Knowledge-Driven CoT: Exploring Faithful Reasoning in LLMs for Knowledge-intensive Question Answering

Anonymous ACL submission

Abstract

Equipped with Chain-of-Thought (CoT), Large language models (LLMs) have shown impressive reasoning ability in various downstream tasks. However, suffering from hallucinations and the inability to access external knowledge, LLMs often come with incorrect or unfaithful reasoning, especially when solving knowledge-intensive tasks such as KBQA. To alleviate this issue, we propose a framework called Knowledge-Driven Chain-of-Thought (KD-CoT) to verify and modify reasoning traces in CoT via interaction with external knowledge, and thus overcome the hallucinations and error propagation. Concretely, we formulate the CoT rationale of LLMs into a structured multi-round QA format. In each round, a QA system retrieves external knowledge related to the sub-question and returns a more precise answer, and then LLMs generate subsequent reasoning steps based on the returned answer. Moreover, we construct a KBQA CoT collection, which can serve as In-Context Learning demonstrations, and be utilized as feedback augmentation to train a multi-hop question retriever. Extensive experiments on WebQSP and ComplexWebQuestion datasets demonstrate the effectiveness of the proposed KD-CoT, which outperforms the vanilla CoT ICL with 8.0% and 5.1%. Furthermore, our proposed feedback-augmented retriever can retrieve more valuable knowledge in the multi-hop scenario, achieving significant improvement in Hit and Recall performance.

1 Introduction

Large language models (LLMs) pre-trained on massive language corpora have shown impressive performance in various NLP tasks (Brown et al., 2020; Du et al., 2022; Touvron et al., 2023a). The ability of LLMs can be further unleashed through in-context learning conditioning on a few concatenated demonstrations without task-specific training or fine-tuning. Recent works have explored LLMs' reasoning ability to tackle complex reasoning problems through prompting (Wei et al., 2023; Zhou et al., 2023) and decoding (Wang et al., 2023b).

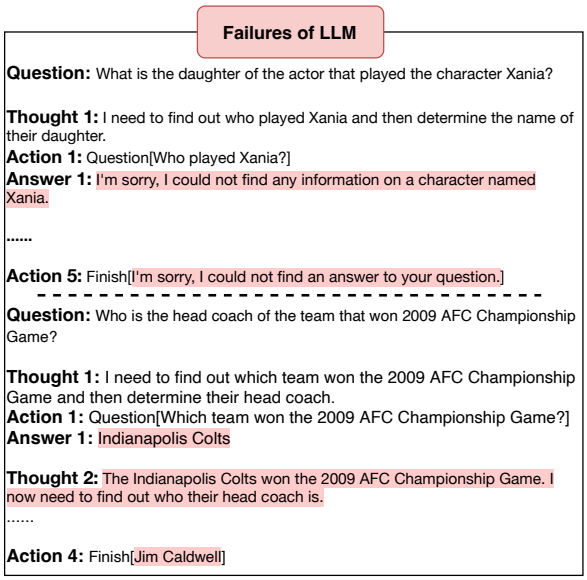


Figure 1: LLMs suffer from hallucination or inability to answer sub-questions while solving QA tasks that require encyclopedic knowledge, resulting in erroneous subsequent reasoning and final answer. We highlight the errors with red blocks.

Despite advancements, LLMs still encounter hallucinations or lack of knowledge while solving knowledge-intensive tasks. As shown in Figure 1, both these failures will lead to error propagation and incorrect final answers. A naive method is to directly input retrieved contextual knowledge into LLMs (Xu et al., 2023; Yao et al., 2023; Wang et al., 2023a) as augmentation. However, ensuring comprehensive knowledge coverage necessitates a large amount of context, making it difficult for LLMs to fully understand (Liu et al., 2023). Considering that current state-of-the-art methods leverage a retrieve-then-read pipeline for solving knowledge-intensive QA tasks, we can address the aforementioned issue by applying such a paradigm.

In this paper, we propose a Knowledge-Driven Chain-of-Thought (KD-CoT), an interactive framework that utilizes a QA system to access external knowledge and provide high-quality answers to LLMs for solving knowledge-intensive KBQA tasks. Specifically, we formulate the CoT rationale of LLMs into a multi-round QA format, and leverage a retriever-reader-verifier QA system to solve sub-questions iteratively. In each round, the retriever first retrieves knowledge related to the sub-question and the reader generates candidate answers based on the retrieved information. The verifier then compares the candidate answers with the original sub-answers generated by LLMs, and delivers the final sub-answers to replace the current one. We re-request LLMs for subsequent reasoning using the preceding corrected rationale. KD-CoT is designed to facilitate dynamic reasoning where we can verify and adjust intermediate reasoning steps by accessing external knowledge. We also construct a KBQA CoT collection that can be applied as demonstrations for ICL to improve the performance of LLMs.

To obtain accurate sub-answers for intermediate sub-questions, a high-quality external QA system is essential. Previous studies leverage a retrieve-then-read pipeline on linearised KB knowledge, achieving SOTA results in KBQA (Oguz et al., 2022; Yu et al., 2023). However, none of these studies focused on knowledge retrieval for complex multi-hop questions. To address the challenge of knowledge retrieval for multi-hop questions (where sub-questions within the CoT may still be multi-hop), we propose a robust retriever that leverages the feedback from the constructed CoT collection. Concretely, as the sub-question of the last round QA is more likely to be one-hop, we leverage it as the augmentation to the original query to identify more valuable positive or hard negative instances, which are then used as training data for the retriever.

Our main contributions can be summarized as:

- We present a KBQA CoT collection by prompting LLMs, which could be used for fine-tuning smaller LMs to acquire CoT reasoning ability and be applied to perform ICL.
- We propose a retriever-reader-verifier QA system to access external knowledge and interact with LLM. We leverage the constructed CoT collection as feedback augmentation to train a more robust retriever for solving multi-hop question retrieval, which achieves significant improvement on WebQSP and CWQ.
- We introduce an interactive framework to improve the reasoning performance of LLMs. Experimental results demonstrate the effectiveness of our proposed framework, achieving 8.0 and 5.1 Hit@1 improvement on WebQSP and CWQ compared to the vanilla ICL method.

