TTQA-RS- A break-down prompting approach for Multi-hop Table-Text Question Answering with Reasoning and Summarization

Anonymous ACL submission

Abstract

Question answering (QA) over tables and text has gained much popularity over the years. Multi-hop table-text QA requires multiple hops between the table and text, making it a challenging QA task. Although several works have attempted to solve the table-text QA task, most involve training the models and requiring labeled data. In this paper, we have proposed a model - "TTQA-RS: A break-down prompting approach for Multi-hop Table-Text Question Answering with Reasoning and Summarization"[1]. Our model uses augmented knowledge including table-text summary with decomposed sub-question with answer for a reasoning-based table-text QA. Using open-source language models our model outperformed all existing prompting methods for table-text QA tasks on existing table-text QA datasets like HybridQA and OTT-QA's development set. Our results are comparable with the training-based state-of-the-art models, demonstrating the potential of prompt-based approaches using open-source LLMs. Additionally, by using GPT-4 with LLaMA3-70B, our model achieved state-of-the-art performance for prompting-based methods on multi-hop table-text QA.

1 Introduction

Question Answering over tables involves extracting the table cell containing the answer to the question. The most popular approach of table QA is to generate SQL queries using the question, i.e. the table-QA task is converted into a text-to-SQL task (Pasupat and Liang, 2015; Yu et al., 2018; Zhong et al., 2017). The SQL queries are then used to retrieve the answer from the tables. Some other recent approaches use an intermediate pre-training method on the flattened tables for QA (Herzig et al., 2020; Yin et al., 2020). QA over table and text is more challenging. Datasets like HybridQA (Chen et al., 2020b) and OTT-QA (Chen et al., 2020a) are examples of multi-hop table-text QA datasets where the answer to the question can exist in the table or the text. These two datasets make use of Wikitables along with text from Wikipedia to answer the questions. The tables in the HybridQA dataset contain hyperlinks linking the table cells to Wikipedia's text, making QA tasks more challenging. Additionally, HybridQA and OTT-QA are both multi-hop table-text datasets, which means that one or more hops between the table and text are required to derive the answer.

Over the years, several works have attempted to solve this task. But the majority of these works have used supervised-training, requiring a large amount of labeled data (Chen et al., 2020b; Sun et al., 2021; Wang et al., 2022; Eisenschlos et al., 2021; Feng et al., 2022; Kumar et al., 2023; Chen et al., 2020a; Li et al., 2021). In this paper, we have proposed a prompting-based approach while using open-source large language models (LLMs) for multi-hop table-text QA.

With the emergence of new generative-based LLM models, prompt-based methods using in-context learning have started being explored (Chen, 2023). Training models from scratch or even fine-tuning the models requires a large amount of labeled data. In-context learning is a cheaper alternative approach that does not need any fine-tuning but instead uses pre-trained language models (LLMs) to solve new tasks using a few examples as part of the prompt. The release of the new openAI models such as GPT 4 has opened new avenues of research in natural language processing and has encouraged further research in prompt learning. (Wei et al., 2022) has shown that reasoning with chain of thought (CoT) can significantly improve the ability of large language models to perform complex reasoning in tasks including QA. But small LLMs, i.e. models with less than 100B parameters using CoT prompting tend to hallucinate and produce incorrect results, urging research communities to

