Enhancing the Completeness of Rationales for Multi-Step Question Answering

Shangzi Xue
State Key Laboratory of Cognitive Intelligence, University of Science and Technology of China
Hefei, China
xueshangzi@mail.ustc.edu.cn

Zhenya Huang*
State Key Laboratory of Cognitive Intelligence, University of Science and Technology of China
Hefei, China
huangzhy@ustc.edu.cn

Xin Lin
State Key Laboratory of Cognitive Intelligence, University of Science and Technology of China
Hefei, China
linx@mail.ustc.edu.cn

Jiayu Liu
State Key Laboratory of Cognitive Intelligence, University of Science and Technology of China
Hefei, China
jy251198@mail.ustc.edu.cn

Longhu Qin
State Key Laboratory of Cognitive Intelligence, University of Science and Technology of China
Hefei, China
qlonghu@mail.ustc.edu.cn

Tianhuang Su
Guangdong OPPO Mobile Telecommunications Corp., Ltd
Shenzhen, China
sutianhuang@oppo.com

Haifeng Liu
University of Science and Technology of China
Hefei, China
bladehliu@gmail.com

Qi Liu
State Key Laboratory of Cognitive Intelligence, University of Science and Technology of China
Hefei, China
qiliuql@ustc.edu.cn

Abstract

Learning to answer multi-step complex questions requires machines to perform like a human to think and reason step by step, which is one of the core abilities of a question answering system. Recent advancements have revealed that large language models exhibit remarkable reasoning capabilities by generating intermediate chain-of-thought rationales. However, the completeness of their rationales lacks assurance as they are susceptible to omitting steps and making factual errors. In this paper, drawing inspiration from human-like reasoning processes in answering multi-step questions, we explicitly plan the rationales to ensure their completeness. We propose a two-stage Decomposition-Evaluation (Dec-Eval) framework including a step decomposition stage and a rationale generation stage. Specifically, in the first stage, we decompose the complex question into simpler sub-ones and simulate a human's ability to grasp logical clues to ensure the integrity of step planning. Then, in the second stage, based on the sub-questions, we generate and evaluate rationales step by step. Both stages work together organically, improving the completeness of rationales and the accuracy of the answer. To further control the question answering process, we propose a novel knowledge injection mechanism that incorporates external knowledge to guide both stages. Extensive experiments on three challenging multi-step QA datasets demonstrate that Dec-Eval can explicitly generate more logical rationales, and significantly improve the reasoning performances of different backbone models.

CCS Concepts

• **Computing methodologies** → **Natural language processing**.

Keywords

Multi-step Question Answering; Question Decomposition; Rationale Evaluation

ACM Reference Format:
Shangzi Xue, Zhenya Huang, Xin Lin, Jiayu Liu, Longhu Qin, Tianhuang Su, Haifeng Liu, and Qi Liu. 2024. Enhancing the Completeness of Rationales for Multi-Step Question Answering. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management (CIKM '24), October 21–25, 2024, Boise, ID, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3627673.3679660

1 Introduction

Multi-step question answering requires machines to simulate human-like abilities that not only understand the questions but also decompose the reasoning process into multiple steps [31]. It is one of the core research areas of natural language processing, which consolidate intelligent question answering and decision making systems [21, 34, 50]. Figure 1 shows an example question in ScienceQA benchmark [21]. The task often inputs a question (e.g.,"What do these...?") and asks for the answer. We need to engage in logical

*Corresponding Author.

CIKM '24, October 21–25, 2024, Boise, ID, USA

ACM ISBN 979-8-4007-0436-9/24/10
https://doi.org/10.1145/3627673.3679660

reasoning and formulate the correct "rationales" (a1), comprising three steps in (a2). Specifically, we should first judge whether the "breaking a piece of glass" or the "butter melting on a hot day" is a physical change (Step1 and Step2), and then draw a conclusion about their commonalities to determine the final answer (Step3).

In the literature, early methods on multi-step question answering use end-to-end ways including structures like memory network [42], graph neural network [36], etc. Next, some solutions try to solve question by breaking the process into sub-questions [25, 29].Recent studies have utilized chain-of-thought (CoT) ability in LLMs to guide the reasoning process [11, 41]. As shown in Figure 1(b1), one of the SOTA models MMCoT [49] can directly generate intermediate CoTs to predict the answer. Such rationales help us better understand the model and the reasoning process.

While CoT approaches perform well, their generated rationales still lack completeness. Specifically, existing methods often suffer from deficiencies in logical steps when answering questions. A closer look at Figure 1(a2) and Figure 1(b2) reveals that MMCoT [49] misses the essential Step 2 in (a2). This incompleteness also leads to factual errors, for instance, it misclassifies "Breaking a piece of glass" as a "chemical change" ((b2), Step 2) instead of the "physical" fact. Furthermore, these issues highlight the necessity of an evaluation mechanism for rationales, a consideration often overlooked by methods like Least-to-Most [51].

To address the above issues, inspired by the question decomposition and reasoning simplification theories [26, 33, 43], we present a novel Decomposition-Evaluation (Dec-Eval) framework for multi-step question answering. Dec-Eval excels in planning and generating more comprehensive rationales, which includes a Step Decomposition (SD) stage and a Rationale Generation (RG) stage. Specifically, in the SD stage, we decompose a question into simpler ones by a SubGen module which simulates human-like ability to grasp logically key clues. Then, in the RG stage, we generate and aggregate rationales step by step while using an elaborate relevance evaluation strategy to determine if each rationale need to be retained. Moreover, to further control the reasoning process, we propose a knowledge injection mechanism (KIM) that integrates information from external knowledge to guide both the Step Decomposition and the Rationale Generation stages. Finally, the answer is extracted from the generated rationales. We conduct experiments on three challenging multi-step question answering datasets. Experimental results demonstrate that Dec-Eval is a general framework which can improve answer accuracy for both small models and LLMs, generating more logical rationales. Our main contributions are:

- In this paper, we propose a general framework for multi-step question answering, which can simulate human reasoning by planning, generating and evaluating rationales. Several existing models can serve as its backbone and benefit from it to improve multi-step question answering ability.
- We design a novel SubGen module which can grasp and utilize logically key clues to decompose questions into sub-questions. These sub-questions serve as reasoning steps for enhancing the logical coherence of the rationales.
- We employ a multi-step rationale generation procedure while using a relevance evaluation strategy to enhance the quality of the rationales, which improves the reasoning accuracy.

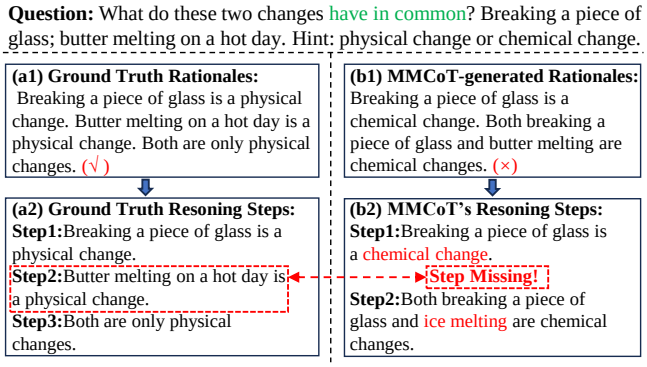

Figure 1: A multi-step question and its reasoning steps. The reasoning steps generated by MMCoT(with T5-Large backbone) misses the Step2 "Butter melting on a hot day is a physical change", and also makes a factual error about "ice melting" (... ice melting are chemical changes.).

