additional tasks, including extractive question answering and multi-hop question answering. The primary use of this model is research on zero-shot and in-context few-shot learning for NLP tasks (Chung et al., 2022). In our experiments, we are using `flan-t5-xl`, a larger version of the Flan-T5 base model, with 2.85 billion parameters.

3.4.4 LLM Evaluation Metrics

We employ the following metrics for evaluating our LLMs' responses:

- **Exact Match (EM)** requires the predicted answer to be identical to the reference answer ($EM = 1$ for a match, $EM = 0$ for no match). This strictness can mark many good predictions as incorrect.

- **ROUGE** measures how much information from the reference is in the generated answer based on recall. We use two ROUGE variations:

$$\text{R-1} = \frac{\sum_{n\in N} \text{match}(\text{1-g}(n))}{\sum_{n\in N} \text{count}(\text{1-g}(n))}$$

$$\text{R-L} = \frac{LCS(\text{pred, ref})}{\text{len}(\text{ref})}$$

  where 1-g(n) are unigrams, and LCS is the Longest Common Subsequence.

- **BERTScore** uses BERT embeddings to compute the similarity between tokens in the candidate and reference text. Let $\mathbf{y}_i$ be the embedding for token i in the reference and $\hat{\mathbf{y}}_j$ be the embedding for token j in the prediction. We compute precision (P) and recall (R):

$$\text{P} = \frac{1}{|\hat{y}|} \sum_{\hat{y}_j \in \hat{y}} \max_{y_i \in y} \mathbf{y}_i \cdot \hat{\mathbf{y}}_j$$

$$\text{R} = \frac{1}{|y|} \sum_{y_i \in y} \max_{\hat{y}_j \in \hat{y}} \mathbf{y}_i \cdot \hat{\mathbf{y}}_j$$

3.5 Prompting

We prompt Llama-3, Mistral and Flan-T5 using the following scheme:

> *You have to answer complex questions based on the provided contexts. Respond with as few tokens as possible. You don't need to explain your answer. Don't add any extra information.*
> *Document[1]: ...*
> *Document[2]: ...*
> ...
> *Question: ...*

This way, we try to constrain the models to reply with direct answers, avoiding long explanation that would negatively affect the score given by the employed metrics (particularly exact match).

4 Experiments

4.1 Context Retrieval

In Table 1, we report the retrieval results that we obtained measured by the different metrics. As we can notice, the pretrained Contriever yields consistently better performance according to all the metrics than BM25 does, and this is true for all the k values tested. Moreover, both are outperformed by the ADORE model we trained. This is as expected since we trained it on the associated training set of our dataset, differently from BM25

	BM25				Contriever				ADORE			
	NDCG	Recall	Precision	MRR	NDCG	Recall	Precision	MRR	NDCG	Recall	Precision	MRR
Top-1	0.2767	0.1283	0.2767	0.2767	0.3367	0.1631	0.3367	0.3367	0.4458	0.2196	0.4458	0.4458
Top-3	0.3953	0.2553	0.1867	0.3681	0.4620	0.3020	0.2042	0.4328	0.5811	0.3601	0.2461	0.5499
Top-5	0.4173	0.2997	0.1307	0.3812	0.4893	0.3447	0.1412	0.4488	0.5992	0.4041	0.1658	0.5616
Top-10	0.4354	0.3635	0.0798	0.3916	0.5102	0.4039	0.0825	0.4584	0.6068	0.4540	0.0942	0.5678

Table 1: Retrieval metrics for Top-k retrieved documents. Higher is better for all metrics.

	Llama-3			Mistral			Flan-T5		
	Contriever	BM25	ADORE	Contriever	BM25	ADORE	Contriever	BM25	ADORE
Top-1	**0.2033**	**0.2017**	**0.1950**	**0.2291**	0.2142	**0.1842**	0.2283	0.2292	**0.2092**
Top-3	0.1717	0.1742	0.1417	0.215	**0.2208**	0.1683	0.2492	0.2717	0.1958
Top-5	0.1558	0.1633	0.1283	0.2233	0.2092	0.1358	**0.265**	**0.28**	0.205

Table 2: Top-k exact match with different LLMs and retriever combination.

and Contriever which we used off-the-shelf. Given that higher retrieval metrics mean that more oracle contexts are present in the sampled documents, we would expect our LLMs to perform better with top-k ADORE retrieved context than Contriever or BM25 ones.