[1]The code is available in the Supplementary section

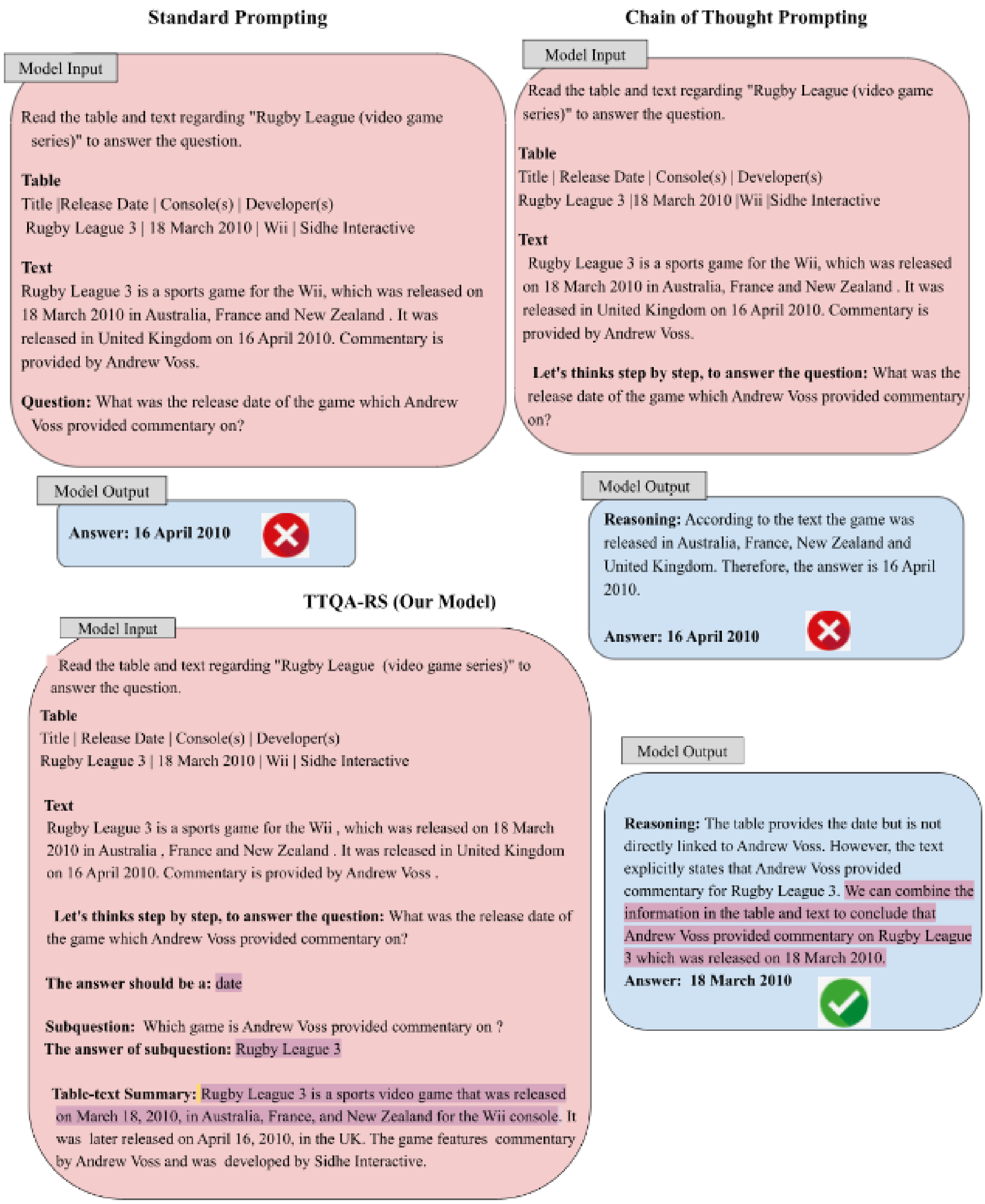


Figure 1: Comparison between Standard prompting, Chain of Thought prompting, and the TTQA-RS model.

use bigger LLMs which are expensive and also not open-source.

In this paper, we introduced a framework - TTQA-RS, a reasoning-based prompting approach for table-text QA that despite CoT's shortcomings on small-parameter models, we were able to reduce the hallucinations on open-source small models (i.e. we obtained a 6% increase in exact match score compared to the baseline CoT model for the HybridQA's test set). Furthermore, our proposed model was able to beat the state-of-the-art model - S3HQA's CoT prompting with GPT 3.5 results (Lei et al., 2023) on HybridQA dataset. By beating their model's performance, we have shown the potential for smaller LLMs in multi-hop table-text QA.

For our experiments, we have used HybridQA dataset and OTT-QA's development set. OTT-QA is an extension of the HybridQA dataset. Similar to the HybridQA dataset, the OTT-QA dataset is also constructed using questions based on Wikipedia tables and text. But unlike the HybridQA dataset, the test set of the OTT-QA dataset does not have hyperlinks in the table cells that can be linked to the Wikipedia text. Hence, the OTT-QA's test set is more challenging. Existing models including ours use a retriever-reader framework for table-text QA. In this paper, we narrowed our focus to the reader of the table-text QA task. Our goal is to develop a prompting strategy for the table-text QA reader that can work even with smaller LLMs. The task of linking tables and text passages for open-domain QA is out of scope of this paper. The development set of OTT-QA, similar to the HybridQA dataset, already has hyperlinks in the table cells linking to the wiki text. In the future, we plan to extend our approach to linking the table and text for cases when hyperlinks are absent in the table cells.

The TTQA-RS model breaks down the table-text QA problem into multiple steps. In the HybridQA and the OTT-QA dataset, the questions require multiple steps of reasoning over table and text to answer. The TTQA-RS model generates the sub-questions that can help in answering the complex questions. It also generates the summary of the table and text, which is in turn used for the table-text QA of the original questions. Breaking down the complex multi-hop QA problem into simple, smaller steps can help boost the model's overall performance. Furthermore, LLMs struggle with multi-level reasoning in a single step. So, breaking down the multi-hop QA problem along with providing an augmented information including the table-text summary can improve the performance of multi-hop table-text QA tasks using small open-source LLMs. In Figure 1, we show an example of a question from multi-hop QA that uses standard prompting, CoT, and the TTQA-RS approach for multi-hop QA.

2 Related Works

Multi-hop table-text QA can be a complex task as it requires multiple hops between the table and text to answer the questions. S3HQA (Lei et al., 2023) and MFORT-QA (Guan et al., 2024) are the only two existing models as per our knowledge that use in-context learning for multi-hop table-text QA. The S3HQA model has demonstrated table-text QA task using the Hybrid-QA dataset, whereas MFORT-QA has used the OTT-QA dataset. The S3HQA model uses a three-step method - a retriever with refinement training, a hybrid selector, and a generation-based reasoner with GPT 3.5 for the hybrid table-text QA task. MFORT-QA uses the Chain-of-thought (CoT) method to break down complex questions into smaller sub-questions, and uses Retrieval Augmented Generation to extract more context. Similar to the MFORT-QA model, we also break down complex questions into smaller sub-questions. With the complexity of the multi-hop QA task broken down into smaller questions, LLMs are in turn working on a smaller problem and perform better as single-step reasoners. Our model - TTQA-RS, additionally generates a summary using the retrieved table rows and passages. Then, for table-text question answering (QA), it uses the generated summary, the predicted entity type of the answer, and the generated sub-questions along with the answer.

