Do Multi-Hop Question Answering Systems Know
How to Answer the Single-Hop Sub-Questions?

Yixuan Tang　　　**Hwee Tou Ng**　　　**Anthony K.H. Tung**

Department of Computer Science
National University of Singapore
{yixuan, nght, atung}@comp.nus.edu.sg

Abstract

Multi-hop question answering (QA) requires a model to retrieve and integrate information from multiple passages to answer a question. Rapid progress has been made on multi-hop QA systems with regard to standard evaluation metrics, including EM and F1. However, by simply evaluating the correctness of the answers, it is unclear to what extent these systems have learned the ability to perform multi-hop reasoning. In this paper, we propose an additional sub-question evaluation for the multi-hop QA dataset HotpotQA, in order to shed some light on explaining the reasoning process of QA systems in answering complex questions. We adopt a neural decomposition model to generate sub-questions for a multi-hop question, followed by extracting the corresponding sub-answers. Contrary to our expectation, multiple state-of-the-art multi-hop QA models fail to answer a large portion of sub-questions, although the corresponding multi-hop questions are correctly answered. Our work takes a step forward towards building a more explainable multi-hop QA system.

1 Introduction

Rapid progress has been made in the field of question answering (QA), thanks to the release of many large-scale, high-quality QA datasets. Early datasets (Hermann et al., 2015; Rajpurkar et al., 2016, 2018; Trischler et al., 2017; Joshi et al., 2017) mainly consist of single-hop questions, where an answer with supporting justification can be found within a short segment of text. These benchmarks focus on evaluating QA models' ability to perform local pattern matching between a passage and a question. Existing models (Lan et al., 2020; Zhang et al., 2020) have achieved super-human performance. Recently, multi-hop QA datasets (Khashabi et al., 2018; Welbl et al., 2018; Yang et al., 2018) have gained increasing attention. They require

models to retrieve multiple pieces of supporting evidence from different documents and to reason over the evidence collected to answer a question. The standard evaluation metrics of QA datasets include exact match (EM) and F1 scores averaged over the test set. HotpotQA (Yang et al., 2018) also provides sentence-level supporting facts required for reasoning. However, providing supporting sentences is not sufficient for us to interpret the choice of an answer for end-to-end complex QA systems. It is unclear whether the systems have performed the desired multi-hop reasoning to reach the correct answer.

In this work, we propose an additional evaluation scheme to test multi-hop QA systems' performance on answering the single-hop sub-questions of a multi-hop question. When designing a multi-hop question, we expect it to require QA models to retrieve a chain of sentences as evidence and then reasoning over them to answer the question. Evaluating QA models on sub-questions helps us to understand their behavior on each hop of the reasoning process. In addition, it evaluates whether multi-hop QA models can generalize well on simpler questions. Figure 1 presents an illustrating example. A successful complex QA model should be able to answer the two sub-questions "*Which movie stars Arnold Schwarzenegger as a former New York Police detective*" and "*What year did Guns N Roses perform a promo for End of Days*" if it understands the underlying reasoning process for the original multi-hop question.

We focus on the HotpotQA (Yang et al., 2018) dataset under the distractor setting, in which multi-hop questions are asked over several Wikipedia paragraphs. We create the evaluation dataset by generating the sub-questions and then extracting their answers automatically. The candidate sub-questions and intermediate answers are then manually verified, which results in 1,000 sub-question

Original Example	Generated Example
Gold paragraph 1: End of Days (film) End of Days is a 1999 American fantasy action horror thriller film directed by Peter Hyams and starring Arnold Schwarzenegger, Gabriel Byrne .... The film follows former New York Police Department detective Jericho Cane (Schwarzenegger) ... **Gold paragraph 2: Oh My God (Guns N' Roses song)** "Oh My God" is a song by Guns N' Roses released in 1999 on the soundtrack to the film "End of Days". The song … **One distracting paragraph: True Lies** True Lies is a 1994 American action film written, co-produced and directed by James Cameron, starring Arnold Schwarzenegger, Jamie Lee Curtis ... **Question:** What year did Guns N Roses perform a promo for a movie starring Arnold Schwarzenegger as a former New York Police detective? **Answer:** 1999 **Predicted answer:** 1999 *(correct)*	**Splitting points predicted:** What year did Guns N Roses perform a promo for | a movie | starring Arnold Schwarzenegger as a former New York Police detective? *Post-processed* Which movie starring Arnold Schwarzenegger as a former New York Police detective? — What year did Guns N Roses perform a promo for \<sub_a1\>? **Bridge entity extracted** sub_a1 = End of Days *Human verified* Which movie *stars* Arnold Schwarzenegger as a former New York Police detective? — What year did Guns N Roses perform a promo for End of Days? **Sub_a1:** End of Days **Sub-a2:** 1999 **Predicted sub_a1:** True Lies *(wrong))* **Predicted sub_a2:** 1999 *(correct)*

Figure 1: An illustrating example from the HotpotQA dataset in the distractor setting, with our construction procedure to generate an evaluation example. We only show one out of eight distracting paragraphs provided in the context due to paper length constraint.

evaluation examples. It is surprising to find that all three top-performing models which we experiment with fail to answer a large portion of sub-questions (49.8% to 60.4%), although their corresponding multi-hop questions are correctly answered.