4.2 Prompting with top-k retrieved contexts

We tested how Llama-3, Mistral, and Flan-T5 perform with top-k retrieved contexts from BM25, Contriever, and fine-tuned ADORE. The Exact Match performances are reported in Table 2, while further metrics can be found in Appendix A.

We observe that, while all models have reasonably similar performance on top-1 experiments, they greatly vary when k increases. Namely, despite being the smallest of the tested models, only Flan-T5 manages to significantly improve its performance when we increase the number of documents retrieved by Contriever and BM25, reaching up to 28% Exact Match. Instead, Mistral cannot stably improve its Exact Match result, and Llama loses up to 5% of its performance when k is increased from 1 to 5. The fact that Flan-T5 does not show this behavior could be related to it having been trained on compositional QA datasets (Chung et al., 2022), therefore being less sensitive to increasing the number of contexts in the prompt.

Conversely, despite ADORE being the best retriever on paper, its top-k results are the worst ones for all tested LLMs. More specifically, on Llama-3 performance decreases by 7% while increasing k from 1 to 5, and Flan-T5 results stay more or less constant, without the nice increase in performance noticed for BM25 and Contriever results. We suggest that other than retrieving more oracle contexts, ADORE also retrieves a high number of hard negatives which leads to severe performance decay. We further discuss how ADORE hard negative contexts affect models' answering capabilities in subsection 4.4.

4.3 Prompting with oracle contexts only

The results are reported in the first line of Table 3. As expected, the performance is much better than with the Top-k retrieved documents, across all the models, with Exact Match improvement of 20.5% on Mistral, 30% on Flan-T5 and 35% on Llama-3 when comparing the best Top-k score obtained to the ones obtained with oracle contexts. Since the majority of the questions in the dataset can only be answered correctly by reasoning on at least 2 different oracle contexts, the LLM usually cannot obtain useful information from similar contexts that are not the exact ground truths. Therefore, the performance on the retrieved contexts is unlikely to reach or even surpass the one using oracle contexts only, unless the retriever really manages to retrieve the oracle contexts in the first positions.

4.4 Prompting with oracle contexts and injected noise

In this section, we explore two different ways of injecting noise in the oracle prompt, namely hard negatives and randomly chosen contexts.

As shown in subsection 4.2, Top-k ADORE contexts can often mislead the LLMs. We leveraged this unexpected behavior to sample hard negatives from the ADORE Top-k retrieved contexts (besides the ground truths).

Tables 3, 4 show that injecting noise in the oracle contexts leads to loss in accuracy on all experiments. However, ADORE and random documents

	Llama-3	Mistral	Flan-T5
Oracle	**0.5525**	**0.4341**	**0.5842**
Oracle + 1 Random	0.5117	0.3600	0.5608
Oracle + 2 Random	0.4442	0.3617	0.5625
Oracle + 3 Random	0.4333	0.3200	0.5733
Oracle + 5 Random	0.3808	–	–

Table 3: Exact Match with oracle contexts and randomly chosen documents.

	Llama-3	Mistral	Flan-T5
Oracle	**0.5525**	**0.4341**	**0.5842**
Oracle + 1 ADORE	0.5226	0.3617	0.5533
Oracle + 2 ADORE	0.4466	0.3375	0.5358
Oracle + 3 ADORE	0.4358	0.2842	0.5167
Oracle + 5 ADORE	0.1950	–	–

Table 4: Exact Match with oracle contexts and ADORE hard negatives.

	Contriever		BM25	
	Top-1	Top-3	Top-1	Top3
No noise	0.2283	0.2492	0.2292	0.2717
2 Random	0.23	0.2475	0.2433	0.2725
3 Random	0.2317	**0.2683**	**0.2525**	**0.2767**
5 Random	**0.2383**	0.2625	0.2483	–

Table 5: Exact Match with retrieved contexts and randomly chosen documents on Flan-T5.