3 Our Model

3.1 System Overview

The TTQA-RS model uses a retriever-reader model. Our reader breaks down the table-text QA problem into five steps - (1) Summary generation using retrieved tables rows and passages, (2) Question decomposition, (3) Entity type prediction of the expected answer, (4) Table-text QA of independent sub-question, and (5) Table-text QA of the original question. Figure 2 shows an overview of the TTRS-QA framework. The following subsections describe the TTQA-RS framework's retriever and reader in detail.

3.2 Retriever

The function of the retriever is to extract relevant rows and passages from the text linked to the table cells using hyperlinks. For the HybridQA dataset, we have used S3HQA model's (Lei et al., 2023) table retriever to extract the relevant row(s) from the table, and HYBRIDER's (Chen et al., 2020b) text-retriever to extract the relevant information from the linked passages. S3HQA's row retriever uses refinement training to train the retriever model. The tables contain hyperlinks to Wikipedia text. So, the passages linked to the retrieved rows are collected to form a pool. The passage retriever contains an ensemble retriever of TF-IDF retriever with longest-substring retriever and selects passages with cosine distance less than a certain threshold.

For experiments on OTT-QA's development set, we don't use any table retriever, i.e. we only use HYBRIDER's text retriever. The text linked to the table rows is extracted and then filtered using HYBRIDER's text retriever.

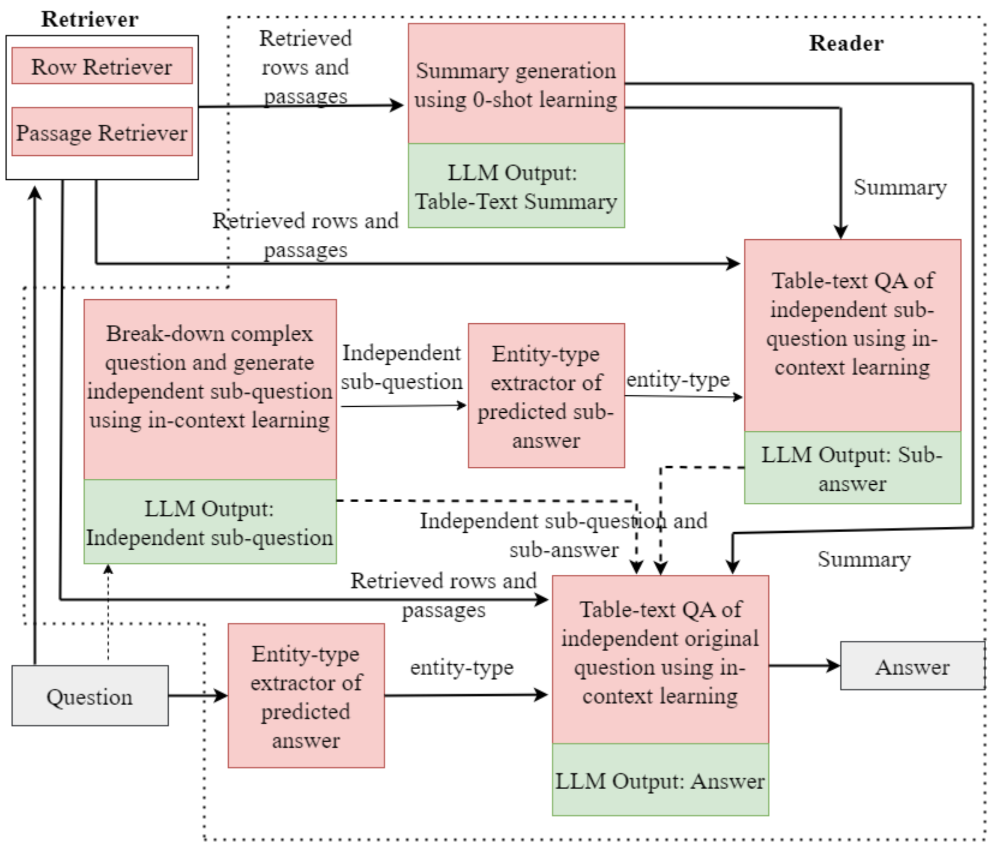


Figure 2: An overview of TTQA-RS framework. The dashed lines represent the reader for the table-text QA model.

3.3 Reader

3.3.1 Table-text Summarization

This is the first step of the reader model. The retrieved table rows are flattened with a delimiter separating the rows and columns of the table. The retrieved rows and passages are used to generate summaries of the table and text. We used zero-shot learning with LLaMA 3-70B model to generate the summaries. In Appendix B we have shown an example of a table-text summarization prompt.

3.3.2 Question decomposition

In the next step, we break down the questions and identify the sub-questions, such that the answer of one sub-question can aid in answering the original complex question. From here onwards, we will refer to the sub-question that can be answered first as the “independent sub-question”. Let’s take the first example of Figure 3. The complex question - "What was the release date of the game which Andrew Voss provided commentary on ?" can be broken down into sub-questions. The independent sub-question for this question is - "Which game has Andrew Voss provided commentary on?". The answer to this sub-question is "Rugby League 3". This can be used to simplify the original complex question to the following - " What was the release date of Rugby League 3?". Thus, including the information about the independent sub-question and the sub-answer helps to reduce the complexity of the multi-hop task. Identifying the independent sub-question and breaking down the complex multi-hop QA problem helps to reduce the complexity of the problem, and in turn, boosts the accuracy of the model. We use in-context learning with LLaMA3-70B model to generate the independent sub-questions for the given complex queries.