2 Methodology

In this section, we first present the procedure for constructing the CoT Collection. Then we introduce the Knowledge-Driven CoT framework, which encompasses the implementation of the interaction and the training of the external QA system.

2.1 CoT Collection

We first manually write several accurate CoT demonstrations as the anchor set to perform ICL, and then we employ an iterative algorithm to construct our full collection. In each iteration, we choose the candidate in the current collection that holds the highest cosine similarity with the question in the training set to serve as the demonstration. We use RoBERTa (Reimers and Gurevych, 2019)[1] to embed questions and compute the cosine similarity between them. Next, we request ChatGPT[2] to generate the structured CoT, and append generated results "Finish" with the correct answer to the collection. The construction details are referred to Algorithm 1. Notably, we observe that concatenating the ground truth answer and the composition answer (if have) as "Hint" before the rationale can greatly improve the efficiency of collection construction, so the final demonstration is presented in the format of <Question, Hint, CoT> as illustrated in Appendix B.

2.2 Knowledge-Driven CoT

Due to hallucinations and the inability to access external knowledge, LLMs struggle to generate faithful reasoning steps for knowledge-intensive QA tasks. To address this issue, we propose Knowledge-Driven Chain-of-Thought Reasoning (KD-CoT), which incorporates a QA system to interact with LLMs (ChatGPT in this paper). The

[1]Downloaded from Sentence Transformers; Roberta-large-nli-stsb

[2]OpenAI gpt-3.5-turbo

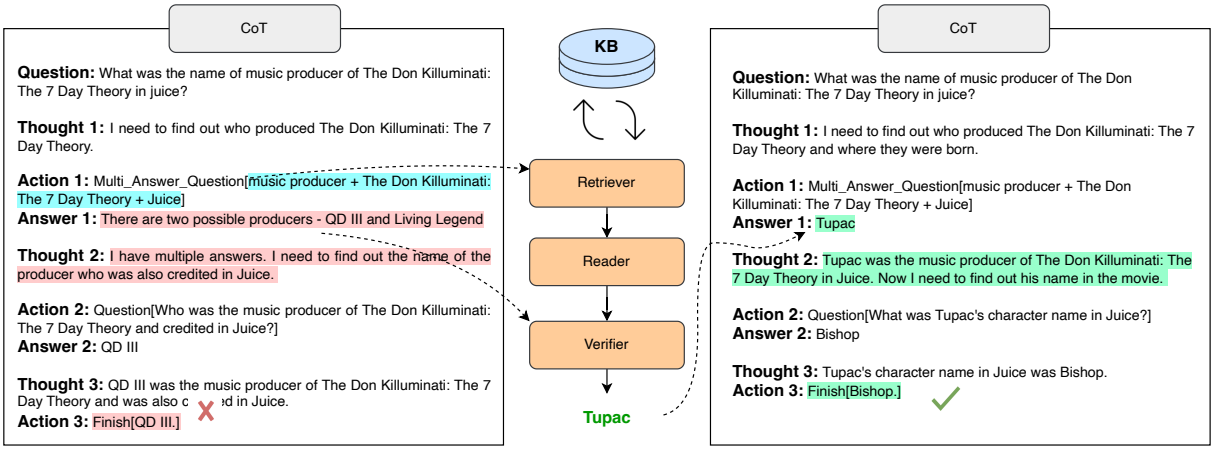


Figure 2: The overall framework of Knowledge-Driven CoT, including a prompted large model and a QA system that accesses external knowledge. By modifying sub-answers of intermediate questions, LLM can generate more faithful subsequent inference steps, which lead to correct final answers. Blue, Green, and Red blocks represent the sub-question fed to QA system, correct/incorrect reasoning and answers, respectively

Algorithm 1 Construct CoT Collection

Require: Human-annotated demonstrations, D_h
Require: A fixed human-annotated instruction, I
Require: Question-Answer training set, Q, A
Require: Large language model, LLM
Require: Demonstration selection pool, P
  $P \leftarrow D_h$
  $iteration \leftarrow 0$
  while Q is not empty and $iteration < 5$ **do**
    $Demons \leftarrow SimilaritySelection(Q,\ P)$
    $Inputs \leftarrow Concat(I,\ Demons,\ Q)$
    $Outputs \leftarrow LLM(inputs)$
    $Constructed \leftarrow Match(outputs,\ A)$
    $P \leftarrow Extend(P,\ Constructed)$
    $Q \leftarrow Q \setminus A$
    $iteration \leftarrow iteration + 1$
  end while
  return P

overall framework of our proposed KD-CoT is shown in Figure 2.

For each question in the test set, we select the instance with the highest cosine similarity from the collection, and utilize its rationale as the demonstration to perform one-shot ICL[3]. Then the extracted intermediate sub-question is taken as the input of the QA system to perform interaction, which is comprised of a retrieve-then-read pipeline and an answer verifier. The former module retrieves external knowledge and proposes candidate answers based on the retrieved information, while the latter chooses between the original sub-answers generated by LLM and the proposed candidate answers. We repeat the above interaction until the CoT is finished.

2.3 Implementation of the external QA system

Our QA system contains three components: a retriever, a reader and a verifier. In this subsection, we first describe how we convert the structured KB knowledge into unstructured text, and then elaborate on these three components in detail.

KB Linearization We aim to interact with both structural (Freebase) and unstructured (Wikipedia) external knowledge. However, directly retrieving information from KB is non-trivial due to its large scale and complications with semantics and structure. To address this issue, we simply follow the linearization method proposed in (Yu et al., 2023) to process Freebase KB data (Bollacker et al., 2008) into unstructured text. Given a head entity, we extract its 1-hop subgraph and concatenate the KB triplets with spaces, then group the entire subgraph into a single passage. For example (music recording, releases, Palavras de Guerra Ao Vivo) and (music recording, artist, Olívia Hime) will be processed into "music recording releases Palavras de Guerra Ao Vivo. music recording artist Olívia Hime".

After pre-processing, we concatenate Wikipedia passages with KB passages to perform knowledge retrieval.