	Contriever		BM25	
	Top-1	Top-3	Top-1	Top3
No noise	0.2283	0.2492	0.2292	0.2717
2 ADORE	0.2392	0.2633	0.2508	**0.2825**
3 ADORE	**0.2458**	**0.265**	0.2517	0.2733
5 ADORE	0.2458	0.2558	**0.26**	0.2758

Table 6: Exact Match with retrieved contexts and ADORE hard negatives on Flan-T5.

affects the LLMs' behavior differently. On Flan-T5, ADORE documents have a higher negative impact, leading to a 7% Exact Match decrease. On the other hand, random documents barely affect its performance, even increasing its accuracy by 1.3% when injecting three random documents rather than one.

Llama's performance is similar in both scenarios when only injecting a low amount of documents. To see whether its performance kept being similar between the two cases even with more noise, we injected 5 randoms and 5 ADORE hard negatives. In this case, the performance with randoms decreases to 38% of EM, while the one with ADORE drops drastically to 19%, confirming the major negative influence of the hard negative contexts.

4.5 Prompting with top-k retrieved contexts and injected noise

In this section, we inject random and hard negative documents as noise to the Top-k retrieved contexts.

After conducting some initial experiments (see tables in Appendix A) we observed that only Flan-T5 shows slight improvements and therefore we conducted further experiments with this model which are reported in Tables 5, 6. We observe that adding noise almost always improves the performance of Flan-T5 with Top-k retrieved contexts regardless of the type of noise, in contrast to what we noticed in subsection 4.4 with the oracle contexts.

4.6 Few-shot prompting

Since injecting ADORE hard negatives highly deteriorates the performance of our models when zero-shot prompting, we conducted other experiments to test whether we can improve the model performance via few-shot prompting. More specifically, we tried two different strategies on the Mistral model, the one that obtains the worst performance when injecting hard negatives into the prompt.

4.6.1 Basic Few-shot prompting

We tried simple 1-shot, 2-shot, and 3-shot prompting by providing the LLM with correct examples of how to reason and answer based on the given documents, using the following prompt structure:

> *You have to answer complex questions based on the provided contexts. Here are some examples of how you should reason and answer:*
>
> #1 *Example documents*
> #1 *Example reasoning*
> #1 *Example final answer*
> ...
> *Now, I will provide you with the contexts and the question. Respond with as few tokens as possible. You don't need to explain your answer. Don't add any extra information, only answer with the final answer.*

The results of this experiment are reported in Table 7. In general, basic few-shot prompting does

	1 ADORE	2 ADORE	3 ADORE
Zero-shot	**0.3617**	**0.3375**	0.2842
1-shot	0.3508	0.3258	**0.3017**
2-shots	0.3425	0.3208	0.2842
3-shots	0.3242	0.3075	0.2867

Table 7: Mistral Exact Match performance for few-shot prompting for different numbers of hard negatives injected in the oracle documents.

	1 ADORE	2 ADORE	3 ADORE
Zero-shot	0.3617	0.3375	0.2842
Elicitive 0-shot	0.3742	**0.3808**	0.3633
Elicitive 1-shot	0.3900	0.3733	**0.3842**
Elicitive 2-shots	0.3842	0.3717	0.3625
Elicitive 3-shots	0.3767	0.3725	0.3725
Elicitive 4-shots	**0.4042**	0.3717	0.3667

Table 8: : Mistral Exact Match performance for elicitive few-shot prompting for different numbers of hard negatives injected in the oracle documents.

not seem to improve the results, except for 1-shot prompting against 3 ADORE hard negatives, which improves the performance by around 1.6% compared to zero-shot prompting.

4.6.2 Elicitive Few-shot prompting

Elicitive prompting is a technique that has been shown to boost performance when answering compositional questions, like the ones found in the 2WikiMultiHopQA dataset, by letting the model think "things through" before outputting the final answer (Wei et al., 2023). In our experiments, we use the same prompt as in basic few-shot prompting, but we first let the model output an explanation of the compositional question based on the provided contexts, and later, we ask the model to answer the question solely based on the generated explanation.