3.3.3 Entity type prediction of the expected answer

We identify the entity type of the expected answer for both the independent sub-question and also for the original question. For the following question - "What was the release date of the game which Andrew Voss provided commentary on?", the entity type of the expected answer is "date". Knowing that the expected answer is of type - "date", makes the LLM’s task of generating the answer considerably easier. We have used Spacy, an open-source Python library to obtain the entity type.

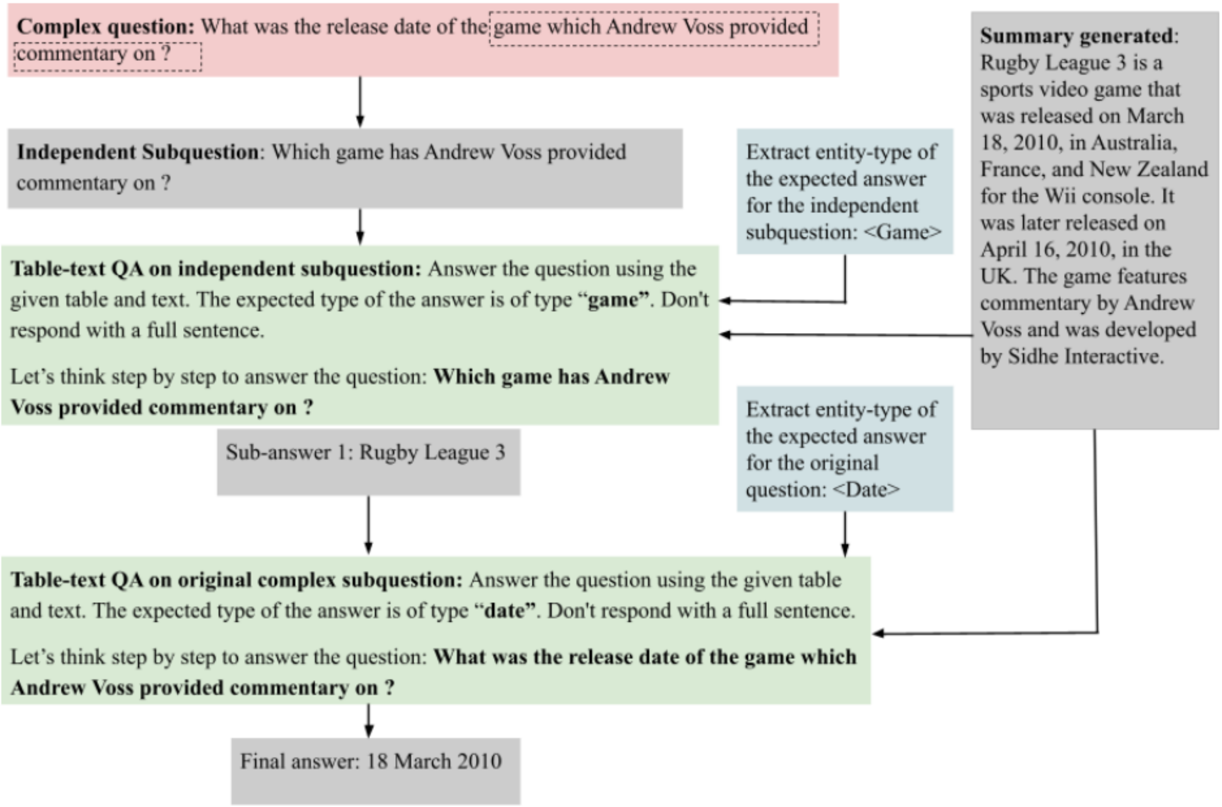


Figure 3: Example of our approach using TTQA-RS model

3.3.4 Table-text QA of independent sub-questions

In this step, we use few-shot learning with CoT to generate the answers of the independent sub-questions. The input prompt contains the retrieved table rows, retrieved passages, the table-text summary, and also the predicted entity type of the expected answer. This is used to generate the answer for the independent sub-question.

3.3.5 Table-text QA of the original questions

This is the final step of the table-text QA framework. To generate the answers of the original questions, we use CoT-based in-context learning similar to the previous step. But in addition to the prompt containing the retrieved rows, retrieved passages, table-text summary, and the expected entity type of the predicted answer of the original question, it also includes the independent sub-question with its generated sub-answer obtained in the previous step. Figure 3 shows an example of our reader's approach. For simplicity, we have excluded mentioning about the few-shot examples in Figure 3.

4 Experimental Setup

4.1 Datasets

HybridQA HybridQA (Chen et al., 2020b) is a large QA dataset that requires multi-hop reasoning over tables and text for QA. The questions in the HybridQA dataset are based on Wikipedia tables and corpora that are linked to the Wikipedia tables through hyperlinks.

OTT-QA (Chen et al., 2020a) is an open-domain multi-hop table-text QA dataset. For our experiments, we only use the development set of the OTT-QA dataset which contains the hyperlinks linking the table and the text (unlike OTTQA's test set).

4.2 Implementation details

The implementation details are shown in Appendix A.