FeedBack-Augmented Retriever To align with

[3]Increasing the number of ICL demonstrations will improve the performance of LLMs, but also much more costly. We only concatenate a single demonstration for CoT-ICL if not specified.

previous work (Oguz et al., 2022; Yu et al., 2023), we apply Dense Passage Retrieval (DPR) (Karpukhin et al., 2020) as the model architecture of the retrieval system. To obtain a robust retriever, we propose to utilize the constructed CoT as feedback to identify relevant passages. Specifically, we extract the last reasoning sub-question from the CoT rationale as the augmentation and concatenate it with the original question and the answer[4]. Then we apply the BM25 algorithm on the concatenated query and extract the top 100 related passages. We identify passages that contain entities present in both the question and answer as positive, while passages that only contain the answer or question entities are considered hard negatives. If no co-occurrence passage is found, we use the passage containing only the answer as positive to ensure the recall rate of the multi-answer question. We utilize Spacy[5] to recognize named entities in the query. Note that the feedback of LLM is only used for identifying positive/negative passages, we use the original questions to train our DPR model.

Fuse-in-Decoder Reader For our reader, we use the mainstream Fuse-in-Decoder architecture (Izacard and Grave, 2021) to train a Transformer (Vaswani et al., 2017) model. Specifically, given a question q and its top-N relevant passages P, the FiD reader first separately encodes each passage p_{q_i} concatenated with q:

$$P_i = \text{Encoder}(\text{Concat}[q,\ p_{q_i}]) \in \mathbb{R}^{L \times H} \quad (1)$$

Where L and H represent sequence length and hidden size, respectively. Then the token embeddings of all passages output from the encoder are concatenated and fed to the decoder to generate the final answer. Different from previous work, we employ all answers as training targets instead of selecting one randomly.

$$A = \text{Decoder}(\text{Concat}[P_1, P_2, ..., P_N]) \quad (2)$$

Verifier We train a Llama2-7B (Touvron et al., 2023b) with Parameter-Efficient-Fine-Tuning (PEFT) on the original KBQA training set as our verifier. Specifically, we adopt LoRA with a low rank equal to 16 for additional parameters.

During the training phase, we generate two training instances for each QA pair. The first instance randomly selects the answer from a different question, enabling the model to choose the correct one. The second instance randomly selects the answers from two other separate questions, encouraging the model to produce the correct answer:

$$a^+ = \text{Decoder}([q,\ a^+,\ a_1^-]) \quad (3)$$

$$a^+ = \text{Decoder}([q,\ a_2^-,\ a_3^-]) \quad (4)$$

During inference, the model takes the original sub-answers generated by LLM and the candidate answers generated by the retrieve-then-read pipeline as input, and outputs its preferred one. If neither answer is selected, the verifier will generate a new answer.

If not specified, greedy decoding is used for the Reader and the Verifier during inference.

# Data	WebQSP		CWQ	
	train	test	train	test
original	3098	1639	27625	3519
CoT collection	2888	1639	26695	3519

Table 1: Data statistics of original datasets and our CoT collections. After collection construction, we obtained 2888 and 26695 rationale data for WebQSP and CWQ, respectively.

3 Experiment

3.1 Dataset

We evaluate KD-CoT on two KBQA datasets: WebQSP (Yih et al., 2016) and ComplexWebQuestions (CWQ) (Talmor and Berant, 2018). We use the original datasets to train our external QA system, and use the constructed CoT collection to apply ICL on ChatGPT. The data statistics are shown in Table 1.

3.2 Experiment Settings

For our main experiment, we use ChatGPT as our backbone model (which is denoted as "LLM" in the subsequent sections of this paper) to interact with an external QA system. The QA system includes a BERT-base (Devlin et al., 2019) retriever, a T5-large (Raffel et al., 2020) reader, and a Llama2-7B verifier fine-tuned with LoRA (Hu et al., 2021) [6]. We prompt LLM to perform structured multi-round QA reasoning with demonstrations selected from our constructed CoT collection.

[4]Query mentioned below stands for <question, rationale question>, and BM25 searches the relevant passages on <question, rationale question, answers>

[5]https://spacy.io/

[6]All models are downloaded from Huggingface.

We train our retriever on a merged dataset of WebQSP and CWQ for saving the cost of embedding massive knowledge and use the same DPR architecture in the original paper (Karpukhin et al., 2020). The number of retrieved passages is 100 if not specified. The reader is also trained on the merged dataset to effectively tackle both single-hop and multi-hop question scenarios.

To show the effectiveness and correctness of our CoT collection, we also conduct experiments that involve fine-tuning smaller models on the constructed CoT data. We use Flan-T5-3B (Chung et al., 2022), T5-3B (Raffel et al., 2020), Llama2-7B, 13B, and compare the results with direct QA fine-tuning. To indicate the CoT paradigm that generates both rationale and answers, we incorporate a trigger phrase "Let's think step by step" into the sequence during training and inference.

Evaluation metric We evaluate our model based on metrics Hits@1 and F1, where Hits@1 focuses on the single top-ranked answer while F1 considers coverage of all the answers. To account for the fact that it's difficult to extract the desired answers from LLM's output, we adjust our evaluation criteria. We deem the generated results to be correct if they contain the ground truth answer.

3.3 Konwledge-Driven CoT Results

Table 2 reports the performance of our proposed KD-CoT. The results show that KD-CoT outperforms vanilla CoT ICL (denoted as "LLM$_{QA-CoT\ selected}$") by 8.0 and 5.1 points on WebQSP and CWQ, respectively. This highlights the effectiveness of interacting with the external QA system, as it enables the LLM to generate more accurate intermediate reasoning steps, leading to more precise final answers. We illustrate several cases in Appendix C.

It is not surprising that KD-CoT underperforms the fine-tuned SOTA models, as we utilize chatGPT as the backbone LLM and perform 1-shot ICL with the greedy decoding strategy (Bang et al., 2023; Sun et al., 2023). However, our main motivation is to "explore faithful reasoning in LLMs", and our method does improve the performance compared to its vanilla ICL reasoning output. As for our retrieve-then-read pipeline that is also not SOTA, here we offer the analysis:

1) UnikQA links entities and relations extracted from the query to the linearized KB text, resulting in more accurate and less noisy retrieved results and thus better performance on downstream QA tasks. However, this method is not applicable for multi-hop questions, as the head entity associated with the answer will not appear in the query.