2 Related Work

Multi-step Question Answering. Multi-step question answering tasks have been used for evaluating the reasoning ability and interpretability of a question answering system [7, 8, 21]. Traditional studies mainly employ an end-to-end way to directly infer the answers without intermediate steps. For example, UnifiedQA [9], which was pretrained on a number of existing question answering datasets, focused on developing a generative method to infer answers; Raffel et al. [32] employed a pretraining-finetuning paradigm to construct a text-to-text generation framework (T5) for various question answering tasks. There are also other end-to-end methods that use memory network [42] and graph neural network [5, 28, 36] to answer multi-step questions. In recent years, question decomposition approaches are popular in solving multi-step problems by decomposing the question into sub ones [25, 38]. By solving these simpler sub-questions respectively, models can improve their performances in predicting the correct answers. For example, Min et al. [25] focused on directly training a model to produce sub-questions using question spans; Perez et al. [29] used a text-to-text model to generate sub-questions for HotpotQA [46]; BREAK [44] followed an alternative paradigm of collecting full question decomposition meaning representations (QDMR) annotations. While these existing question decomposition methods can be effective, they either rely on costly human annotations or neglect some logical features in the questions, which may not generalize to domains with new decomposition operations.

Currently, with the remarkable progress of large language models (LLMs) in question answering tasks, some studies utilize LLMs' exceptional reasoning and generalization ability to perform multi-step reasoning [10, 51], which have achieved exceptional performances on various datasets. For example, Lu et al. [21] uses GPT-3 to learn to generate lectures and explanations as the chain of thought (CoT) to mimic the multi-step reasoning process when answering ScienceQA [21] questions; Lu et al. [22] proposed a plug-and-play compositional reasoning framework to plan the execution sequence of different tool-using steps by GPT-4, which got promising results

on ScienceQA [21] dataset. Least-to-most [51] sequentially breaks down and solves the original multi-step question, the solution of the current sub-question is facilitated by the previous one. Tree-of-Thoughts (ToT) prompting [47] proposed a tree-based searching paradigm for LLMs to perform deliberate reasoning by generating multiple reasoning path. Self-Refine [24] uses the same LLM to provide feedback for its output, and refine the output iteratively.

Chain-of-Thought Reasoning. Chain-of-Thought (CoT) Reasoning was first proposed by [41] to explore the complex reasoning ability of large language models. By prompting large language models (LLMs) with human-annotated intermediate reasoning steps, i.e., demonstrations or rationales, before reaching the answer, their performance on multi-step complex questions is significantly improved. Currently, Zero-shot-CoT [11] is widely used in various question answering tasks by asking LLMs to generate reasoning steps before answer, which improves the interpretability and accuracy of reasoning. CoT methods with LLMs can be categorized into two major research lines: optimizing the demonstrations [12, 23, 52] or optimizing the rationales [1, 10, 40, 51]. Recent advancements extend the application of CoT beyond large models to smaller language models [35], demonstrating broader applicability and scalability of this approach. E.g., Lu et al. [21] fine-tuned a T5 model with human-annotated rationales; Zhang et al. [49] developed a two-stage framework called MMCoT, which separates the generation of rationales from the process of answer inference. This separation allows the model to better leverage the generated rationales as intermediate steps in CoT, thus enhancing its performance on complex reasoning tasks. The MMCoT framework notably exceeds the capabilities of the previous state-of-the-art LLM, GPT4, and even surpasses human performance on the ScienceQA benchmark [21]. These advancements highlights the potential of CoT in enhancing the performance of both large and small language models.

Our work differs from previous methods as follows. First, previous CoT-based methods usually generate the entire rationales in one single process. While effective in straightforward scenarios, CoT often struggles with more complex problems where multiple, distinct reasoning steps are necessary. In contrast, our method follows a multi-step rationale generation procedure based on decomposed sub-questions. We also innovatively introduced "logical clues" into the problem to guide the question decomposition process. Second, recent studies, like Least-to-Most [51], use language models to reason step-by-step and sequentially generate rationales before reaching the final answer. These studies often neglect to perform thorough evaluations of the rationales, focusing primarily on the end results rather than the reasoning path. In contrast, we incorporate a relevance evaluation technique that actively assesses the rationales at each step of generation. It not only ensures the logical coherence and relevance of the rationales but also contributes to the overall completeness and robustness of the reasoning process. Third, approaches such as Self-Refine [24] aim to enhance the completeness of rationales through self-correction. However, these approaches frequently neglect the significance of integrating external knowledge sources. In contrast, our approach introduces a novel knowledge injection mechanism that effectively mitigates the occurrence of factual inaccuracies within the rationales. It not only bolsters the robustness of the reasoning steps but also significantly contributes to the enhancement of the overall answer accuracy.

3 Dec-Eval Framework

3.1 Problem Formulation

In this paper, we focus on the multi-step question answering task. The input of the task is a question context qc (e.g., "What do these two changes ... Hint: ... chemical change." in Figure 2). To predict the correct answer A (e.g., "Both are only physical changes."), one should form a multi-step reasoning process based on qc. Here, we define the multi-step question answering process as follows: First, we use information in qc to generate an ordered rationale list $rationales = (rationale_1, rationale_2, \ldots, rationale_n)$ in Figure 2, where each $rationale_i$ represents one reasoning step (e.g., Step1, Step2, Step3 in Figure 1(a2)) for solving the problems. Then, we integrate qc and $rationales$ to predict the answer A.

3.2 Overview

Following the question decomposition and reasoning simplification theories [26, 33, 43] in cognitive psychology to first plan a decomposed structure of the question then perform reasoning, we establish the Dec-Eval framework with two stages: Step Decomposition (SD) stage and Rationale Generation (RG) stage as shown in Figure 2. First, the Step Decomposition stage decomposes the original question context qc ("What do these two changes...") into sub-questions $subq1, subq2, ..., subqn$ (section 3.3). Then, the Rationale Generation stage generates and evaluates rationales $rationale_1, rational - e_2, ..., rationale_n$ based on the decomposed sub-questions (section 3.4). Finally, the generated rationales are combined to inference the answer " Both are only physical changes". We also design a knowledge injection mechanism (KIM) that incorporates external knowledge in both Step Decomposition and Rationale Generation stages to facilitate the reasoning process (section 3.5). In the following sections, we will describe each part in details.

3.3 Stage 1: Step Decomposition (SD)

Recently, there are various question decomposition methods aiming to reason by breaking the question into simpler sub-ones [25, 38], and often prioritize the direct generation of sub-questions in an end-to-end manner [29]. These approaches, however, tends to overlook "logical clues" in questions, which are crucial for understanding complex multi-step questions [15]. For example, in Figure 1, when we see some key words like "have in common" (highlighted in green) in the question, we know that we should compare the commonalities between two "changes" ("Breaking a piece of glass" and "Butter melting on a hot day") and form clear reasoning steps as shown in Figure 1(a2). In other words, some "logical clues" in the question context tell us how to decompose it.

While it seems easy for humans to capture such "logical clues", enabling language models with such ability is challenging. To this end, we introduce Step Decomposition (SD) stage that captures "logical clues" to decompose a question into sub-questions iteratively. At the heart of SD is a SubGen module composed of two parts: Reasoning Type Classifier (RTC) which captures "logical clues" in a question, and Next Sub-question Generator (NSG) which generates the next sub-question $subq_{i+1}$ (e.g., "Is breaking a piece of glass...") based on previously generated contents qc^i_{prev} (e.g., "What do these two changes...") as shown in Figure 3.