Previous work has investigated the necessity of multi-hop reasoning on HotpotQA dataset. Jiang and Bansal (2019) construct distracting paragraphs adversarially to demonstrate that models learn to exploit reasoning shortcuts to locate the answer rather than performing multi-hop reasoning. Chen and Durrett (2019) show that a sentence-factored model can solve a large number of questions in HotpotQA, suggesting multi-hop reasoning is not really needed. Min et al. (2019a) also achieve similar result using a single-hop BERT-based model. Our sub-question evaluation is complementary to these approaches. While existing work shows the lack of multi-hop reasoning by limiting or adding text input to QA models, we provide sub-questions and intermediate answers explicitly to interpret model behavior on each hop of the reasoning process. It can be used as a complementary metric to ensure that models which can correctly answer both intermediate sub-questions and the final multi-hop question actually go through the reasoning steps

as desired. Our work takes a step forward towards building a more explainable multi-hop QA system.

2 Construction of Evaluation Examples

In this section, we introduce our semi-automatic approach to generate two sub-questions and their corresponding answers for multi-hop questions from the HotpotQA dataset. As shown in Figure 1, the evaluation examples are generated in three steps. First, we decompose each source question into several sub-strings by predicting the breaking points and post-process them to generate two sub-questions. Then, the answers for the sub-questions are extracted from the paragraphs using some heuristics. Lastly, the candidate evaluation examples generated are sent for human verification. We first introduce the HotpotQA dataset and then elaborate on each step of the construction pipeline.

2.1 HotpotQA

HotpotQA contains 113K crowd-sourced multi-hop QA pairs on Wikipedia articles. We focus on bridge-type questions that actually require multiple steps of reasoning under the distractor setting. During the construction of such an example in HotpotQA, two related paragraphs p_{gold1}, p_{gold2}

from different Wikipedia articles titled t_{gold1}, t_{gold2} are presented to crowd-workers. The two paragraphs are related since the text content in one paragraph contains the title entity of the other paragraph. This shared title entity is referred to as the bridge entity. Using Figure 1 as an example, the second paragraph about *Oh My God* contains the title entity of the first paragraph, *End of Days* (underlined). Thus, *End of Days* is referred as the bridge entity. The crowd-workers are encouraged to ask a multi-hop question using both paragraphs and to annotate the supporting sentences which help to determine the answer. Then, eight other related distracting paragraphs are retrieved from Wikipedia and mixed with the two gold paragraphs to serve as the context for the question. Given an example $E = \{C, q, a\}$ from HotpotQA, we aim to generate an evaluation example $E' = \{C, q, a, sub\_q_1, sub\_a_1, sub\_q_2, sub\_a_2\}$, where sub_q_1 and sub_q_2 are the two sub-questions, and sub_a_1 and sub_a_2 are their corresponding answers.

2.2 Sub-Question Generation

Given a multi-hop question, the first step is to decompose it into sub-questions. We adopt the model introduced in DecompRC (Min et al., 2019b) to generate the sub-questions using a copying and editing mechanism. The multi-hop question is first converted into BERT word embeddings (Devlin et al., 2019), and then sent to a fully connected neural network to predict the splitting points. It is trained on 400 annotated examples. The separated text spans are post-processed to form the two sub-questions, following a set of handcrafted rules.

2.3 Intermediate Answer Extraction

One particular characteristic of bridge-type questions from HotpotQA is that the two gold paragraphs are linked by a bridge entity. Since the crowd-workers are required to form a multi-hop question which makes use of information from both paragraphs, there is a high probability that the bridge entity is the answer to the first sub-question. For the example shown in Figure 1, *End of Days* in gold paragraph 2 is the bridge entity. It is also the intermediate answer for the multi-hop question, i.e., the answer for the first sub-question.

Three different situations are considered in order to extract the bridge entity. First, if the title entity E_A of paragraph A occurs in the other paragraph B, while the title entity E_B of B does not

Case	Gold Answer	Predicted Answer
1	from 1986 to 2013	1986 to 2013
2	City of Angles (film)	City of Angles
3	Mondelez International, Inc.	the company Mondelez International

Table 1: Examples of partially matched answer string pairs.