2) DeCAF generates both logic forms and the target answers and utilizes the "beam search + re-rank" strategy to obtain the final answer, which is extremely computationally costly. Instead, we do not generate logic form and only utilize greedy decoding for all experiments to save inference costs. Even so, compared to their method without logic form under the "beam search + re-rank" strategy, our pipeline still achieves competitive or better results.

Despite not being SOTA, the faithful reasoning of LLMs is of great value, enabling researchers to construct high-quality CoT fine-tuning data.

To further demonstrate that our process of verifying and correcting sub-answers can lead to faithful reasoning so as to rectify previously incorrect responses, we tally the alterations in the count of correct and incorrect answers before and after undergoing our interactive framework. The results are shown in Figure 3. On WebQSP and CWQ, we correct 13.5% and 10.6% of questions that were incorrectly answered previously, while only 5.8% and 5.4% are modified to be incorrect.

Despite its efficiency in extracting sub-questions to interact with the external QA system, structured CoT also reduces its flexibility in formulating reasoning steps due to the structural constraint (Yao et al., 2023). Once the output generated is inadequately structured and unable to extract sub-questions, we consider it a failure in answering.

Method \ Dataset	**WebQSP**		**CWQ**
	Hit@1	F1	Hit@1
UnikQA (Oguz et al., 2022)	79.1	-	-
DeCAF (Yu et al., 2023)	80.7	77.1	67.0
DeCAF$_{w/o\ LF}$ (Yu et al., 2023)	74.2	49.5	47.9
Our $_{Retrieve-then-read}$	73.7	50.2	50.5
LLM $_{Retrieval\ 4-passages}$	52.4	38.2	26.9
LLM $_{QA\ pairs\ 4-shot}$	53.2	39.2	42.2
LLM $_{CoT\ fixed}$	50.3	37.8	34.0
LLM $_{QA-CoT\ fixed}$	56.6	42.5	42.4
LLM $_{QA-CoT\ selected}$	60.6	47.8	50.6
KD-CoT	68.6	52.5	55.7
KD-CoT$_{w/o\ Retrieve-then-read}$	66.8	49.4	49.2
KD-CoT$_{w/o\ Verifier}$	59.9	47.6	49.2

Table 2: Experimental results on WebQSP and CWQ. KD-CoT significantly outperforms the vanilla CoT ICL. The bottom two blocks are all conducted using ChatGPT.

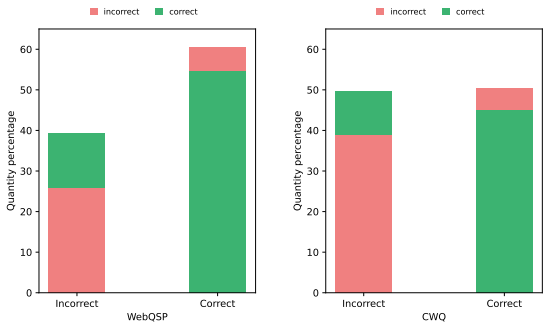


Figure 3: Quantity percentage of questions answered correctly/incorrectly by the LLM. The horizontal axis represents the state before passing through the interaction framework. Red and Green blocks represent the proportion of questions answered correctly/incorrectly after the entire interaction.

Method	WebQSP		CWQ	
	H / R@20	H / R@100	H / R @20	H / R@100
BM25	66.8 / 49.8	83.8 / 69.8	47.8 / 42.7	65.4 / 59.3
DeCAF-DPR	- / -	91.6 / 80.6	- / -	71.4 / 65.6
FBA-DPR	89.0 / 75.6	95.4 / 88.4	68.7 / 62.5	81.3 / 76.5
w/o wiki	88.9 / 74.2	94.8 / 86.3	65.0 / 58.5	77.8 / 72.6

Table 3: Retrieval results on WebQSP and CWQ. H@N and R@N stand for the answer hits rate and recall rate of Top-N retrieved passages, respectively. DPR results are copied from (Yu et al., 2023), BM25, and FBA-DPR results are obtained in our setting.

Consequently, the capability of LLM might be underestimated.

3.4 Retrieval Results

We evaluate the effectiveness of our FeedBack-Augmented DPR (FBA-DPR) in Table 3. It can be seen that FBA-DPR significantly outperforms previous results in (Yu et al., 2023) on both WebQSP and CWQ, achieving 3.8 and 9.9 points of improvement on Hit@100, and 7.8 and 9.9 points on Recall@100. For a fair comparison, we also conduct retrieval on external knowledge resources excluding Wikipedia passages, which still significantly surpasses the previous results. This further demonstrates the effectiveness of our proposed method.

3.5 CoT Fine-tuning Results

This experiment is conducted to validate the correctness of the CoT collection we constructed. Additionally, we aim to explore whether smaller models are capable of acquiring reasoning ability from such structured CoT data. We conduct the experiment under two different settings: **Direct Fine-tuning** and **CoT Fine-tuning**. For **Direct Fine-tuning** we train the model on the original QA pairs, where the model takes the questions as input and directly generates the answers. The results are reported in Table 4.

Model \ Dataset	WebQSP		CWQ	
	Direct	CoT	Direct	CoT
T5-3B	40.8	41.9	39.4	38.6
FlanT5-3B	46.1	47.0	50.5	43.8
Llama2-7B-LoRA	63.8	64.1	48.4	45.1
Llama2-13B-LoRA	75.0	73.8	53.5	54.7

Table 4: Comparison of Direct Fine-tuning and CoT Fine-tuning. Hits@1 score is reported.

We observe that for smaller models, fine-tuning LMs with CoT rationales slightly outperforms Direct Fine-tuning for solving simpler questions. In complex multi-hop question scenarios, CoT fine-tuning brings negative gains. This might be because 1) The reasoning procedure of the original LM differs from that of the CoT collection. Fine-tuning the LM may potentially disrupt the original knowledge, resulting in a degradation in performance; 2) LMs still struggle to generate faithful multi-step reasoning even fine-tuned with CoT when solving knowledge-intensive tasks. However, when the number of model parameters is increased to 13B, the benefits of CoT fine-tuning become evident, yielding better results than direct fine-tuning on the CWQ dataset. This suggests the larger the model, the more reasoning capacity it can acquire from the CoT fine-tuning, and demonstrates the correctness of our constructed collection.