Table 8 shows that elicitive prompting reduces the performance decay caused by the introduction of hard negatives retrieved by ADORE. Elicitive zero-shot, without any examples, already improves over basic zero-shot prompting by a margin that becomes more significant as the number of hard negatives increases. Specifically, zero-shot EM score raises by around 1.3% with one hard negative, 5% with two hard negatives, and 8% with three of them. Furthermore, we also notice some improvements when adding few-shot examples. Particularly, when adding one hard negative, performance raises over 4% going from 0-shot to 4-shot elicitive prompting.

5 Qualitative Analysis

In this section, we perform a qualitative analysis with Flan-T5 on selected questions from our dataset.

1. ***Question***: *Are both businesses, Telus and Ztr Control Systems, located in the same country?*

   Type: *Comparison*

   Answer: *yes*

   Here, two ground-truth documents inform about the country of origin of Telus and Ztr, which is Canada. Given these two documents only, the model returns the correct answer. However, adding 3 ADORE documents, which contain similarly named companies from other countries (i.e. Telmex from Mexico), misleads the model into giving the wrong answer ("*no*"). The same happens when we add 3 random texts (i.e. one mentioning the American lawyer Virginia B. Smith, whose name matches a U.S. State's name).

2. ***Question***: *Who is the paternal grandfather of Kujō Yoritsugu?*

   Type: *Inference*

   Answer: *Kujō Michiie*

   To answer this question, the model would have to first locate the father of Kujō Yoritsugu, and then the grandfather, from two independent documents. We observe that the model, with the oracle contexts, is unable to give the correct answer, as it returns the father. Using the Contriever, we manage to retrieve both ground-truths and another context-related document, that leads the model to give the correct answer. The same behavior holds, even with the addition of random noise.

3. ***Question***: *Where was the place of death of the director of film Mole Men Against The Son Of Hercules?*

   Type: *Compositional*

   Answer: *Roma*

   For this question, we need to extract the name of the director from one ground-truth document, and, then, his place of death from another. Using the Contriever, only one of the two ground-truths is present among the top-3, making it impossible for the model to answer correctly. The model gives a factually correct answer, "*Italy*", but not the desired one. However, injecting three ADORE hard negatives leads to the correct answer, "*Rome*". This happens because, among the hard negatives, there is a text about a sequel of the film, called "*The terror of Rome against the son of Hercules*". This is an example, where a hard negative document, containing semantically related but irrelevant information, unexpectedly helps the model in answering the question

4. ***Question***: *Do both films, Dial M For Murder and Marius (1931 Film), have the directors who are from the same country?*

   Type: *Bridge-Comparison*

   Answer: *yes*

   Within our dataset, bridge-comparison questions constitute the most difficult type to answer, with three processing steps. For this example, the LLM would have to extract the directors of each of the two films, then the country of each director, and, then, combine the information to answer. This involves 4 independent ground-truth documents. None of the contexts we experimented with led to the correct answer.

6 Conclusions and Future Work

Based on these findings, we now come back to our original research questions:

- **RQ1:** Negative contexts–both random and hard negatives–always decrease the performance in our experiments when injected to the oracle contexts, regardless of the model used. Since the oracle contexts already contain all the necessary information for answering the multi-hop questions, we argue that negative documents do not add any useful information and only distract the LLMs.

  Conversely, adding negative contexts to the Top-k retrieved documents can increase the performance. In cases where not all oracle contexts are retrieved, negative documents may induce the LLMs to answer correctly despite not containing relevant information, as in Example 3 of our qualitative analysis.

- **RQ2:** ADORE contexts always yield worse performance than Contriever or BM25 when prompting LLMs with Top-k retrieved contexts. Despite retrieving more oracle contexts, ADORE's additional documents are mostly hard negatives, negatively impacting performance. These documents also yield a worse performance decay than random noise when injected to the oracle contexts. This underlines the larger negative impact that related but irrelevant contexts have to the LLM reasoning compared to completely unrelated ones.

- **RQ3:** Only providing gold contexts leads the models to the best performance across all our experiments, since as already mentioned, any kind of further information added to the prompt does not bring additional value for answering compositional questions specifically designed to be answered through combining the associated gold contexts. Instead, the added information works as a distractor, as exemplified in Example 1 of the qualitative analysis.