Model	q		q_{sub1}		q_{sub2}	
	EM	F1	EM	F1	EM	F1
DFGN	58.1	71.96	54.6	68.54	49.3	60.83
DecompRC	63.1	77.61	61	75.21	56.8	70.77
CogQA	53.2	67.82	58.6	69.65	54	68.49

Table 2: EM and F1 scores of models on 1,000 human-verified sub-question evaluation examples.

occur in A, then E_A is recognized as the bridge entity. Second, if neither E_A nor E_B is contained in the other paragraph, then the title entity with more overlapping text with the other paragraph is chosen as the bridge entity (since sometimes the alias of the Wikipedia title is used in the paragraph). Lastly, if both E_A and E_B appear in the other paragraph, then the title entity which does not appear in both the question and the answer is chosen as the bridge entity, since an entity mentioned in the multi-hop question or included in the final answer is unlikely to be the bridge entity. The bridge entity is set to be unidentified if none or both of the title entities satisfy at least one of the requirements. As illustrated in Figure 1, once the bridge entity is retrieved, the blank in the second sub-question will be updated. The answer to the second sub-question should be the same as the original multi-hop question.

2.4 Human Verification

Sub-question generation and intermediate answer extraction help to efficiently generate candidate sub-questions and their answers. To ensure the quality of the evaluation dataset, the examples generated are manually verified. For each example, we present to an annotator the original multi-hop question, the answer, two sub-questions generated and their answers, and two gold paragraphs. Questions that actually do not require multi-hop reasoning or with the wrong answer (due to wrong annotation by the HotpotQA crowd workers) are first filtered out. Then, the annotator is required to review whether sub_q_1 and sub_q_2 are two syntactically and se-

mantically correct sub-questions of q and whether sub_a_1 and sub_a_2 are valid and to correct them if not. In total, a sample of 1,000 examples generated for the HotpotQA development set are manually verified for use in our evaluation[1].

3 Experiments and Results

In order to interpret the behavior of existing models on each hop of the reasoning process required for multi-hop questions and to determine their ability to answer simple questions, we perform sub-question evaluation on three published top-performing QA models with publicly available open-source code: DFGN (Qiu et al., 2019), DecompRC (Min et al., 2019b), and CogQA (Ding et al., 2019). For all experiments, we measure EM and F1 scores for q, sub_q_1, and sub_q_2 on 1,000 human-verified examples. To measure the correctness of a predicted answer, we first use exact string match as the only metric. However, during error analysis, we find that many predicted answers that partially match the gold answers should also be regarded as correct. Some representative examples are shown in Table 1. Although these predicted answers have zero EM scores, they are semantically equivalent to the correct answers given. Therefore, we define a more flexible metric named partial match (PM) as an additional evaluation of correctness. Given a gold answer text span a_g and a predicted answer text a_p, they partially match if either one of the following two requirements is satisfied:

$$f1 > 0.8$$
$$f1 > 0.6 \land \{(a_g \ contains \ a_p) \lor (a_p \ contains \ a_g)\} \tag{1}$$

Table 2 shows the performance of the three models on multi-hop questions and their single-hop sub-questions. Compared to multi-hop questions, the performance of DFGN and DecompRC drops on simpler sub-questions, especially on the second sub-questions (11.13 F1 reduction for DFGN and 6.84 F1 reduction for DecompRC). CogQA achieves slightly better performance on sub-questions, which shows that it is also able to answer single-hop questions. The EM and F1 scores are averaged over all examples. In order to understand whether models are able to answer

---

[1]The verified dataset is available at https://github.com/yxxytang/subqa

q	q_{sub1}	q_{sub2}	DFGN	DecompRC	CogQA
c	c	c	23.0	31.3	26.7
c	c	w	9.7	7.2	5.8
c	w	c	17.9	19.1	17.8
c	w	w	7.5	5.5	2.9
w	c	c	4.9	3.0	3.6
w	c	w	17.0	18.6	22.5
w	w	c	3.5	3.4	5.9
w	w	w	16.5	11.9	14.8

Table 3: Categorical EM statistics (%) of sub-question evaluation for the three models. Under the first three columns, *c* stands for *correct* and *w* stands for *wrong*. For example, the second row shows the percentage of questions where models correctly answer both multi-hop question and the first sub-question but wrongly answer the second sub-question.

q	q_{sub1}	q_{sub2}	DFGN	DecompRC	CogQA
c	c	c	36.3	47.4	40.9
c	c	w	11.9	8.5	6.1
c	w	c	16.4	17.2	16.5
c	w	w	6.5	3.9	3.4
w	c	c	4.2	4.0	4.5
w	c	w	12.1	11.1	15.2
w	w	c	3.1	1.9	5.6
w	w	w	9.5	6.0	7.8

Table 4: Categorical PM statistics (%) of sub-question evaluation for the three models.

the sub-questions of correctly answered multi-hop questions, we collect the correctness statistics with regard to each individual example. Table 3 and Table 4 present the results. The first four rows show the percentage of examples whose multi-hop question can be correctly answered. Among these examples, we notice that there is a high probability that the models fail to answer at least one of the sub-questions, as shown in rows 2 to 4. We refer to these examples as model failure cases. The percentage of model failure cases over all correctly answered multi-hop questions is defined as model failure rate. As shown in Figure 2, all three models evaluated have a high model failure rate, indicating that the models learn to answer the complex questions without exploring the multiple steps of reasoning process as desired. The same phenomenon appears when evaluated using exact match and the less strict partial match scores.