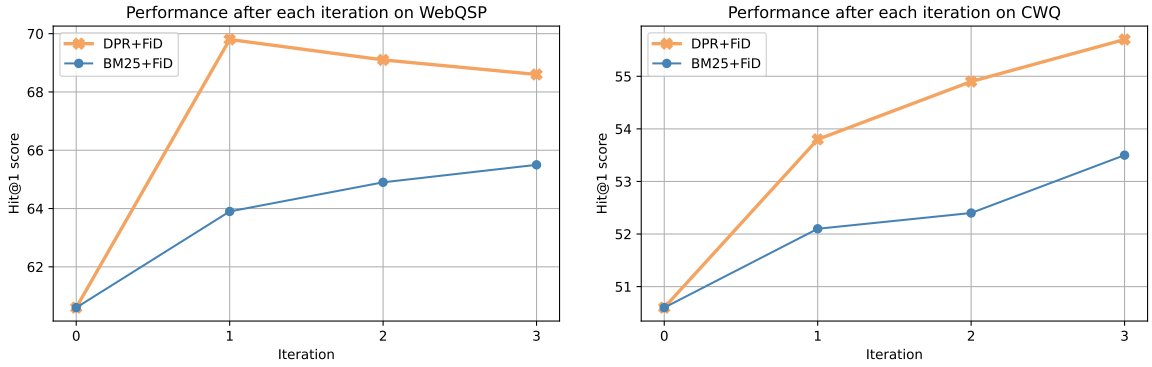
3.6 Analysis & Ablation Study

This section aims to address the following question through analysis and ablation experiments.

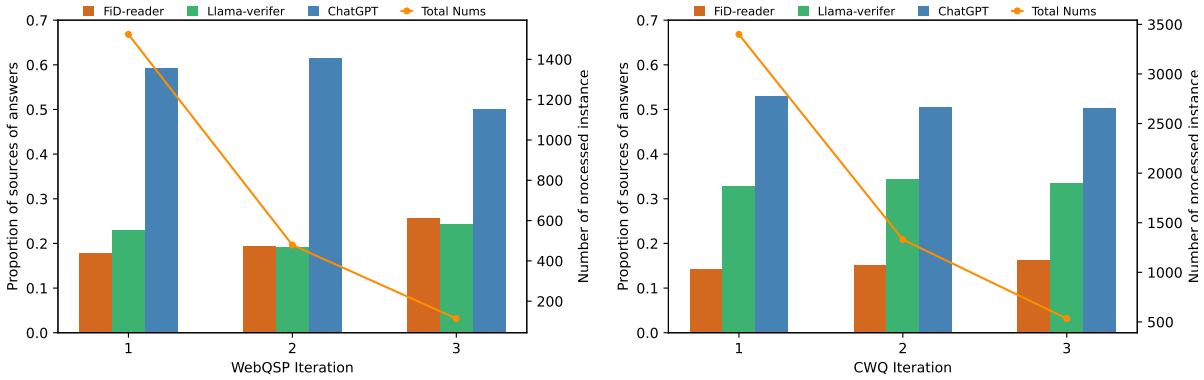
Benefits of structured CoT and CoT collection? To discuss the benefits of structured multi-round QA rationale, and to highlight the significance of the CoT collection we construct, we evaluate the following model settings, with the name corresponding to the rows in Table 2:

- LLM $_{Retrieval\ 4-passages}$ We roughly concatenate the top-ranked retrieved passages with the question as input and instruct LLM to answer the question.

- LLM $_{QA\ pairs\ 4-shot}$ We utilize other QA pairs with the highest cosine similarity to the



(a) LLM performance after each iteration of interaction. The highest performance is achieved in the first iteration for WebQSP and the last iteration for CWQ, as WebQSP is primarily comprised of single-hop questions, whereas CWQ contains more complex multi-hop questions.



(b) Answer source during each iteration. In most cases, the verifier prefers sub-answers output by ChatGPT, about half of the sub-answers are modified by the external QA system in each iteration.

Figure 4: a) LLM performance after each iteration of interaction. b) Answer source during each iteration.

target question as the demonstrations to perform ICL.

- LLM $_{CoT\ fixed}$ We manually design unstructured rationales aligned with the content of our structured CoT, and utilize them as demonstrations to prompt LLM. The in-context demonstration is selected within human-annotated unstructured rationales.

- LLM $_{QA-CoT\ fixed}$ The in-context demonstration is selected within human-annotated structured rationales.

- LLM $_{QA-CoT\ selected}$ The in-context demonstration is selected within our constructed CoT collection.

To align with one-shot CoT ICL, we restrict the input length and concatenate only 4 passages/QA pairs as the context that fed into LLM. Experimental results are shown in Table 2. We observe that LLM achieves superior performance when utilizing structured rationale as the demonstration, outperforming other ICL methods. This suggests that our proposed multi-round QA format rationale is more effective in unleashing LLM's reasoning capability.

Directly concatenating the retrieved knowledge does not have a positive contribution to the model's ability, especially in complex multi-hop question scenarios, and it performs the worst among all ICL methods. The low accuracy in knowledge retrieval could be the cause of this, as evidenced by the Hit@20 and Recall@20 scores reported in Table 3, which are only 68.7 and 62.5 respectively. The top-4 contexts might contain a significant amount of noise, which is not beneficial for the ICL of LLMs. It's worth noting that previous work primarily utilized a similar methodology(Yao et al., 2023; Xu et al., 2023).

By employing our constructed CoT collection, we further improve the LLM's ability, highlighting the effectiveness and necessity of the collection construction.

Benefits of QA system?

To assess the effectiveness of the QA system, we conduct two supplementary experiments by removing the retrieve-then-read pipeline and the verifier separately. The results are shown in Table 2. We observe that the performance degrades when the retrieve-then-read pipeline is removed, showing the importance of accessing external knowledge for precise sub-answers generation. Moreover, the performance without the verifier is worse, showing that in certain cases the sub-answers generated by the LLM are superior. Further combining the output of the reader and LLM to generate better answers is important for improving performance.