- **RQ4:** Basic few-shot prompting cannot effectively handle the effect of adding hard negatives. However, elicitive zero-shot prompting increases the LLM's robustness. We argue this is because of the two-step approach that gives the model more time to answer a complex question instead of answering directly. Combining elicitive promoting with few-shot examples can further improve the performance.

Future research could explore how the discussed experiments generalize to larger models with stronger reasoning capabilities and larger context windows. More retrieved contexts could be tested in this case, as the restricted ability of the employed LLMs to handle long prompts has been a limitation in our study.

Additionally, we tested these models on a single dataset comprising the particular feature of multi-hop questions. It would be interesting to see whether our results translate to other common large QA benchmarks with different characteristics.

Finally, we found elicitive prompting to be an effective technique against hard negatives, but we only conducted a limited amount of experiments using it. Thus, deeper studies in this direction would be needed to quantify the advantages of this approach.

References

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners.

Zhiyu Chen, Wenhu Chen, Charese Smiley, Sameena Shah, Iana Borova, Dylan Langdon, Reema Moussa, Matt Beane, Ting-Hao Huang, Bryan Routledge, and William Yang Wang. 2021. FinQA: A dataset of numerical reasoning over financial data. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3697–3711, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Zhao, Yanping Huang, Andrew Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. 2022. Scaling instruction-finetuned language models.

Florin Cuconasu, Giovanni Trappolini, Federico Siciliano, Simone Filice, Cesare Campagnano, Yoelle Maarek, Nicola Tonellotto, and Fabrizio Silvestri. 2024. The power of noise: Redefining retrieval for rag systems. *arXiv preprint arXiv:2401.14887*.

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

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

Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. 2017. Program induction by rationale generation: Learning to solve and explain algebraic word problems. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 158–167, Vancouver, Canada. Association for Computational Linguistics.

Yuning Mao, Pengcheng He, Xiaodong Liu, Yelong Shen, Jianfeng Gao, Jiawei Han, and Weizhu Chen. 2020. Generation-augmented retrieval for open-domain question answering. *arXiv preprint arXiv:2009.08553*.

Rishiraj Saha Roy and Avishek Anand. 2022. *Question answering for the curated web: Tasks and methods in qa over knowledge bases and text collections*. Springer Nature.

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

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2023. Chain-of-thought prompting elicits reasoning in large language models.

Sang Michael Xie, Aditi Raghunathan, Percy Liang, and Tengyu Ma. 2022. An explanation of in-context learning as implicit bayesian inference.

Jingtao Zhan, Jiaxin Mao, Yiqun Liu, Jiafeng Guo, Min Zhang, and Shaoping Ma. 2021. Optimizing dense retrieval model training with hard negatives. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1503–1512.

A Full Experimental Results

	Exact Match	ROUGE-1	ROUGE-L	BERT Precision	BERT Recall
Oracle	0.5525	0.6761	0.6760	0.9394	0.9386
Contriever Top-1	0.2033	0.2733	0.2731	0.8889	0.8714
Contriever Top-3	0.1717	0.2608	0.2604	0.8836	0.8669
Contriever Top-5	0.1558	0.2594	0.2590	0.8855	0.8677
BM25 Top-1	0.2017	0.2598	0.2597	0.8878	0.8680
BM25 Top-3	0.1742	0.2549	0.2546	0.8836	0.8632
BM25 Top-5	0.1633	0.2546	0.2545	0.8812	0.8624
ADORE Top-1	0.1950	0.2702	0.2699	0.8895	0.8696
ADORE Top-3	0.1417	0.2372	0.2362	0.8806	0.8620
ADORE Top-5	0.1283	0.2383	0.2375	0.8788	0.8625
Contriever Top-1 and Rand-2	0.1158	0.2050	0.2045	0.8742	0.8584
Contriever Top-3 and Rand-2	0.1392	0.2359	0.2353	0.8768	0.8611
BM25 Top-1 and Rand2	0.1350	0.2077	0.2074	0.8713	0.8533
BM25 Top-3 and Rand2	0.1633	0.2456	0.2454	0.8798	0.8616
Oracle and Rand-1	0.5117	0.6417	0.6414	0.9345	0.9301
Oracle and Rand-2	0.4442	0.5800	0.5800	0.9255	0.9219
Oracle and Rand-3	0.4333	0.5750	0.5748	0.9195	0.9168
Oracle and Rand-5	0.3808	0.5159	0.5157	0.9111	0.9077
Oracle and ADORE-1	0.5116	0.6414	0.6412	0.9338	0.9301
Oracle and ADORE-2	0.4466	0.5826	0.5826	0.9256	0.9224
Oracle and ADORE-3	0.4358	0.5734	0.5732	0.9198	0.9168
Oracle and ADORE-5	0.1950	0.2702	0.2699	0.8895	0.8696