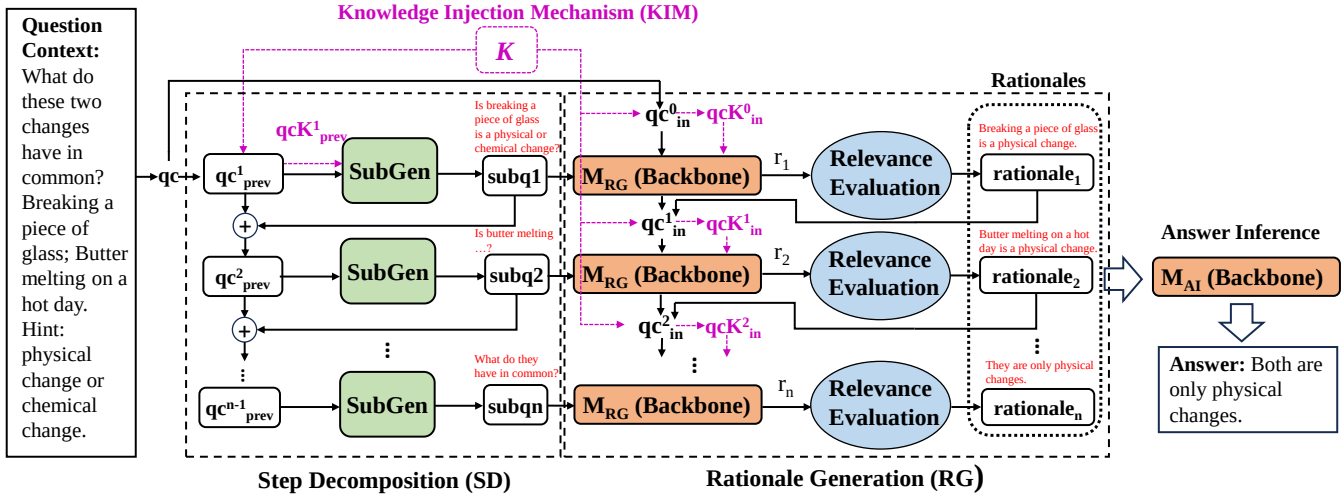

Figure 2: The overall architecture of the Dec-Eval framework. In SD stage, qc^i_{prev} is the previous contents generated by *SubGen*; $subqi$ is the sub-question generated at step i (section 3.3.2), qcK^1_{prev} is the knowledge-enhanced original question (section 3.5). In RG stage, qc^i_{in} is the combination of $subqi$ and $rationale_i$ at each step (section 3.4), where KIM can incorporate knowledge K to form qcK^i_{in} (section 3.5).

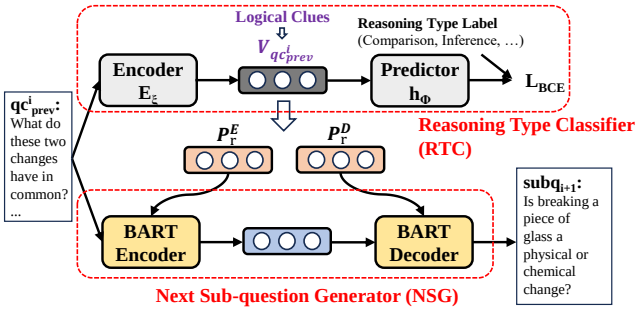

Figure 3: The architecture of SubGen. SubGen consists of two parts: Reasoning Type Classifier (RTC) and Next Sub-question Generator (NSG). $V_{qc^i_{prev}}$ is the vector representation of qc^i_{prev} at step i, and contains "logical clues" extracted from qc^i_{prev} by RTC; $V_{qc^i_{prev}}$ is then mapped into the prefix vectors P^E_r and P^D_r which are used for tuning the NSG. At step i, SubGen takes qc^i_{prev} (e.g., What do these two changes have in common?...) as input to generate the next sub-question $subq_{i+1}$ (e.g., Is breaking a piece of glass...?).

3.3.1 Reasoning Type Classifier (RTC). The first step of SD is to extract "logical clues" (embeddings containing logical information) from questions. To achieve this, we design a Reasoning Type Classifier (RTC) composed of a text encoder E_ξ and a predictor h_Φ. We use E_ξ to generate text embedding v_q for the input question q:

$$v_q = E_\xi(q). \tag{1}$$

Then we feed v_q to h_Φ to predict its one-hot reasoning type $t \in \{0,1\}^{|T|}$. Especially, RTC is pretrained on a multi-step reasoning dataset and $|T|$ denotes the predefined number of reasoning type labels (e.g., Comparison, Inference, etc.). Through minimizing the following binary cross-entropy (BCE) loss:

$$\ell_{BCE} = \frac{1}{|T|}\sum_{j=1}^{|T|}(t_j \log\left(h_\Phi(v_q)_j\right) + (1-t_j)\log\left(1-h_\Phi(v_q)_j\right)). \tag{2}$$

We use questions and their reasoning type labels from 2WikiMultiHopQA [6] to train the RTC. In the training process, we took the question(e.g., "What do these two changes in common?" in Figure 2), as the input of E_ξ, then we expected the h_Φ to correctly predict the corresponding reasoning type label(e.g., Comparison, Inference). We represented the reasoning type label with one-hot encoding, and we used all the questions in 2WikiMultiHopQA (192,606 questions) along with their reasoning type labels to train the RTC. After training, E_ξ can extract "logical clues" implicitly in v_q that are closely related to the question's reasoning type.

3.3.2 Next Sub-question Generator (NSG).. To utilize "logical clues" in question decomposition process, we first train a Next Sub-question Generator (NSG) which can predict the next sub-question based on previous contents, then we propose a Prefix-tuning [14] process in NSG to integrate the extracted "logical clues".

Training NSG. NSG is a BART-structured model [13] which iteratively predicts the next sub-question $subq_i$ based on previously generated contents $qc^i_{prev} = \{qc, subq_1, ..., subq_{i-1}\}$. Compared to methods that decompose problems using large language models [51], our approach not only conserves computational resources but also effectively leverages key features of the problem, i.e., "logical clues" in the training process.

Specifically, we collect human-annotated question decomposition examples from the 2WikiMultiHopQA [6].Each decomposition example contains the question context qc and its annotated sub-questions $subqs = \{subq_1, ..., subq_N\}$. We then use these pairs to

Algorithm 1 The process of Step Decomposition

Input: Question context qc.
Output: sub-questions.
Let $qc_{prev} = qc$; set $i = 1$
while $SubGen_\theta(qc_{prev}) \neq [EOQ]$ **do**
  $subq_i = SubGen_\theta(qc_{prev})$
  $qc_{prev} \leftarrow Concat(qc_{prev}, subq_i)$
  $i \leftarrow i + 1$
end while
Set $n = i$
return $subqs = (subq_1, subq_2, \ldots, subq_n)$

train NSG model by maximizing:

$$p(subqs|qc_{prev}) = \prod_{i=1}^{N} p_\theta(subq_i|qc^i_{prev}). \quad (3)$$

For example, if the original question Q is decomposed into two sub-questions: $Subq1$ and $Subq2$, we take Q as qc and $(Subq1, Subq2)$ as $subqs$, to construct two pairs of training data:

$$qc_{prev} = Q, subq = (Subq1) \quad (4)$$

$$qc_{prev} = (Q, Subq1), subq = (Subq2) \quad (5)$$

We extracted 50,755 such pairs of $\{qc_{prev}, subq\}$ and used all of them as the training data for NSG. The BART-Large model took q_{prev} as input to predict the $subq$.

Prefix-tuning. To integrate the "logical clues" into the NSG generation process, we design a prefix-tuning method that makes the BART model perceive the "logical clues" of the question. We use E_ξ in RTC to extract "logical clues" $v_{qc_{prev}} = E_\xi(qc_{prev})$, and map it into prefixes for tuning the BART model. Specifically, following Liu et al. [20], we use MLPs to map reasoning type features $v_{qc_{prev}}$ into prefixes $\{P^E_r, P^D_r\}$, which are added to each attention layer of BART's encoder and decoder respectively. The generation of the next sub-question is under the control of the "logical clues", and thus we can provide models with logical features in the question. We name this model as $SubGen_\theta$ and use it to generate $subqs = (subq_1, \ldots, subq_n)$ ended by a special token [EOQ]. The entire process is shown in Algorithm 1.