4.3 Baseline Models

Standard prompting - For the baseline standard prompting model, we used the same retriever as in TTQA-RS model, (i.e. HYBRIDER's (Chen et al., 2020b) passage retriever with S3HQA's (Lei et al., 2023) table retriever for the HybridQA dataset). For experiments on OTT-QA's dev set, we don't use any table-retriever, i.e. we only use the HY-

BRIDER's passage retriever. For the reader, we performed in-context learning with standard prompting (Brown et al., 2020) for the QA task.

Chain of Thought Prompting (CoT) - Similar to the standard prompting baseline model, the CoT baseline model uses the same retriever as the TTQA-RS model. The reader uses in-context learning with CoT prompting (Wei et al., 2022).

5 Results and Discussion

5.1 Main Results

In this section, we discuss all our major findings. Table 1 displays the performance of our model with other existing models on the HybridQA dataset. We used exact match (EM) and F1 score to evaluate the performance the table-text QA models. From Table 1, we can observe that most existing models train their models for table-text QA. S3HQA (Lei et al., 2023) is the only model among the existing works that uses in-context learning for the HybridQA dataset. Please note that "S3HQA GPT-3.5 direct" refers to S3HQA model with standard prompting using GPT-3.5. Our TTQA-RS model with LLaMA 3-70B on the HybridQA's development set was able to beat S3HQA's CoT with GPT 3.5 by 3% exact match. Our 2-shot model with LLaMA-4 also beats the baseline standard and CoT prompting models by a huge margin (i.e. by 9% exact match when compared with standard prompting and by 6% exact match when compared with CoT in the test set). Furthermore, in Figure 4 and in Figure 5, we have shown the performance of our TTQA-RS model on different parameter models of LLaMA 2 and LLaMA 3 models on HybridQA test set and the OTT-QA development set respectively. For all the different parameter models of LLaMA-2 and LLaMA-3, our framework performed better than the baseline prompting models (i.e. standard prompting and CoT prompting). Our experiments show that our breakdown prompting approach with summarization and reasoning can improve the performance of all open-source models for table-text QA tasks. To show that our TTQA-RS approach can improve table-text QA on also GPT model, we have experimented with GPT-4 in the last stage of our model, i.e. table-text QA on the original question. For the remaining stages of the reader, we have used LLaMA 3- 70b. By adding GPT-4 in the last step, we were able to show the best performance with an exact match of 65.49 and F1 score of 76.43 in the development set, and an exact match of 63.69 and F1 score of 71.83 on the test set. With 2-shot learning using TTQA-RS LLaMA 3 70B + GPT-4 model, we were able to reach a model performance very close to the best existing training-based model (S3HQA with supervised learning) on the HybridQA dataset. For cost limitations, we have limited experiments with GPT-4 to only the last stage of our model.

Table 2 shows the performance of our model - TTQA-RS on the OTT-QA development set. To the best of our knowledge, MFORT-QA (Guan et al., 2024) is the only model that has used in-context learning for the OTT-QA dataset, but since they have not reported their performance on the development set, we therefore compare our model's performance with other existing works that trained the models. Our TTQA-RS model with LLaMA 3-70B + GPT-4 model achieved the best performance (exact match of 67.27 and F1 score of 79.55) on the development set and has achieved new state-of-the-art performance of the OTT-QA's development set.

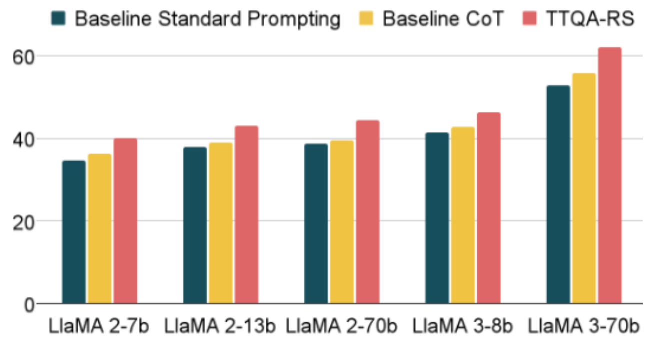
With the evaluation of our model - TTQA-RS on the HybridQA and OTT-QA development set, we have shown the potential of prompting approaches with small language models (like LLaMA) and also using GPT-4.

5.2 Analysis and Ablation Studies

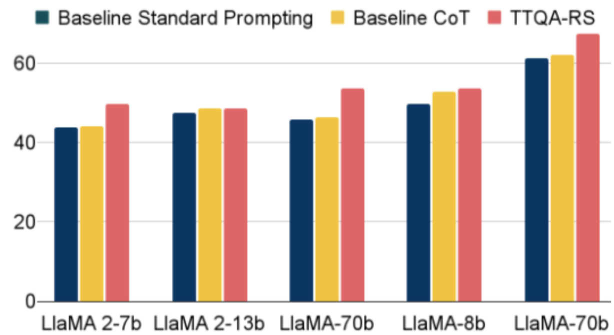
This section describes all the analysis and ablation studies performed on our model. Table 3 shows the ablation studies of our model using HybridQA dataset and OTT-QA's development set. We can observe that baseline CoT model outperforms the baseline standard prompting model. This shows the importance of reasoning in the multi-hop table-text QA task. Then, we test the model by adding the entity type prediction of the expected answer in the CoT prompt. We notice a significant increase in the performance of the model for all the datasets. In the 4th row of Table 3 we have added all the components of our final model except the table-text summary and we can see a further increase in performance in the HybridQA and OTT-QA datasets. Finally, we show the performance of our model TTQA-RS (i.e. last row) by including the generated table-text summary, and we can observe that our model performs the best with all the steps included (including summarization). Adding the table-text summary in the QA input prompt, helps the LLM model to recognize relevant information related to the table or text passages that might have otherwise