We further investigate the performance gain after each iteration and count the source of the modified answer. The results are shown in Figure 4. As can be seen in Figure 4(a), the highest performance of LLM is achieved in the first iteration for WebQSP and the last iteration for CWQ, as WebQSP is primarily comprised of single-hop questions, whereas CWQ contains more complex multi-hop questions. We also observe that LLM tends to produce redundant inference steps despite being able to answer questions within two hops of reasoning. This leads to the necessity of second and third iterations to terminate the CoT while solving WebQSP questions. An extra Halter (Creswell and Shanahan, 2022) to determine whether the current reasoning step can answer the questions can be a possible method for solving this issue. As the number of iterations increases, the performance on the CWQ dataset also improves. This suggests that our interaction framework can assist the model in better reasoning for complex multi-hop questions.

Figure 4(b) shows the source of modified answers. In most cases, the verifier will keep the original sub-answers generated by ChatGPT, about half of the sub-answers are modified and fed to the next iteration. This implies that a robust reader capable of producing varied and accurate answers is crucial in fully unleashing the potential of LLM.

4 Related Work

4.1 Chain-of-thought Prompting

Wei et al. (2023) have shown the Chain-of-Thought (CoT) to be effective in enhancing LLMs reasoning. Several studies have been conducted to improve the effectiveness. For example, some studies focus on how to make LLM generate more accurate and reliable chains of thought (He et al., 2022; Wang et al., 2023b; Lyu et al., 2023), while some works investigate more efficient ways of generating chains of thought to unleash the potential of LLM reasoning (Creswell et al., 2022; Zhou et al., 2023; Jin and Lu, 2023). As LLMs are confined to the knowledge learned from the training corpus, extensive efforts have been made recently to facilitate LLMs in dynamically interacting with the real world (Yao et al., 2023; Zhao et al., 2023; Peng et al., 2023) or KGs (Baek et al., 2023; Li et al., 2023) to obtain the information required for model reasoning. These works follow a fixed pipeline that retrieves extra information to augment the LLM prompt. In contrast, we make use of a more advanced retriever-reader pipeline to furnish the model with more accurate and targeted knowledge.

4.2 Knowledge Base Question Answering

The retrieve-then-read pipeline is a commonly employed technique for solving KBQA tasks. Certain studies concentrate on enhancing the retriever's efficiency (Karpukhin et al., 2020; Izacard et al., 2021; Chuang et al., 2023), whereas others prioritize the reader's performance (Izacard and Grave, 2021; Yu et al., 2022) or both (Dong et al., 2023). Additionally, some studies delve into the incorporation of structured knowledge from the knowledge base into the QA system (Oguz et al., 2022; Yu et al., 2023), or utilize contexts generated by LLM as knowledge enhancement (Zhang et al., 2023). Previous works have formed an efficient system of information retrieval, condensing the extracted knowledge corpus into brief statements. Therefore, our research integrates this system with LLMs to offer necessary knowledge and aid the LLMs in producing more dependable chains of thought.

5 Conclusion

In this paper, we investigate the faithful reasoning of LLMs on knowledge-intensive KBQA tasks. We propose a Knowledge-Driven Chain-of-Thought framework to improve the reasoning performance of LLMs. Through experiments on knowledge-intensive KBQA tasks, we show that KD-CoT leads to superior performance with interpretable inference steps. We also present a CoT collection on the KBQA datasets that can be utilized for CoT fine-tuning and few-shots ICL. Additionally, we propose a feedback-augmented retriever that can efficiently access external knowledge, which results in a substantial improvement in Hit and recall scores in the multi-hop question scenario.

Limitations

Although our method can efficiently access external knowledge and correct the sub-answers generated by LLMs, the final performance of LLM still leaves behind the current SOTA. This could be caused by: 1) The inability of the QA system to generate precise answers for all sub-questions, as our simply designed reader achieves only 73.7 Hit@1 on the WebQSP dataset. 2) The sub-question hallucination. LLM can still hallucinate producing sub-question despite our corrections to sub-answers. Future work can focus on supervising both the intermediate reasoning questions and answers.

Besides, with a limited budget, we only use ChatGPT for 1-shot ICL and greedy decoding in CoT reasoning. As budget allows, performance can be improved by 1) using a stronger LLM like GPT-4; 2) adding more demonstrations for ICL or generating multiple reasoning paths and all interact with the QA system.

References

Jinheon Baek, Alham Fikri Aji, and Amir Saffari. 2023. Knowledge-augmented language model prompting for zero-shot knowledge graph question answering. In *Proceedings of the 1st Workshop on Natural Language Reasoning and Structured Explanations (NLRSE)*, pages 78–106, Toronto, Canada. Association for Computational Linguistics.

Yejin Bang, Samuel Cahyawijaya, Nayeon Lee, Wenliang Dai, Dan Su, Bryan Wilie, Holy Lovenia, Ziwei Ji, Tiezheng Yu, Willy Chung, Quyet V. Do, Yan Xu, and Pascale Fung. 2023. A multitask, multilingual, multimodal evaluation of chatgpt on reasoning, hallucination, and interactivity.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: A collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, SIGMOD '08, page 1247–1250, New York, NY, USA. Association for Computing Machinery.

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.

Yung-Sung Chuang, Wei Fang, Shang-Wen Li, Wen tau Yih, and James Glass. 2023. Expand, rerank, and retrieve: Query reranking for open-domain question answering.

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.

Antonia Creswell and Murray Shanahan. 2022. Faithful reasoning using large language models.

Antonia Creswell, Murray Shanahan, and Irina Higgins. 2022. Selection-inference: Exploiting large language models for interpretable logical reasoning.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Guanting Dong, Rumei Li, Sirui Wang, Yupeng Zhang, Yunsen Xian, and Weiran Xu. 2023. Bridging the kb-text gap: Leveraging structured knowledge-aware pre-training for kbqa.

Zhengxiao Du, Yujie Qian, Xiao Liu, Ming Ding, Jiezhong Qiu, Zhilin Yang, and Jie Tang. 2022. Glm: General language model pretraining with autoregressive blank infilling.

Hangfeng He, Hongming Zhang, and Dan Roth. 2022. Rethinking with retrieval: Faithful large language model inference.

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

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2021. Unsupervised dense information retrieval with contrastive learning.

Gautier Izacard and Edouard Grave. 2021. Leveraging passage retrieval with generative models for open domain question answering.

Ziqi Jin and Wei Lu. 2023. Tab-cot: Zero-shot tabular chain of thought.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen tau Yih. 2020. Dense passage retrieval for open-domain question answering.