3.4 Stage 2: Rationale Generation (RG)

After getting the sub-questions, we employ them to generate rationales in an organized manner. Given such steps, previous approaches usually execute them sequentially [27], and often use each step's result to prompt the next step in a "Least-to-Most" manner [51]. However, sometimes it is risky to directly use the results because they may contain irrelevant information (e.g., in Figure 1(b2), the Step2 of MMCoT includes "ice melting") that could divert to the wrong answer. For LLMs, they also frequently generate hallucinated rationales [37].

To this end, in the Rationale Generation (RG) stage, we propose a novel relevance evaluation strategy to evaluate the quality of rationales at each step. Specifically, we first use a relevance evaluation strategy by training another model RSP_β that can judge the relevance between a rationale and a question. Then we combine RSP_β with a backbone LM M_{RG} to sequentially generate $rationale_i$ for each decomposed sub-question $subq_i \in subqs$.

Algorithm 2 The process of Rationale Generation

Input: The original question context qc; The decomposed $subqs$ of qc which has n sub-questions.
Parameters: A preset Relevance Threshold ε.
Output: Rationales of qc.
Let $qc_{in} = qc$; set $i = 1$
while $i \leq n - 1$ **do**
  $r_i \leftarrow M_{RG}(qc_{in})$
  if $RSP_\beta(qc, r_i) > \varepsilon$ **then**
    $rationale_i \leftarrow r_i$
  else
    $rationale_i \leftarrow \phi$(empty string)
  end if
  $qc_{in} \leftarrow$ Concat$(qc_{in}, rationale_i, subq_{i+1})$
  $i \leftarrow i + 1$
end while
Rationales $\leftarrow (rationale_1, \ldots, rationale_n)$
return Rationales

3.4.1 Relevance Evaluation Strategy. In order to evaluate the relevance between a rationale text and a question text thereby avoiding hallucination, we train a Relevance Score Predictor RSP_β that can map a question-rationale pair (q, r) to a score $\alpha \in (0, 1)$:

$$\alpha \leftarrow MLP\left[BERT(q, r)\right]. \quad (6)$$

We sampled question-rationale pairs to train the Relevance Score Predictor (RSP). Each question q_i in the training data has human-annotated ground truth rationales r_i. We considered r_i to be relevant with q_i with a relevance score $s = 1$. For each q_i, we also randomly sampled three rationales r_j from another question q_j, and we took $(q_i, r_j)_{(i \neq j)}$ as an irrelevant pair with $s = 0$. We use the MSE loss:

$$\ell_{MSE} = \frac{1}{2k} \sum_{i=0}^{2k} (\alpha_i - s_i)^2. \quad (7)$$

After that, the framework sequentially generates rationales for the n decomposed sub-questions in n steps. As shown in Figure 2, in step i, we first use a backbone model M_{RG} to generate a rationale r_i for previously generated contents qc^i_{in}, then we employ the pre-trained RSP_β to compute the relevance score α_i between r_i and qc. If α_i is greater than a preset threshold ε, we set $rationale_i \leftarrow r_i$ as one of the inputs for predicting $rationale_{i+1}$; otherwise, we won't use r_i for the next prediction step. In this way, we can alleviate the error accumulation issue caused by irrelevant rationales. The Rationale Generation process is described in Algorithm 2.

After getting the *Rationales* for the original question, we feed them to an Answer Inference backbone model M_{AI} to predict the answer. M_{RG} and M_{AI} are the same backbone models which can be either smaller models fine-tuned on training data, such as BART and T5, or LLMs used directly for inference, such as GPT-3.5.

3.5 Knowledge Injection Mechanism (KIM)

In multi-step reasoning tasks, existing methods like Self-Refine [24] often overlooks the role of knowledge in reasoning. While there are some studies extracting knowledge from various sources, the knowledge may not be fully learned and utilized in the reasoning process [4, 16–19, 30], and thus, intermediate steps generated by

these methods often have factual errors (e.g., in Figure 1(b2), "Breaking a piece of glass is a chemical change" in Step1). To address this issue, we propose a knowledge injection mechanism (KIM) that retrieves related knowledge for the two stages.

Specifically, in SD, We identify and link concepts in the question to an open-domain resource that provides background knowledge. We adopt an existing method in [45] by first extracting queries from question context then retrieving textual facts and concepts from external knowledge bases. We used the parsing method in [45] to extract noun phrases in the question text and options as our query set. Next, we searched each query in Wikipedia and computed the BERTScore [48] between the retrieved sentences and original question text. For each question, the top-5 sentences (evaluated by BERTSore) were added to our retrieved knowledge pool K. We used the same knowledge retrival method for all datasets.

The retrieved knowledge K is in the form of plain-text (e.g., "A chemical change...") and is appended to the qc_{prev} (e.g., What do these two changes ...") to form a knowledge-enhanced question context qcK_{prev} as the input of the SubGen module (line 3 of Algorithm 1). In RG, we use the same method to get external knowledge. Then in each step i of rationale generation, we concatenate K with qc^i_{in} to form the knowledge-enhanced qcK^i_{in} as the input of M_{RG} (line 3 of Algorithm 2), facilitating the generation of each $rationale_i$.

4 Experiments

4.1 Experimental Setup

4.1.1 Datasets & Baselines. We use three multi-step question answering datasets to verify the effectiveness of our Dec-Eval, i.e., **ScienceQA**, **ComplexWebQuestions** and **AppQA**. **ScienceQA** [21] is a science question answering benchmark that requires multi-step reasoning. **ComplexWebQuestions** [38] is a multi-step reasoning dataset built on WebQuestions and Freebase. **AppQA** is a multi-step QA dataset collected from a smart voice assistant of OPPO Co.,Ltd.

We take BART-Base [13], BART-Large, T5-Base [32] and T5-Large as backbones (M_{RG}), and use the following methods as baselines: **Seq2Seq**, **UnifiedQA** [9], **MMCoT** [49], **Tree of Thoughts (ToT)** [47], **Least-to-Most** [51] and **Self-Refine** [24]. **We also verify the effectiveness of Dec-Eval on LLMs in section 4.3.**

- **Seq2Seq** We fine-tune the backbone models on the training sets following [32] and [13], and directly use the fine-tuned models to infer answers in test sets.
- **UnifiedQA** Khashabi et al. [9] designs a unified pretraining framework for QA based on Seq2Seq models, which performs well on various reasoning tasks.
- **MMCoT** Zhang et al. [49] proposes a two-stage framework that separates rationale generation and answer inference.
- **Tree-of-Thoughts(ToT)** [47] allows language models to perform deliberate decision making by generating and searching multiple different reasoning path.
- **Least-to-Most** [51] is a prompting method for LLMs that breaks down a complex problem into a series of simpler sub-questions and then solve them in a sequence without evaluating the rationales.
- **Self-Refine** [24] generates an initial output using a language model, then the same model provides feedback for its output and uses it to refine itself.

4.1.2 Implementation details. In step decomposition (SD) (section 3.3), we use the pretrained BERT [2] from huggingface[1] as the text encoder E_ξ for reasoning type classifier (RTC), and train RTC for 4 epochs using AdamW with learning rate 1e-5. We train a BART-Large model for next sub-question generator (NSG) on all question decomposition samples for 5 epochs. We set the batch size to 64 and learning rate to 5e-6. In rationale generation (RG) (section 3.4), the relevance score predictor RSP_β is composed of a BERT encoder and an MLP. We train RSP_β on the extracted questions and corresponding rationales for 5 epochs with learning rate 1e-5. In prefix-tuning of NSG, we only modify the parameters of the MLP mapping v_{qc} to P_r. In testing, we set ε for RSP_β to 0.6, 0.4 and 0.8 on ScienceQA, ComplexWebQuestions and AppQA respectively. Analysis of the influence of the threshold is available in section 4.2.4. Dec-Eval and all baselines incorporate the same retrieved knowledge for fairness in comparison. In Dec-Eval's implementation, we utilize the knowledge injection mechanism (KIM) that leverages external knowledge to guide both the step decomposition stage and the rationale generation stage. For other baseline methods, external knowledge is merely used to augment the question text.