Table 1: Performance of our model-TTQA-RS and other related works on the HybridQA dataset

Type	Model	Dev EM / F1	Test EM / F1
Train	HYBRIDER (Chen et al., 2020b)	44.0 / 50.7	43.8 / 50.6
Train	DocHopper (Sun et al., 2021)	47.7 / 55.0	46.3 / 53.3
Train	MuGER2 (Wang et al., 2022)	571.1 / 67.3	56.3 / 66.2
Train	POINTR + MATE (Eisenschlos et al., 2021)	63.4 / 71.0	62.8 / 70.2
Train	DEHG (Feng et al., 2022)	65.2 / 76.3	63.9 / 75.5
Train	MITQA (Kumar et al., 2023)	65.5 / 72.7	64.3 / 71.9
Train	MAFiD (Lee et al., 2023)	66.2 / 74.1	65.4 / 73.6
Train	S3HQA (supervised learning) (Lei et al., 2023)	68.4 / 75.3	67.9 / 75.5
2-shot	S3HQA GPT 3.5 direct (Lei et al., 2023)	57.1 / 68.8	-
2-shot	S3HQA GPT 3.5 CoT (Lei et al., 2023)	60.3 / 72.1	-
2-shot	Baseline Standard prompting LLaMA 3-70B	48.97 / 60.18	52.88 / 61.42
2-shot	Baseline CoT LLaMA 3-70B	54.22 / 64.98	55.71 / 62.24
2-shot	TTQA-RS LLaMA 3 - 70B	63.12 / 73.61	61.97 / 67.56
2-shot	TTQA-RS LLaMA 3 - 70B + GPT-4	**65.49 / 76.43**	**63.69 / 71.83**
	Human	-	88.2 / 93.5

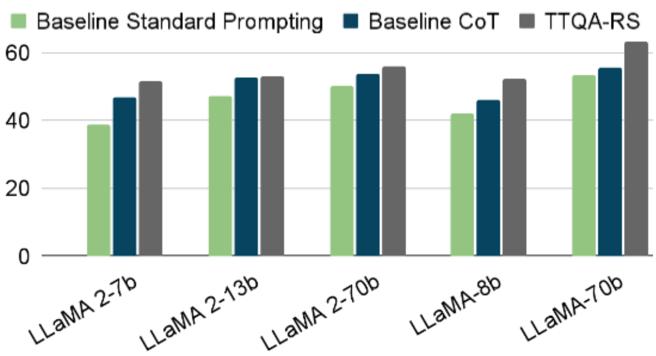


(a) EM score

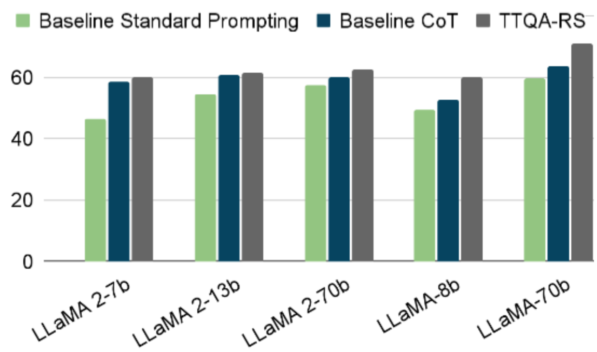


(b) F1 score

Figure 4: Performance of HybridQA test set on different LLaMA models



(a) EM score



(b) F1 score

Figure 5: Performance of OTT-QA dev set on different LLaMA models

gone unnoticed. This shows the importance of every component of our model and the need for our break-down prompting approach for table-text QA. In Appendix C, we evaluated the impact of the number of shots on the model's performance.

5.3 Human Evaluation Results

We have manually evaluated the first 100 samples of the table-text summaries generated by LLaMA3-70B, and also the independent sub-questions generated using LLM prompting. We obtained an accu-

Table 2: Performance of our model - TTQA-RS and other related works on OTT-QA development set.

Type	Model	Dev	
		EM	F1
Train	HYBRIDER (Top-1) (Chen et al., 2020b) (Chen et al., 2020b)	8.9	11.3
Train	HYBRIDER (best Top-K)	10.3	13.0
Train	Iterative-Retrieval + Single-Block Reader (Chen et al., 2020a)	7.9	11.1
Train	Fusion-Retrieval + Single-Block Reader (Chen et al., 2020a)	13.8	17.2
Train	Iterative-Retrieval + Cross-Block Reader (Chen et al., 2020a)	14.4	18.5
Train	Fusion-Retrieval + Cross-Block Reader (Chen et al., 2020a)	28.1	32.5
Train	CARP (Zhong et al., 2022)	33.2	38.6
Train	MITQA (Kumar et al., 2023)	40.0	45.1
2-shot	Baseline Standard prompting LLaMA3-70B	53.28	59.65
2-shot	Baseline CoT LLaMA3-70B	55.74	63.50
2-shot	TTQA-RS LLaMA 3 - 70B	63.15	70.84
2-shot	TTQA-RS LLaMA 3 - 70B + GPT-4	**67.27**	**79.55**