Xingxuan Li, Ruochen Zhao, Yew Ken Chia, Bosheng Ding, Shafiq Joty, Soujanya Poria, and Lidong Bing. 2023. Chain-of-knowledge: Grounding large language models via dynamic knowledge adapting over heterogeneous sources.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2023. Lost in the middle: How language models use long contexts.

Qing Lyu, Shreya Havaldar, Adam Stein, Li Zhang, Delip Rao, Eric Wong, Marianna Apidianaki, and Chris Callison-Burch. 2023. Faithful chain-of-thought reasoning.

Barlas Oguz, Xilun Chen, Vladimir Karpukhin, Stan Peshterliev, Dmytro Okhonko, Michael Schlichtkrull, Sonal Gupta, Yashar Mehdad, and Scott Yih. 2022. Unik-qa: Unified representations of structured and unstructured knowledge for open-domain question answering.

Baolin Peng, Michel Galley, Pengcheng He, Hao Cheng, Yujia Xie, Yu Hu, Qiuyuan Huang, Lars Liden, Zhou Yu, Weizhu Chen, and Jianfeng Gao. 2023. Check your facts and try again: Improving large language models with external knowledge and automated feedback.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Kai Sun, Yifan Ethan Xu, Hanwen Zha, Yue Liu, and Xin Luna Dong. 2023. Head-to-tail: How knowledgeable are large language models (llm)? a.k.a. will llms replace knowledge graphs?

Alon Talmor and Jonathan Berant. 2018. The web as a knowledge-base for answering complex questions. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 641–651, New Orleans, Louisiana. Association for Computational Linguistics.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023a. Llama: Open and efficient foundation language models.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023b. Llama 2: Open foundation and fine-tuned chat models.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need.

Jianing Wang, Qiushi Sun, Nuo Chen, Xiang Li, and Ming Gao. 2023a. Boosting language models reasoning with chain-of-knowledge prompting.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023b. Self-consistency improves chain of thought reasoning in language models.

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.

Shicheng Xu, Liang Pang, Huawei Shen, Xueqi Cheng, and Tat-Seng Chua. 2023. Search-in-the-chain: Towards accurate, credible and traceable large language models for knowledge-intensive tasks.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. 2023. React: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations*.

Wen-tau Yih, Matthew Richardson, Chris Meek, Ming-Wei Chang, and Jina Suh. 2016. The value of semantic parse labeling for knowledge base question answering. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 201–206, Berlin, Germany. Association for Computational Linguistics.

Donghan Yu, Sheng Zhang, Patrick Ng, Henghui Zhu, Alexander Hanbo Li, Jun Wang, Yiqun Hu, William Wang, Zhiguo Wang, and Bing Xiang. 2023. Decaf: Joint decoding of answers and logical forms for question answering over knowledge bases.

Donghan Yu, Chenguang Zhu, Yuwei Fang, Wenhao Yu, Shuohang Wang, Yichong Xu, Xiang Ren, Yiming Yang, and Michael Zeng. 2022. Kg-fid: Infusing knowledge graph in fusion-in-decoder for open-domain question answering.

Yunxiang Zhang, Muhammad Khalifa, Lajanugen Logeswaran, Moontae Lee, Honglak Lee, and Lu Wang. 2023. Merging generated and retrieved knowledge for open-domain qa.

Ruochen Zhao, Xingxuan Li, Shafiq Joty, Chengwei Qin, and Lidong Bing. 2023. Verify-and-edit: A knowledge-enhanced chain-of-thought framework. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5823–5840, Toronto, Canada. Association for Computational Linguistics.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, and Ed Chi. 2023. Least-to-most prompting enables complex reasoning in large language models.

A Implementation details

We present here in detail the parameter settings we used to train the QA system and to perform CoT fine-tuning.

Model	# Params	# Total Params
BERT-base-uncased	110M	110M
T5-large	770M	770M
Llama2-7B_lora	12M	7B
T5-3B	3B	3B
flan-T5-3B	3B	3B
Llama2-7B_lora	12M	7B

Table 5: Models utilized and their number of parameters. # Params represents the trainable parameters.

	BERT	T5-large	Llama-7B	T5-3B	Flan-T5-3
Lr	2e-5	5e-5	1e-4	1e-4	1e-4
Batch Size	128	16	32	32	32
Epoch	40	-	5	5	5
Clip Norm	2.0	1.0	1.0	1.0	1.0

Table 6: Hyper-params settings for training QA system and for CoT fine-tuning.

Specifically, we train our model using Deepspeed, with pytorch==2.0.0, peft==0.2.0, and transformers==4.29.1. For models larger than 1B, we train with precision bfloat16. For the optimizer and learning scheduler, we apply AdamW with Beta=[0.9, 0.95] and LinearDecay with a warmup ratio equal to 0.1. All experiments except the inference of ChatGPT are conducted on 8 $\times$ 40G Nvidia A100.

B Demonstration illustration

We add final answers and composition answers as "Hint" before structured rationale to construct our CoT collection. When conducting in-context learning during inference, we eliminate the "Hint" from the chosen demonstration. We provide two examples as illustrated in Figure 5, one for constructing the CoT collection and the other for performing ICL on large language models during inference.

CoT collection

Instruction: Solve a question answering task with interleaving Thought, Action, Answer steps. Thought can reason about the current situation, and Action can be three types:
(1) Question[question], which returns a sub-question with single answer that helps solve the original problem.
(2) Multi_Answer_Question[keyword], which returns a sub-question with multiple answers.
(3) Finish[answer], which returns the answer and finishes the task.

Question: Who are the children of Ike and the woman who composed Nutbush City Limits?
Hint: answer: ["Ronnie Turner", "Raymond Craig Turner"], composition_answer: ["tina turner"]
Thought 1: I need to find out who Ike is and who composed Nutbush City Limits, and then determine their children.
Action 1: Question[Who is Ike?]
Answer 1: Ike Turner
Thought 2: Ike Turner is the person being referred to. I now need to find out who composed Nutbush City Limits.
Action 2: Question[Who composed Nutbush City Limits?]
Answer 2: Tina Turner
Thought 3: Tina Turner composed Nutbush City Limits. I now need to find out who the children of Ike and Tina Turner are.
Action 3: Multi_Answer_Question[Who are the children of Ike and Tina Turner?]
Answer 3: Ronnie Turner and Raymond Craig Turner
Thought 4: I found the children of Ike and Tina Turner: Ronnie Turner and Raymond Craig Turner.
Action 4: Finish[Ronnie Turner and Raymond Craig Turner.]