ScienceQA and ComplexWebQuestions are public datasets. AppQA is a private dataset that contains multi-step questions collected from a smart voice assistant, and thus we do not release the data. For ScienceQA, we adhere to its original training and test set splits. For ComplexWebQuestions, we utilize its validation set for testing. For AppQA, we manually constructed a test set[2].

4.2 Experimental Results

4.2.1 Overall Results. We conduct experiments to verify the effectiveness of Dec-Eval, and report the results in Table 1. We use the accuracy (ACC) as metric for ScienceQA where each question has only one correct answer, and use the F1 score and exact match score (EM) for ComplexWebQuestions and AppQA, as the questions may have several answers. We statistically test the improvement of Dec-Eval over baselines with paired t-test, and find the improvement to be significant with $p \leq 0.01 \sim 0.05$ ($**: p \leq 0.01$, $*: p \leq 0.05$). We get the following observations. First, Dec-Eval performs better than all baselines, which demonstrates our framework is more effective in enhancing backbone models' reasoning ability. Second, Dec-Eval significantly outperforms MMCoT in ScienceQA with all backbones (BART-Base by 6.0%, BART-Large by 4.3%, T5-Base by 1.4% and T5-Large by 1.1%). As MMCoT uses one single step to generate rationales, Dec-Eval's better performance indicates the significance of the step decomposition stage. Third, the CoT-based methods (i.e., MMCoT and Dec-Eval) outperform UnifiedQA and Seq2Seq in all cases, proving that generating rationales before inferring answers is better. Moreover, Dec-Eval is superior to Least-to-Most which does not evaluate intermediate rationales, confirming the rationality of our evaluation approach for reasoning steps. Last, Dec-Eval remarkably outperforms recent SOTA frameworks, i.e., ToT [47] and Self-Refine [24], which do not decompose the question, suggesting that in answering multi-step questions, planning the reasoning steps in advance through question decomposition is more effective.

[1] https://huggingface.co/transformers
[2] Our codes are available at https://github.com/ShangziXue/DecEval

Table 1: Overall results on ScienceQA, ComplexWebQuestions and AppQA. Backbone names: BART_B = BART-Base; BART_L = BART-Large; T5_B = T5-Base; T5_L = T5-Large. ($**: p \leq 0.01$, $*: p \leq 0.05$).

Datasets	ScienceQA				ComplexWebQuestions								AppQA							
Backnones	BART_B	BART_L	T5_B	T5_L	BART_B		BART_L		T5_B		T5_L		BART_B		BART_L		T5_B		T5_L	
Metrics	ACC	ACC	ACC	ACC	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM
Seq2Seq	0.565	0.615	0.672	0.700	0.852	0.765	0.863	0.778	0.866	0.791	0.870	0.805	0.598	0.449	0.608	0.451	0.614	0.460	0.628	0.463
UnifiedQA	0.623	0.652	0.701	0.734	0.861	0.770	0.866	0.779	0.870	0.801	0.871	0.809	0.603	0.456	0.610	0.457	0.617	0.463	0.629	0.466
MMCoT	0.644	0.692	0.851	0.923	0.864	0.775	0.868	0.785	0.870	0.804	0.873	0.808	0.603	0.457	0.613	0.461	0.618	0.465	0.630	0.466
ToT	0.633	0.665	0.712	0.787	0.859	0.772	0.862	0.776	0.869	0.793	0.865	0.799	0.601	0.453	0.610	0.455	0.615	0.462	0.628	0.464
Least-to-Most	0.652	0.677	0.743	0.826	0.868	0.774	0.865	0.779	0.870	0.798	0.871	0.803	0.606	0.457	0.617	0.460	0.619	0.466	0.631	0.466
Self-Refine	0.643	0.671	0.739	0.818	0.861	0.772	0.863	0.778	0.870	0.796	0.870	0.797	0.603	0.451	0.616	0.457	0.616	0.464	0.631	0.464
Dec-Eval	**0.683****	**0.722****	**0.863****	**0.934****	**0.872***	**0.789***	**0.874***	**0.807***	**0.879***	**0.812***	**0.882***	**0.818***	**0.616***	**0.466***	**0.621***	**0.474***	**0.627***	**0.472***	**0.638***	**0.473***

Table 2: Ablation studies on three pretrained models (i.e., RTC, NSG, RSP) and Knowledge Injection Mechanism (KIM). BART_B = BART-Base; BART_L = BART-Large; T5_B = T5-Base; T5_L = T5-Large.

Datasets	ScienceQA				ComplexWebQuestions								AppQA							
Backbones	BART_B	BART_L	T5_B	T5_L	BART_B		BART_L		T5_B		T5_L		BART_B		BART_L		T5_B		T5_L	
Metrics	ACC	ACC	ACC	ACC	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM
w/o RTC	0.652	0.705	0.801	0.881	0.855	0.766	0.866	0.776	0.868	0.792	0.870	0.809	0.601	0.450	0.609	0.453	0.616	0.462	0.629	0.462
w/o NSG	0.644	0.671	0.755	0.823	0.853	0.765	0.863	0.777	0.866	0.791	0.870	0.806	0.599	0.449	0.608	0.452	0.615	0.460	0.628	0.462
w/o RSP	0.663	0.693	0.821	0.865	0.859	0.767	0.864	0.777	0.870	0.796	0.871	0.808	0.602	0.453	0.610	0.456	0.617	0.460	0.629	0.465
w/o KIM	0.668	0.694	0.850	0.918	0.861	0.776	0.863	0.785	0.865	0.799	0.867	0.806	0.602	0.451	0.607	0.459	0.613	0.460	0.620	0.457
Dec-Eval	**0.683**	**0.722**	**0.863**	**0.934**	**0.872**	**0.789**	**0.874**	**0.807**	**0.879**	**0.812**	**0.882**	**0.818**	**0.616**	**0.466**	**0.621**	**0.474**	**0.627**	**0.472**	**0.638**	**0.473**

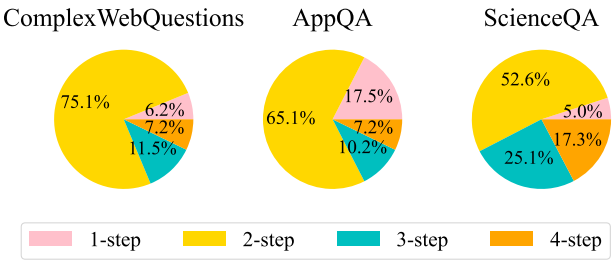

Figure 4: The proportion of number of reasoning steps.

4.2.2 Ablation Study. We conduct ablation experiments to study the effects of four key components: RTC, NSG, RSP and KIM (Table 2). Here we introduce four variants of Dec-Eval: "w/o RTC" removes the prefix-tuning process of NSG so that the "logical clues" extracted by RTC is not used in NSG; "w/o NSG" uses the untrained NSG in question decomposition; "w/o RSP" removes RSP_β and makes no evaluation on the rationales; "w/o KIM" removes KIM in both stages. We can observe that Dec-Eval performs better than all variants, which indicates that all the four components are necessary. Besides, the performances of "w/o NSG" are the lowest, which means the question decomposition part in the SD stage may be the most important for our Dec-Eval framework.