Table 3: Ablation studies of TTQA-RS on HybridQA and OTT-QA dataset using LLaMA 3-70B

Model	HybridQA		OTT-QA
	Dev	Test	Dev
Baseline Standard prompting	48.97 / 60.18	52.88 / 61.24	53.28 / 59.65
Baseline CoT	54.22 / 64.98	55.71 / 62.24	55.74/63.50
CoT + entity-type prediction of expected answer	59.68 / 67.47	57.75 / 64.65	57.53 / 66.24
Question decomposition and including sub-question with generated sub-answer + entity-type prediction of expected answer with COT (no summarization)	61.14 / 70.45	58.72 / 66.40	61.23 / 69.14
Our model - Question decomposition and including sub-question with generated sub-answer + entity-type prediction of expected answer + summarization with CoT	63.12 / 73.61	61.97 / 67.56	63.15 / 70.84

Table 4: Human evaluation of generated summaries for a sampled test set of HybridQA

Human Evaluation Metrics	Performance
Correctness	0.94
Inclusivity	0.98
Completeness	0.71

racy of 91% for question decomposition.

Table 4 tabulated the human evaluation results of the generated summaries for the sampled HybridQA test set. For evaluating the generated summaries using retrieved table rows and passages, we have used three evaluation metrics - correctness, inclusivity, and completeness. For correctness, we checked if the summary generated is overall correct and if the model generates any hallucination. For inclusivity, we checked if the generated summaries included information about both the retrieved rows and passages. Completeness was used to check if the generated summaries had complete sentences. We have included all our human evaluation results in the Supplementary section.

6 Conclusion

This paper proposes a prompting strategy of multi-hop table-text QA by generating table-text summaries and answers of sub-questions. We show that including summaries of retrieved table rows and passages in the prompt with our breakdown approach can substantially increase the performance of CoT prompting in table-text QA. The proposed method achieves new state-of-the-art performance among the prompting approaches for multi-hop table-text QA tasks using both open-source (i.e. LLaMA3-70B) and GPT-4 models. Our experiments specifically focussed on improving prompting strategies in the table-text QA readers. In the future, we plan to extend our work on the table-text QA retrievers which can further improve the QA performance.

Limitations

Our work has several limitations. Firstly, we are breaking down our problem into individual steps. Even though breaking down the problem into sub-problems helps to reduce hallucination while reasoning with open-source LLMs, it also causes error propagation. Errors made in the initial steps can result in wrong answers. Furthermore, in the experiment of using GPT-4 on our model, we have limited its usage only to the last step of the reader model as GPT-4 is expensive. Using GPT-4 in the remaining steps of the reader could have further improved the performance of our model.

Secondly, the performance of our prompting-based approach, even though is on par with the fine-tuned state-of-the-art models (or has outperformed the training-based state-of-the-model for OTT-QA development set), it's performance is still not close to the human performance. Also, we are using an existing retriever and the focus of this paper has only been to improve the reader's performance for multi-hop table-text QA. There is still potential to improve the overall performance of the model by using a better table and text retriever for this problem. Also, currently, we have only experimented with multi-hop table-text datasets in which the questions are already linked to the tables. The test set of the OTT-QA dataset does not have links between the tables with texts. This is out of scope of this current work, but in the future, we plan to explore more in this area.

References

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

Wenhu Chen. 2023. Large language models are few (1)-shot table reasoners. In *Findings of the Association for Computational Linguistics: EACL 2023*, pages 1120–1130.

Wenhu Chen, Ming-Wei Chang, Eva Schlinger, William Wang, and William W Cohen. 2020a. Open question answering over tables and text. *arXiv preprint arXiv:2010.10439*.

Wenhu Chen, Hanwen Zha, Zhiyu Chen, Wenhan Xiong, Hong Wang, and William Yang Wang. 2020b. Hybridqa: A dataset of multi-hop question answering over tabular and textual data. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1026–1036.

Julian Eisenschlos, Maharshi Gor, Thomas Müller, and William Cohen. 2021. MATE: Multi-view attention for table transformer efficiency. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7606–7619. Association for Computational Linguistics.

Yue Feng, Zhen Han, Mingming Sun, and Ping Li. 2022. Multi-hop open-domain question answering over structured and unstructured knowledge. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 151–156. Association for Computational Linguistics.

Che Guan, Mengyu Huang, and Peng Zhang. 2024. Mfort-qa: Multi-hop few-shot open rich table question answering. *arXiv preprint arXiv:2403.19116*.

Jonathan Herzig, Pawel Krzysztof Nowak, Thomas Müller, Francesco Piccinno, and Julian Eisenschlos. 2020. TaPas: Weakly supervised table parsing via pre-training. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4320–4333. Association for Computational Linguistics.

Vishwajeet Kumar, Yash Gupta, Saneem Chemmengath, Jaydeep Sen, Soumen Chakrabarti, Samarth Bharadwaj, and Feifei Pan. 2023. Multi-row, multi-span distant supervision for Table+Text question answering. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8080–8094. Association for Computational Linguistics.