After analyzing the model failure cases, we observe a common phenomenon that there is a high
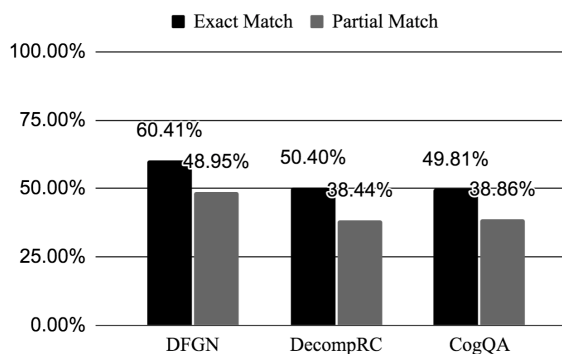
Figure 2: Model failure rates under EM and PM.

similarity between the words in the second sub-question and the words near the answer in the context. The model has learned to answer multi-hop question by local pattern matching, instead of going through the multiple reasoning steps. For the example presented in Figure 1, the model may locate the answer "*1999*" for the multi-hop question by matching the surrounding words "*Guns N Roses*" in the second sub-question. Despite answering the multi-hop question correctly, the model fails to identify the answer of the first sub-question which it is expected to retrieve as a multi-hop QA system.

4 Conclusion

We propose a new way to interpret whether multi-hop QA systems explore the multiple steps of reasoning over the evidence as desired by asking sub-questions. An automatic approach is designed to generate sub-questions for a multi-hop question. On a human-verified test set, our experiments demonstrate that top-performing multi-hop QA models fail to answer a large portion of sub-questions whose parent multi-hop questions can be correctly answered. We believe that progress on building complex QA systems that truly understand multi-hop reasoning is only possible if the evaluation metrics reward this kind of behavior. As an initial step towards a more explainable QA system, we hope our work would motivate the construction of multi-hop QA datasets with explicit reasoning paths annotated and the development of better multi-hop QA models.

References

Jifan Chen and Greg Durrett. 2019. Understanding dataset design choices for multi-hop reasoning. In *NAACL-HLT*, pages 4026–4032.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, pages 4171–4186.

Ming Ding, Chang Zhou, Qibin Chen, Hongxia Yang, and Jie Tang. 2019. Cognitive graph for multi-hop reading comprehension at scale. In *ACL*, pages 2694–2703.

Karl Moritz Hermann, Tomás Kociský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *NIPS*, pages 1693–1701.

Yichen Jiang and Mohit Bansal. 2019. Avoiding reasoning shortcuts: Adversarial evaluation, training, and model development for multi-hop QA. In *ACL*, pages 2726–2736.

Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *ACL*, pages 1601–1611.

Daniel Khashabi, Snigdha Chaturvedi, Michael Roth, Shyam Upadhyay, and Dan Roth. 2018. Looking beyond the surface: A challenge set for reading comprehension over multiple sentences. In *NAACL-HLT*, pages 252–262.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A lite BERT for self-supervised learning of language representations. In *ICLR*.

Sewon Min, Eric Wallace, Sameer Singh, Matt Gardner, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2019a. Compositional questions do not necessitate multi-hop reasoning. In *ACL*, pages 4249–4257.

Sewon Min, Victor Zhong, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2019b. Multi-hop reading comprehension through question decomposition and rescoring. In *ACL*, pages 6097–6109.

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 *ACL*, pages 6140–6150.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. In *ACL*, pages 784–789.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100, 000+ questions for machine comprehension of text. In *EMNLP*, pages 2383–2392.

Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. 2017. NewsQA: A machine comprehension dataset. In *Rep4NLP@ACL*, pages 191–200.

Johannes Welbl, Pontus Stenetorp, and Sebastian Riedel. 2018. Constructing datasets for multi-hop reading comprehension across documents. In *TACL*, pages 287–302.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *EMNLP*, pages 2369–2380.

Zhuosheng Zhang, Yuwei Wu, Junru Zhou, Sufeng Duan, Hai Zhao, and Rui Wang. 2020. SG-Net: syntax-guided machine reading comprehension. In *AAAI*, pages 9636–9643.