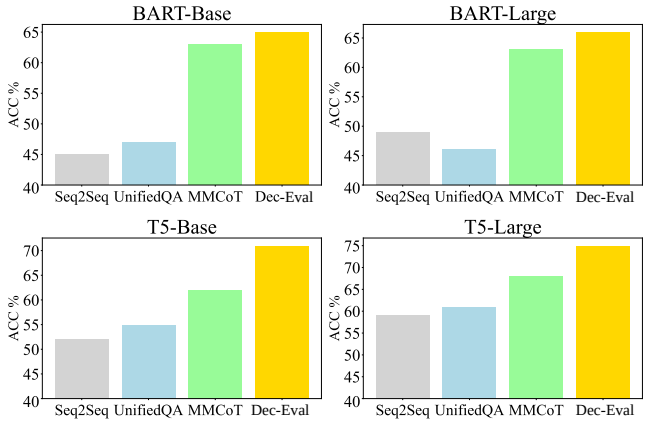

Figure 5: Human evaluation of Reasoning Step ACC.

4.2.3 Analyses of Reasoning Steps. We investigate the reasoning steps generated by Dec-Eval from the perspectives of both sub-questions and rationales. We first calculate the number of the decomposed sub-questions, and then assess the correctness of the generated rationales.

Number of Reasoning Steps. We use Dec-Eval (T5-Large) to decompose questions, and define the sub-questions as the steps. We

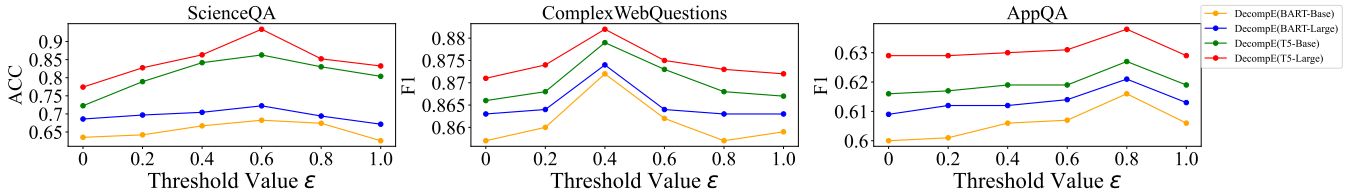

Figure 6: The performance of Dec-Eval under different RSP threshold ε.

Table 3: Experimental results about Knowledge Injection Mechanism (KIM) performance over two stages (SD and RG). Backbone names: BART_B = BART-Base; BART_L = BART-Large; T5_B = T5-Base; T5_L = T5-Large.

Datasets	ScienceQA				ComplexWebQuestions								AppQA							
Backbones	BART_B	BART_L	T5_B	T5_L	BART_B		BART_L		T5_B		T5_L		BART_B		BART_L		T5_B		T5_L	
Methods	ACC	ACC	ACC	ACC	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM
Dec-Eval-K1	0.672	0.705	0.851	0.921	0.863	0.779	0.865	0.797	0.868	0.802	0.872	0.809	0.605	0.455	0.611	0.462	0.616	0.461	0.629	0.462
Dec-Eval-K2	0.674	0.711	0.855	0.923	0.865	0.779	0.865	0.799	0.870	0.804	0.874	0.810	0.608	0.459	0.614	0.466	0.617	0.463	0.630	0.466
Dec-Eval	**0.683**	**0.722**	**0.863**	**0.934**	**0.872**	**0.789**	**0.874**	**0.807**	**0.879**	**0.812**	**0.882**	**0.818**	**0.616**	**0.466**	**0.621**	**0.474**	**0.627**	**0.472**	**0.638**	**0.473**

report the proportion of the number of reasoning steps in Figure 4. We can observe that 2-step questions account for the vast majority, and there are no questions with more than 4 steps. Besides, the proportion of 3-step and 4-step questions in ScienceQA is larger than the other two datasets, which indicates that ScienceQA dataset has more complex questions that require more reasoning steps.

Accuracy of Reasoning Steps. We also evaluate the correctness of reasoning steps for the rationales generated by Dec-Eval and baselines with different backbones. We randomly sampled 100 questions from ScienceQA [21]. Then, 5 annotators were asked to compared the model-generated rationales with the ground truth rationales to determine whether the reasoning steps are accurate. For each question, each annotator should select one from two options: (1) Accurate (no missing or incorrect reasoning steps), (2) Inaccurate (having missing or incorrect reasoning steps). After all annotators had made their selections, we employed a majority vote method to determine whether the rationales are "accurate". We count the number of questions (N) with correct generated reasoning steps, and evaluate the Reasoning Step ACC (N/100). The results are shown in Figure 5. We can get the following observations. First, the reasoning step ACC of Dec-Eval is higher than all baseline methods, proving that Dec-Eval can generate more complete reasoning steps. Besides, the performance improvements on T5(Base & Large) are greater than improvements on BART(Base & Large), showing that the Dec-Eval enhances larger model's reasoning ability better.

4.2.4 Analysis of Rationale Evaluation Threshold. Here, we analyze the influence of threshold ε of RSP in RG. We take $\varepsilon \in \{0, 0.2, 0.4, 0.6, 0.8, 1.0\}$ and show the results in Figure 6. $\varepsilon = 1$ means the generated rationale at each step is not used in next step, and $\varepsilon = 0$ means the generated rationale at each step must be used in next step. We observe on ScienceQA that the ACC of Dec-Eval shows an increasing trend when ε increases from 0 to 0.6, but declines immediately if ε continues to increase. We also observe similar trends for F1 and EM on ComplexWebQuestions and AppQA. This is probably because setting ε too high or too low may have detrimental effects on Dec-Eval to retain an appropriate part of relevant rationales for inference. Thus we set $\varepsilon = 0.6$ on ScienceQA in our experiments, which is approximately the best value for all backbone models. Similarly, we set $\varepsilon = 0.4$ and $\varepsilon = 0.8$ in ComplexWebQuestions and AppQA respectively according to observation of the changes of F1 and EM.

4.2.5 Analysis about KIM Performance on Two Stages. We conduct another study to further investigate the difference between employing KIM in the Step Decomposition (SD) stage and in the Rationale Generation (RG) stage. We design another two variants of Dec-Eval: "Dec-Eval-K1" and "Dec-Eval-K2". "Dec-Eval-K1" only incorporates K in SD stage while "Dec-Eval-K2" only incorporates K in RG stage. As shown in Table 3, Dec-Eval performs better than "Dec-Eval-K1" and "Dec-Eval-K2" in all datasets, which indicates the necessity to incorporate knowledge in both stages. Besides, in most cases, incorporating external knowledge on the RG ("Dec-Eval-K2") results in a more pronounced improvement, which demonstrates that it may be more effective to incorporate knowledge in rationale generation than in question decomposition.

4.2.6 Case Study. Finally, we conduct case studies to show the complete and interpretable rationales generated by Dec-Eval.

First, we present one typical case from ScienceQA in Table 4 to show how Dec-Eval (with BART-Large backnone) corrects the missing step in the rationales of CoT (with BART-Large backnone). As shown in Table 4, CoT(BART-Large) overlooked the step of analyzing "A piece of apple turning brown...", thereby generating an incomplete reasoning chain. However, Dec-Eval(BART-Large) can generate the missing analysis step ("A piece of apple turning brown is also a chemical change"), guided by Subq2 ("Is a piece of apple turning brown a physical change...?") in the decomposed

Table 4: Case 1 of Dec-Eval in ScienceQA. Dec-Eval supplements the missing steps in BART-Large ("A piece of apple turning brown..."), guided by the generated Subq2 ("Is a piece of apple turning brown...?").