Sung-Min Lee, Eunhwan Park, Daeryong Seo, Donghyeon Jeon, Inho Kang, and Seung-Hoon Na. 2023. MAFiD: Moving average equipped fusion-in-decoder for question answering over tabular and textual data. In *Findings of the Association for Computational Linguistics: EACL 2023*, pages 2337–2344. Association for Computational Linguistics.

Fangyu Lei, Xiang Li, Yifan Wei, Shizhu He, Yiming Huang, Jun Zhao, and Kang Liu. 2023. S3HQA: A three-stage approach for multi-hop text-table hybrid question answering. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 1731–1740. Association for Computational Linguistics.

Alexander Hanbo Li, Patrick Ng, Peng Xu, Henghui Zhu, Zhiguo Wang, and Bing Xiang. 2021. Dual reader-parser on hybrid textual and tabular evidence for open domain question answering. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4078–4088. Association for Computational Linguistics.

Panupong Pasupat and Percy Liang. 2015. Compositional semantic parsing on semi-structured tables. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1470–1480. Association for Computational Linguistics.

Haitian Sun, William W Cohen, and Ruslan Salakhutdinov. 2021. End-to-end multihop retrieval for compositional question answering over long documents. *arXiv preprint arXiv:2106.00200*.

Yingyao Wang, Junwei Bao, Chaoqun Duan, Youzheng Wu, Xiaodong He, and Tiejun Zhao. 2022. MuGER2: Multi-granularity evidence retrieval and reasoning for hybrid question answering. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 6687–6697. Association for Computational Linguistics.

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

Pengcheng Yin, Graham Neubig, Wen-tau Yih, and Sebastian Riedel. 2020. TaBERT: Pretraining for joint understanding of textual and tabular data. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8413–8426. Association for Computational Linguistics.

Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, Zilin Zhang, and Dragomir Radev. 2018. Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-SQL task. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3911–3921, Brussels, Belgium. Association for Computational Linguistics.

Victor Zhong, Caiming Xiong, and Richard Socher. 2017. Seq2sql: Generating structured queries from natural language using reinforcement learning. *arXiv preprint arXiv:1709.00103*.

Wanjun Zhong, Junjie Huang, Qian Liu, Ming Zhou, Jiahai Wang, Jian Yin, and Nan Duan. 2022. Reasoning over hybrid chain for table-and-text open domain qa. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence (IJCAI-22)*.

Appendix

A Implementation Details

For all our experiments we have made use of Nvidia Geforce GTX 1660 Ti. We have set the temperature to 0.5 for all experiments that uses zero-shot or few-shot learning. The table-text summary was generated using zero-shot learning with LLaMA 3-70B and for the question decomposition step, we used two-shot learning with LLAMA3B-70B. This was consistent for all our experiments. In the last two stages of our TTQA-RS framework, i.e. for steps involving table-text QA of independent sub-questions and for table-text QA of original questions, we have used few-shot learning with CoT, and we have experimented with different language models such as LLaMA 2-7b, LLaMA2-13b, LLaMA2-70b, LLaMA3-8b, and LLaMA3-70b.

B Table-text summarization prompt

Figure 6 shows an example of a table-text summarization prompt. The LLM output shows the generated summary.

Input Prompt:

Read the table and text regarding "2013 Montreal Impact season" and create a summary.

No | Pos | Name | MS
1 | GK | Perkins | 10

Text
Troy Perkins (born July 29, 1981) is a retired American soccer player. During his career, he played for clubs in the United States, Canada, and Norway . The 2006 Major League Soccer Goalkeeper of the Year award winner earned seven caps with the United States national team .

Summarize the given table and text.

LLM Output:

Summary: The table shows information about the 2013 Montreal Impact season with a single entry for the goalkeeper position held by Troy Perkins, who appeared in 10 matches. Perkins is a retired American soccer player who won the 2006 MLS Goalkeeper of the Year award and earned seven caps with the US national team, having played for clubs in the US, Canada, and Norway.

Figure 6: Prompt for zero-shot table-text summarization

C Impact of number of shots

In this section, we have performed an ablation study by increasing the number of shots while evaluating our model on the test set of the HybridQA dataset. This is shown in Figure 7. We have evaluated the impact of increasing k in k-shot learning on the baseline standard prompting model, baseline CoT model, and and the TTQA-RS model using LLaMA3 -70B. For standard prompting and CoT, we observe that with an increase in k from 0 to 3, there is an increase in the exact match score. After 3 shots, increasing the number of shots does not improve the performance. For the TTQA-RS model, there is an improvement in EM score from 0-shot to 2-shot, after which increasing the k value does not improve the exact match score of the model.

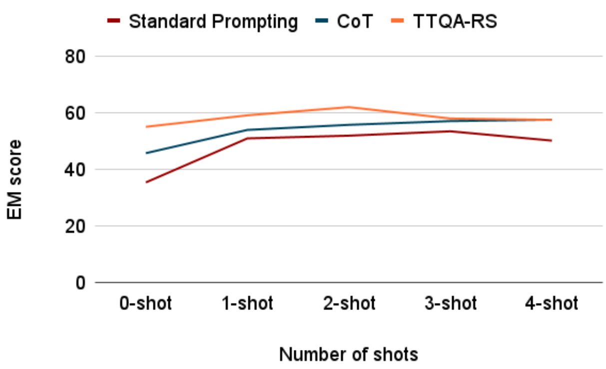


Figure 7: k-shot ablation study over Hybrid-QA test set