Question: What disease did the music producer of Sometimes Late at Night have?
Hint: answer: ["Mysophobia"], composition_answer: ["michael jackson"]

LLM inference

Instruction: Solve a question answering task with interleaving Thought, Action, Answer steps. Thought can reason about the current situation, and Action can be three types:
(1) Question[question], which returns a sub-question with single answer that helps solve the original problem.
(2) Multi_Answer_Question[keyword], which returns a sub-question with multiple answers.
(3) Finish[answer], which returns the answer and finishes the task.

Question What movie did Ron Howard direct that was produced designed by James L. Schoppe? Thought 1 I need to find out which movie Ron Howard directed that was designed by James L. Schoppe.
Action 1 Question[Which movie did Ron Howard direct that was designed by James L. Schoppe?]
Answer 1 Gung Ho
Thought 2 Ron Howard directed the movie Gung Ho that was designed by James L. Schoppe.
Action 2 Finish[Gung Ho]

Question What movie involving Toonexplainers that Ron Howard worked on?

Figure 5: Input for constructing CoT collection (up) and input for LLM inference (down).

C Case Analysis

We present in Figure 6 and 7 several cases before and after our Knowledge-Driven Chain-of-Thought.

We observe that by correcting the sub-answers, the subsequent reasoning steps become more dependable and accurate, ultimately resulting in the correct final answers.

CWQ case

Question: What founder of New York University had a position in the 4th United States Congress?

Thought 1 I need to find out which founder of New York University held a position in the 4th United States Congress.
Action 1 Multi_Answer_Question[founder + New York University + 4th United States Congress]
Answer 1 1. Albert Gallatin 2. Samuel L. Mitchill 3. James Kent 4. Gouverneur Morris Albert Gallatin

Thought 2 There are four founders of New York University who held a position in the 4th United States Congress - Albert Gallatin, Samuel L. Mitchill, James Kent, and Gouverneur Morris. Albert Gallatin is the founder of New York University who held a position in the 4th United States Congress.
Action 2 Question[Which one of the four founders of New York University held a position in the 4th United States Congress?]Finish[Albert Gallatin]
Answer 2 Gouverneur Morris

Thought 3 Gouverneur Morris is the founder of New York University who held a position in the 4th United States Congress.
Action 3 Finish[Gouverneur Morris]

Question: Who was the Arizona's Governor in 2009 who also suffered with Breast cancer?

Thought 1 I need to find out who was the Arizona's Governor in 2009 and also suffered with Breast cancer.
Action 1 Question[Who was the Arizona's Governor in 2009?]
Answer 1 Jan Brewer Janet Napolitano

Thought 2 Jan Brewer was the Arizona's Governor in 2009. I now need to find out if she suffered with Breast cancer. Janet Napolitano was the Arizona's Governor in 2009. Now, I need to find out if she suffered from Breast cancer.
Action 2 Multi_Answer_Question[Jan Brewer Breast cancer] Question[Did Janet Napolitano suffer from Breast cancer]
Answer 2 Jan Brewer was diagnosed with Breast cancer in 2006. Breast cancer.

Thought 3 Jan Brewer was diagnosed with Breast cancer in 2006. I now need to find out if she continued to suffer from it in 2009. Janet Napolitano suffered from Breast cancer. I need to find out if she had it in 2009.
Action 3 Question[Did Jan Brewer continue to suffer from Breast cancer in 2009?] Question[Did Janet Napolitano have Breast cancer in 2009?]
Answer 3 No Yes

Thought 4 Jan Brewer was the Arizona's Governor in 2009 and was diagnosed with Breast cancer in 2006. but did not continue to suffer from it in 2009. Janet Napolitano was the Governor of Arizona in 2009 and she suffered from Breast cancer during that time.
Action 4 Finish[Jan Brewer was the Arizona's Governor in 2009 and was diagnosed with Breast cancer in 2006. but did not continue to suffer from it in 2009.] Finish[Janet Napolitano was the Arizona's Governor in 2009 and suffered from Breast cancer.]

Figure 6: Cases of CWQ. Red and Green blocks represent the original hallucinations of LLM and the faithful reasoning after sub-answer correction. The yellow block signifies that the answer generated by the QA system is not entirely precise, but it does not impact the inference of the subsequent models.

WebQSP case

Question: What episode of how i met your mother is carrie underwood on?

Thought 1 I need to find out which episode of How I Met Your Mother features Carrie Underwood.
Action 1: Multi_Answer_Question[Carrie Underwood How I Met Your Mother episode]
Answer 1 How I Met Your Mother: Sandcastles in the Sand (Season 3, Episode 16) - How I Met Your Mother: Ultimate Playbook (Season 9, Episode 2) Hooked

Thought 2 Carrie Underwood appeared in two episodes of How I Met Your Mother: ""Sandcastles in the Sand"" in season 3 and ""Ultimate Playbook"" in season 9.The episode of How I Met Your Mother featuring Carrie Underwood is called Hooked.
Action 2: Finish[""Sandcastles in the Sand"" and ""Ultimate Playbook""]Hooked

---

Question: What type of cancer did eva peron have?

Thought 1 I need to find out what type of cancer Eva Peron had.
Action 1 Multi_Answer_Question[cancer type in Eva Peron]
Answer 1 (1) Cervical Cancer, (2) Uterine Cancer, (3) Ovarian Cancer

Thought 2 Eva Peron had one of three types of cancer: cervical, uterine, or ovarian.
Action 2 Question[Which type of cancer did Eva Peron have?]
Answer 2 Uterine CancerCervical Cancer

Thought 3 Eva Peron had uterine cancer. Eva Peron had cervical cancer.
Action 2: Finish[Uterine Cancer]Cervical Cancer

Figure 7: Cases of WebQSP, by applying KD-CoT we rectify the sub-answers and make the reasoning of LLM more faithful. Red and Green blocks represent the original hallucinations of LLM and the faithful reasoning after sub-answer correction.