Question Context: What do these two changes have in common? Compost rotting; A piece of apple turning brown. Options: (A)Both are physical changes. (B)Both are chemical changes.(✓)
CoT(BART-Large)'s Rationales: Compost rotting is a chemical change. As the compost rots, it breaks down and turns into a different type of matter. Both are chemical changes.
Dec-Eval (BART-Large)'s Rationales: Compost rotting is a chemical change. As the compost rots, it breaks down and turns into a different type of matter. A piece of apple turning brown is also a chemical change. Both are chemical changes.
Dec-Eval Generated Sub-questions: Subq1: Is Compost rotting a physical change or a chemical change? Subq2: Is a piece of apple turning brown a physical change or a chemical change? Subq3: What do they have in common?

Table 5: Case 2 of Dec-Eval(with GPT3.5 backbone) in ComplexWebQuestions. Dec-Eval successfully avoided generating rationales that contain factual errors (highlighted in red).

Question Context: Which tourist attractions in Sydney, Australia opened after 1950? Options: (A) Sydney Opera (✓) (B) Melbourne Skydeck ...
CoT(GPT3.5)'s Rationales: ...Sydney Opera opened in 1932 and Melbourne Skydeck is a popular tourist attraction situated in Sydney, Australia opened in 1950... **CoT(GPT3.5)'s Choice:** Melbourne Skydeck(✗)
Dec-Eval(GPT3.5)'s Rationales: .....Sydney Opera House opened in 1973, which is one of the most iconic landmarks in Sydney, the Sydney Opera House is renowned for its unique architecture and hosts various performing arts events..... **Dec-Eval(GPT3.5)'s Choice:** Sydney Opera(✓)

sub-questions. Although both methods reach the correct conclusion ("Both are chemical changes"), our Dec-Eval enhances the completeness of inference.

Then we conduct another case study to verify Dec-Eval can effectively avoid factual errors when generating rationales. In Table 5, we compare the rationales generated by Dec-Eval(T5-Large) and those generated by CoT(T5-Large), and we also present the final answers that they predicted. As we can see in Table 5, the CoT (GPT3.5)'s rationales contain certain factual inaccuracies, which are highlighted in red, for example, stating that "Sydney Opera opened in 1932" and "Melbourne Skydeck is a popular tourist attraction situated in Sydney". These errors mislead the model into selecting the wrong answer, "Melbourne Skydeck". Conversely, the rationales generated by Dec-Eval (GPT3.5)(highlighted in blue), are factually accurate ("Sydney Opera House opened in 1973..."). Using the correct rationales in the answer inference, Dec-Eval's method generates the correct answer "Sydney Opera".

4.3 Verification of Dec-Eval on LLMs

To verify that our Dec-Eval is also effective on large language models (LLMs), we take two open-source LLMs: **LLaMA-7B** [39] and **ChatGLM-6B** [3], and one API-based LLM **GPT-3.5** as backbones. Our experiments compared the performance of the traditional Chain-of-Thoughts (CoT) method [41] with our Dec-Eval approach on two benchmark datasets: ScienceQA and ComplexWebQuestions. The results, as detailed in Table 6, demonstrate that Dec-Eval significantly enhances the accuracies and F1 scores across all tested LLMs when compared to CoT. These improvements are statistically significant ($p \leq 0.05$, marked with $*$) and highlight the effectiveness of our framework.

Table 6: Comparison between CoT and Dec-Eval with three LLMs(LLaMA-7B, ChatGLM-6B, GPT-3.5). ($* : p \leq 0.05$). (CWQ = ComplexWebQuestions). Dec-Eval outperforms CoT with different LLMs as backbones.

Model	ScienceQA(ACC)	CWQ(F1)
LLaMA-7B + CoT	0.712	0.874
LLaMA-7B + Dec-Eval	**0.745***	**0.877***
ChatGLM-6B + CoT	0.668	0.868
ChatGLM-6B + Dec-Eval	**0.705***	**0.872***
GPT3.5 + CoT	0.792	0.871
GPT3.5 + Dec-Eval	**0.823***	**0.882***

These results further underscore the adaptability of Dec-Eval to different underlying models, showing that it not only enhances performance but also maintains robustness across diverse LLM architectures. The consistent improvement also suggests that Dec-Eval may be a valuable addition to the current methodologies used to augment the reasoning capabilities of LLMs.

5 Conclusion

In this paper, we proposed a two-stage Decomposition-Evaluation (Dec-Eval) framework for generating more complete and interpretable rationales in answering multi-step questions. Dec-Eval is a general framework for enhancing backnone models' reasoning ability. In the Step Decomposition (SD) stage, we proposed a novel SubGen module, which can decompose the original question into multiple sub-questions by capturing and utilizing logically "key clues". This stage can effectively allieviate the step missing issue in multi-step reasoning. Then in the Rationale Generation (RG) stage, the framework generated rationales based on the decomposed sub-questions and used a relevance evaluation strategy to enhance the quality of the rationales. Moreover, we proposed a knowledge injection mechanism (KIM) that extracted and incorporated external knowledge to guide both Step Decomposition stage and Rationale Generation stage. Extensive experiments on three challenging multi-step question answering datasets demonstrated that, with both small models and LLMs as backbones, our framework Dec-Eval can generate more logical and explainable rationales, achieving superior performances over all baselines.

Acknowledgments

This research was supported by grants from the National Key Research and Development Program of China (Grant No. 2021YFF0901000), the National Natural Science Foundation of China (62106244, 62337001), and OPPO Research Fund.

References

[1] Wenhu Chen, Xueguang Ma, Xinyi Wang, and William W Cohen. 2022. Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks. *arXiv preprint arXiv:2211.12588* (2022).

[2] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[3] 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. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 320–335.

[4] Jiale Han, Bo Cheng, and Xu Wang. 2020. Hypergraph convolutional network for multi-hop knowledge base question answering (student abstract). In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 13801–13802.

[5] Dongxiao He, Rui Guo, Xiaobao Wang, Di Jin, Yuxiao Huang, and Wenjun Wang. 2022. Inflation improves graph neural networks. In *Proceedings of the ACM Web Conference 2022*. 1466–1474.

[6] 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*. 6609–6625.

[7] Peter Jansen, Elizabeth Wainwright, Steven Marmorstein, and Clayton Morrison. 2018. WorldTree: A Corpus of Explanation Graphs for Elementary Science Questions supporting Multi-hop Inference. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

[8] Aniruddha Kembhavi, Minjoon Seo, Dustin Schwenk, Jonghyun Choi, Ali Farhadi, and Hannaneh Hajishirzi. 2017. Are you smarter than a sixth grader? textbook question answering for multimodal machine comprehension. In *Proceedings of the IEEE Conference on Computer Vision and Pattern recognition*. 4999–5007.

[9] Daniel Khashabi, Sewon Min, Tushar Khot, Ashish Sabharwal, Oyvind Tafjord, Peter Clark, and Hannaneh Hajishirzi. 2020. UNIFIEDQA: Crossing Format Boundaries with a Single QA System. In *Findings of the Association for Computational Linguistics: EMNLP 2020*. 1896–1907.

[10] Tushar Khot, Harsh Trivedi, Matthew Finlayson, Yao Fu, Kyle Richardson, Peter Clark, and Ashish Sabharwal. 2022. Decomposed Prompting: A Modular Approach for Solving Complex Tasks. In *The Eleventh International Conference on Learning Representations*.

[11] Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *Advances in neural information processing systems* 35 (2022), 22199–22213.

[12] Itay Levy, Ben Bogin, and Jonathan Berant. 2022. Diverse demonstrations improve in-context compositional generalization. *arXiv preprint arXiv:2212.06800* (2022).

[13] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. 7871–7880.

[14] Xiang Lisa Li and Percy Liang. 2021. Prefix-Tuning: Optimizing Continuous Prompts for Generation. 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)*. 4582–4597.

[15] Xin Lin, Zhenya Huang, Hongke Zhao, Enhong Chen, Qi Liu, Hao Wang, and Shijin Wang. 2021. Hms: A hierarchical solver with dependency-enhanced understanding for math word problem. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 35. 4232–4240.