Table 9: Llama-3 Performance Metrics

	Exact Match	ROUGE-1	ROUGE-L	BERT Precision	BERT Recall
Oracle	0.4341	0.5507	0.5503	0.9114	0.9115
Contriever Top-1	0.2291	0.2901	0.2894	0.8827	0.8683
Contriever Top-3	0.215	0.2767	0.2764	0.8757	0.8664
Contriever Top-5	0.2233	0.2876	0.2872	0.8758	0.8689
BM25 top-1	0.2142	0.2618	0.2615	0.8768	0.8609
BM25 top-3	0.2208	0.2713	0.2710	0.8705	0.8618
BM25 top-5	0.2092	0.2671	0.2665	0.8628	0.8579
Oracle and Rand-1	0.3600	0.4793	0.4786	0.8987	0.9028
Oracle and Rand-2	0.3617	0.4755	0.4747	0.8969	0.9003
Oracle and Rand-3	0.3200	0.4389	0.4382	0.8890	0.8939
Oracle and ADORE-1	0.3617	0.4857	0.4850	0.8992	0.9028
Oracle and ADORE-2	0.3375	0.4610	0.4602	0.8926	0.8988
Oracle and ADORE-3	0.2842	0.4174	0.4174	0.8842	0.8904
BM25 Top-1 and Rand2	0.2008	0.2474	0.2470	0.8678	0.8575
BM25 Top-3 and Rand2	0.2133	0.2723	0.2721	0.8656	0.8604
Contriever Top-1 and Rand-2	0.1900	0.2444	0.2439	0.8697	0.8586
Contriever Top-3 and Rand-2	0.1908	0.2521	0.2515	0.8673	0.8610
ADORE Top-1	0.1842	0.2224	0.2222	0.8710	0.8542
ADORE Top-3	0.1683	0.2095	0.2095	0.8574	0.8505
ADORE Top-5	0.1358	0.1943	0.1941	0.8484	0.8464

Table 10: Mistral Performance Metrics

	Exact Match	ROUGE-1	ROUGE-L	BERT Precision	BERT Recall
Oracle	0.5842	0.7056	0.7053	0.9405	0.9281
Contriever Top-1	0.2283	0.2878	0.2875	0.8956	0.8633
Contriever Top-3	0.2492	0.3166	0.3160	0.9045	0.8721
Contriever Top-5	0.265	0.3302	0.3297	0.9075	0.8748
BM25 Top-1	0.2292	0.2772	0.2772	0.8993	0.8673
BM25 Top-3	0.2717	0.3265	0.3264	0.9064	0.8753
BM25 Top-5	0.28	0.3387	0.3386	0.9083	0.8761
ADORE Top-1	0.2092	0.2408	0.2406	0.8834	0.8643
ADORE Top-3	0.1958	0.2319	0.2317	0.8806	0.8630
ADORE Top-5	0.205	0.2443	0.2442	0.8832	0.8672
Oracle and Rand-1	0.5608	0.6790	0.6786	0.9379	0.9340
Oracle and Rand-2	0.5625	0.6842	0.6839	0.9393	0.9355
Oracle and Rand-3	0.5733	0.6926	0.6925	0.9386	0.9353
Oracle and Rand-4	0.5742	0.6899	0.6896	0.9359	0.9359
Oracle and ADORE-1	0.5533	0.6744	0.6741	0.9369	0.9326
Oracle and ADORE-2	0.5358	0.6533	0.6529	0.9337	0.9305
Oracle and ADORE-3	0.5167	0.6325	0.6322	0.9315	0.9276
Contriever Top-1 and Rand-2	0.23	0.2847	0.2845	0.9034	0.8702
Contriever Top-1 and Rand-3	0.2317	0.2864	0.2861	0.8953	0.8725
Contriever Top-1 and Rand-5	0.2383	0.2953	0.2949	0.8923	0.8723
Contriever Top-3 and Rand-2	0.2475	0.3139	0.3135	0.9048	0.8733
Contriever Top-3 and Rand-3	0.2683	0.3340	0.3335	0.9006	0.8800
Contriever Top-3 and Rand-5	0.2625	0.3236	0.3231	0.8974	0.8774
BM25 Top-1 and Rand-2	0.2433	0.2928	0.2928	0.8988	0.8674
BM25 Top-1 and Rand-3	0.2525	0.3013	0.3012	0.8913	0.8717
BM25 Top-1 and Rand-5	0.2483	0.2952	0.2952	0.8935	0.8730
BM25 Top-3 and Rand-2	0.2725	0.3262	0.3261	0.9047	0.8751
BM25 Top-3 and Rand-3	0.2767	0.3324	0.3322	0.8976	0.8791
Contriever Top-1 and ADORE-2	0.2392	0.3030	0.3028	0.9000	0.8759
Contriever Top-1 and ADORE-3	0.2458	0.3099	0.3094	0.8990	0.8774
Contriever Top-1 and ADORE-5	0.2458	0.3096	0.3093	0.8972	0.8767
Contriever Top-3 and ADORE-2	0.2633	0.3317	0.3314	0.8989	0.8773
Contriever Top-3 and ADORE-3	0.265	0.3327	0.3324	0.8996	0.8792
Contriever Top-3 and ADORE-5	0.2558	0.3238	0.3236	0.8979	0.8785
BM25 Top-1 and ADORE-2	0.2508	0.3129	0.3125	0.8989	0.8769
BM25 Top-1 and ADORE-3	0.2517	0.3085	0.3081	0.8972	0.8766
BM25 Top-1 and ADORE-5	0.26	0.3214	0.3212	0.8981	0.8784
BM25 Top-3 and ADORE-2	0.2825	0.3429	0.3424	0.9033	0.8818
BM25 Top-3 and ADORE-3	0.2733	0.3336	0.3333	0.8991	0.8795
BM25 Top-3 and ADORE-5	0.2758	0.3373	0.3371	0.9003	0.8803

Table 11: Flan-T5 Performance Metrics

	BM25					Contriever					ADORE				
	MRR	NDCG	Recall	Precision	MAP	MRR	NDCG	Recall	Precision	MAP	MRR	NDCG	Recall	Precision	MAP
Top-1	0.2767	0.2767	0.1283	0.2767	0.2767	0.3367	0.3367	0.1631	0.3367	0.3367	0.4458	0.4458	0.2196	0.4458	0.4458
Top-2	0.3483	0.3671	0.2137	0.2304	0.3483	0.4125	0.4324	0.2595	0.2633	0.4125	0.5296	0.5515	0.3186	0.3263	0.5296
Top-3	0.3681	0.3953	0.2553	0.1867	0.3645	0.4328	0.4620	0.3020	0.2042	0.4307	0.5499	0.5811	0.3601	0.2461	0.5478
Top-5	0.3812	0.4173	0.2997	0.1307	0.3735	0.4488	0.4893	0.3447	0.1412	0.4433	0.5616	0.5992	0.4041	0.1658	0.5531
Top-8	0.3888	0.4304	0.3409	0.0934	0.3720	0.4558	0.5038	0.3875	0.0983	0.4461	0.5663	0.6051	0.4409	0.1141	0.5470
Top-10	0.3916	0.4354	0.3635	0.0798	0.3691	0.4584	0.5102	0.4039	0.0825	0.4468	0.5678	0.6068	0.4540	0.0942	0.5440
Top-15	0.3933	0.4376	0.3861	0.0569	0.3636	0.4617	0.5178	0.4342	0.0594	0.4433	0.5693	0.6075	0.4765	0.0662	0.5367
Top-20	0.3948	0.4422	0.4049	0.0448	0.3619	0.4631	0.5203	0.4612	0.0474	0.4381	0.5702	0.6079	0.4940	0.0517	0.5310

Table 12: Retrieval metrics