[16] Xin Lin, Tianhuang Su, Zhenya Huang, Shangzi Xue, Haifeng Liu, and Enhong Chen. 2024. A Knowledge-Injected Curriculum Pretraining Framework for Question Answering. In *Proceedings of the ACM on Web Conference 2024*. 1986–1997.

[17] Jiayu Liu, Zhenya Huang, Xin Lin, Qi Liu, Jianhui Ma, and Enhong Chen. 2022. A cognitive solver with autonomously knowledge learning for reasoning mathematical answers. In *2022 IEEE International Conference on Data Mining (ICDM)*. IEEE, 269–278.

[18] Jiayu Liu, Zhenya Huang, Zhiyuan Ma, Qi Liu, Enhong Chen, Tianhuang Su, and Haifeng Liu. 2023. Guiding Mathematical Reasoning via Mastering Commonsense Formula Knowledge. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 1477–1488.

[19] Jiayu Liu, Zhenya Huang, Chengxiang Zhai, and Qi Liu. 2023. Learning by applying: A general framework for mathematical reasoning via enhancing explicit knowledge learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 37. 4497–4506.

[20] Xiao Liu, He-Yan Huang, Ge Shi, and Bo Wang. 2022. Dynamic Prefix-Tuning for Generative Template-based Event Extraction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 5216–5228.

[21] Pan Lu, Swaroop Mishra, Tanglin Xia, Liang Qiu, Kai-Wei Chang, Song-Chun Zhu, Oyvind Tafjord, Peter Clark, and Ashwin Kalyan. 2022. Learn to explain: Multimodal reasoning via thought chains for science question answering. *Advances in Neural Information Processing Systems* 35 (2022), 2507–2521.

[22] Pan Lu, Baolin Peng, Hao Cheng, Michel Galley, Kai-Wei Chang, Ying Nian Wu, Song-Chun Zhu, and Jianfeng Gao. 2023. Chameleon: Plug-and-play compositional reasoning with large language models. *arXiv preprint arXiv:2304.09842* (2023).

[23] Pan Lu, Liang Qiu, Kai-Wei Chang, Ying Nian Wu, Song-Chun Zhu, Tanmay Rajpurohit, Peter Clark, and Ashwin Kalyan. 2022. Dynamic Prompt Learning via Policy Gradient for Semi-structured Mathematical Reasoning. In *The Eleventh International Conference on Learning Representations*.

[24] Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. 2023. Self-refine: Iterative refinement with self-feedback. *arXiv preprint arXiv:2303.17651* (2023).

[25] Sewon Min, Victor Zhong, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2019. Multi-hop Reading Comprehension through Question Decomposition and Rescoring. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. 6097–6109.

[26] Allen Newell, Herbert Alexander Simon, et al. 1972. *Human problem solving*. Vol. 104. Prentice-hall Englewood Cliffs, NJ.

[27] Pruthvi Patel, Swaroop Mishra, Mihir Parmar, and Chitta Baral. 2022. Is a Question Decomposition Unit All We Need?. In *2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022*.

[28] Hongbin Pei, Yu Li, Huiqi Deng, Jingxin Hai, Pinghui Wang, Jie Ma, Jing Tao, Yuheng Xiong, and Xiaohong Guan. 2024. Multi-Track Message Passing: Tackling Oversmoothing and Oversquashing in Graph Learning via Preventing Heterophily Mixing. In *The forty-first International Conference on Machine Learning*.

[29] Ethan Perez, Patrick Lewis, Wen Tau Yih, Kyunghyun Cho, and Douwe Kiela. 2020. Unsupervised question decomposition for question answering. In *2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020*. Association for Computational Linguistics (ACL), 8864–8880.

[30] Longhu Qin, Jiayu Liu, Zhenya Huang, Kai Zhang, Qi Liu, Binbin Jin, and Enhong Chen. 2023. A Mathematical Word Problem Generator with Structure Planning and Knowledge Enhancement. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1750–1754.

[31] Lin Qiu, Yunxuan Xiao, Yanru Qu, Hao Zhou, Lei Li, Weinan Zhang, and Yong Yu. 2019. Dynamically fused graph network for multi-hop reasoning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. 6140–6150.

[32] 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. *The Journal of Machine Learning Research* 21, 1 (2020), 5485–5551.

[33] Roger C Schank and Robert P Abelson. 2013. *Scripts, plans, goals, and understanding: An inquiry into human knowledge structures*. Psychology Press.

[34] Zhengxiang Shi, Qiang Zhang, and Aldo Lipani. 2022. Stepgame: A new benchmark for robust multi-hop spatial reasoning in texts. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 36. 11321–11329.

[35] Kumar Shridhar, Alessandro Stolfo, and Mrinmaya Sachan. 2023. Distilling reasoning capabilities into smaller language models. In *Findings of the Association for Computational Linguistics: ACL 2023*. 7059–7073.

[36] Linfeng Song, Zhiguo Wang, Mo Yu, Yue Zhang, Radu Florian, and Daniel Gildea. 2018. Exploring graph-structured passage representation for multi-hop reading comprehension with graph neural networks. *arXiv preprint arXiv:1809.02040* (2018).

[37] Weiwei Sun, Zhengliang Shi, Shen Gao, Pengjie Ren, Maarten de Rijke, and Zhaochun Ren. 2023. Contrastive learning reduces hallucination in conversations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 37. 13618–13626.

[38] 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)*. 641–651.

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

[40] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2022. Self-Consistency Improves Chain of Thought Reasoning in Language Models. In *The Eleventh International Conference on Learning Representations*.

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

[42] Jason Weston, Antoine Bordes, Sumit Chopra, Alexander M Rush, Bart Van Merriënboer, Armand Joulin, and Tomas Mikolov. 2016. Towards AI-complete question answering: A set of prerequisite toy tasks. In *4th International Conference on Learning Representations, ICLR 2016*.

[43] Robert Wilensky. 1983. Planning and understanding: A computational approach to human reasoning. (1983).
[44] Tomer Wolfson, Mor Geva, Ankit Gupta, Matt Gardner, Yoav Goldberg, Daniel Deutch, and Jonathan Berant. 2020. Break it down: A question understanding benchmark. *Transactions of the Association for Computational Linguistics* 8 (2020), 183–198.
[45] Jialin Wu, Jiasen Lu, Ashish Sabharwal, and Roozbeh Mottaghi. 2022. Multi-modal answer validation for knowledge-based vqa. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 36. 2712–2721.
[46] Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 2018. HotpotQA: A Dataset for Diverse, Explainable Multi-hop Question Answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.
[47] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. Tree of thoughts: Deliberate problem solving with large language models. *arXiv preprint arXiv:2305.10601* (2023).
[48] Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. BERTScore: Evaluating Text Generation with BERT. In *International Conference on Learning Representations*.
[49] Zhuosheng Zhang, Aston Zhang, Mu Li, Hai Zhao, George Karypis, and Alex Smola. 2023. Multimodal chain-of-thought reasoning in language models. *arXiv preprint arXiv:2302.00923* (2023).
[50] Hongke Zhao, Chuang Zhao, Xi Zhang, Nanlin Liu, Hengshu Zhu, Qi Liu, and Hui Xiong. 2023. An ensemble learning approach with gradient resampling for class-imbalance problems. *INFORMS Journal on Computing* 35, 4 (2023), 747–763.
[51] Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc V Le, et al. 2022. Least-to-Most Prompting Enables Complex Reasoning in Large Language Models. In *The Eleventh International Conference on Learning Representations*.
[52] Jin Ziqi and Wei Lu. 2023. Tab-CoT: Zero-shot Tabular Chain of Thought. In *Findings of the Association for Computational Linguistics: ACL 2023*. 10259–10277.