

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

Yixuan Tang, Hwee Tou Ng, Anthony K.H. Tung  
 School of Computing, 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 different parts of a long text to answer a question. Humans answer this kind of complex questions via a divide-and-conquer approach. In this paper, we investigate whether top-performing models for multi-hop questions understand the underlying sub-questions like humans. We adopt a neural decomposition model to generate sub-questions for a multi-hop complex question, followed by extracting the corresponding sub-answers. We show that multiple state-of-the-art multi-hop QA models fail to correctly answer a large portion of sub-questions, although their corresponding multi-hop questions are correctly answered. This indicates that these models manage to answer the multi-hop questions using some partial clues, instead of truly understanding the reasoning paths. We also propose a new model which significantly improves the performance on answering the sub-questions. 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 such as SQuAD [Rajpurkar *et al.*, 2016, 2018], NewsQA [Trischler *et al.*, 2017], and TriviaQA [Joshi *et al.*, 2017] mainly consist of single-hop questions, where an answer with supporting justification can be found within one passage or a short segment of text. These benchmarks focus on evaluating QA models' ability to perform pattern matching between a passage and a question. Recently, multi-hop QA datasets, such as QAngaroo WikiHop [Welbl *et al.*, 2018] and HotpotQA [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. However, it is unclear to what extent the multi-hop QA models truly master the ability of multi-hop reasoning.

### Context:

**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) ...

**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 was ...

**True Lies** True Lies is a 1994 American action film written, co-produced and directed by James Cameron, starring Arnold Schwarzenegger ...

**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*)

**Sub-question 1:** Which movie starring Arnold Schwarzenegger as a former New York Police detective?

**Sub-answer 1:** End of Days

**Predicted sub-answer 1:** True Lies (*wrong*)

**Sub-question 2:** What year did Guns N Roses perform a promo for End of Days?

**Sub-answer 2:** 1999

**Predicted sub-answer 2:** 1999 (*correct*)

Figure 1: An illustrating example from the HotpotQA dataset in the distractor setting, with sub-questions and their answers shown. Out of ten paragraphs provided in the context, only parts of two gold paragraphs and one related distracting paragraph are shown here due to paper length constraint.

In this work, we propose an additional evaluation scheme to test whether multi-hop QA systems know how to answer the single-hop sub-questions of a multi-hop question. Our motivation is that if a system can correctly answer a multi-hop question, it should be able to answer the corresponding single-hop sub-questions which form the complete reasoning path, just like what humans can naturally do. Figure 1 presents an illustrating example. A successful QA model needs to be able to answer the two sub-questions "Which movie starring 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 truly understands the multi-hop question "What year did Guns N Roses perform a promo for a movie starring Arnold Schwarzenegger as a former New York

*Police detective?*”.

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 set by automatically generating the sub-questions and then extracting their answers. The candidate answers to the sub-questions are then manually verified, which results in 1,000 human-verified sub-question evaluation examples. We show 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. This observation indicates that the models learn to answer the multi-hop questions without truly understanding the underlying reasoning path.

To motivate the development of new multi-hop reasoning models, we propose an initial architecture by treating the sub-questions explicitly. Our model consists of four components, namely paragraph selector, question type classifier, multi-hop QA model, and single-hop QA model. Instead of performing end-to-end training, we choose to train and evaluate each component individually. The availability of intermediate results also makes our model more interpretable. We show that with automatically generated sub-questions and their answers used for training, our model outperforms other top models by a large margin on the sub-question evaluation<sup>1</sup>. Overall, we believe that explicit reasoning should play an important role in multi-hop question answering. 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 for the HotpotQA dataset. As shown in Figure 2, the evaluation examples are generated in three steps. Firstly, 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. At last, 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 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 that makes use of information from 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. The ten paragraphs serve as the source of answers for the question. Given an example  $E$  from HotpotQA, our objective is to generate an evaluation example  $E'$  as follows:

$$\begin{aligned} E &= \{C, q, a\}, \text{ where } C = \{p_1, p_2, \dots, p_{10}\} \\ E' &= \{C, q, a, sub\_q1, sub\_a1, sub\_q2, sub\_a2\} \end{aligned} \quad (1)$$

where  $sub\_q1$  and  $sub\_q2$  are the two sub-questions, and  $sub\_a1$  and  $sub\_a2$  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 use the model introduced in Decom-PRC [Min *et al.*, 2019] to generate the sub-questions. Instead of generating a target sequence word by word, we adopt a copying and editing approach. 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 predicted 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 dataset 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 for the first sub-question. For the example shown in Figure 2, *Shirley Temple* 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 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 2, 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 multi-hop question.

<sup>1</sup>We will release the sub-question evaluation dataset and the code upon publication.

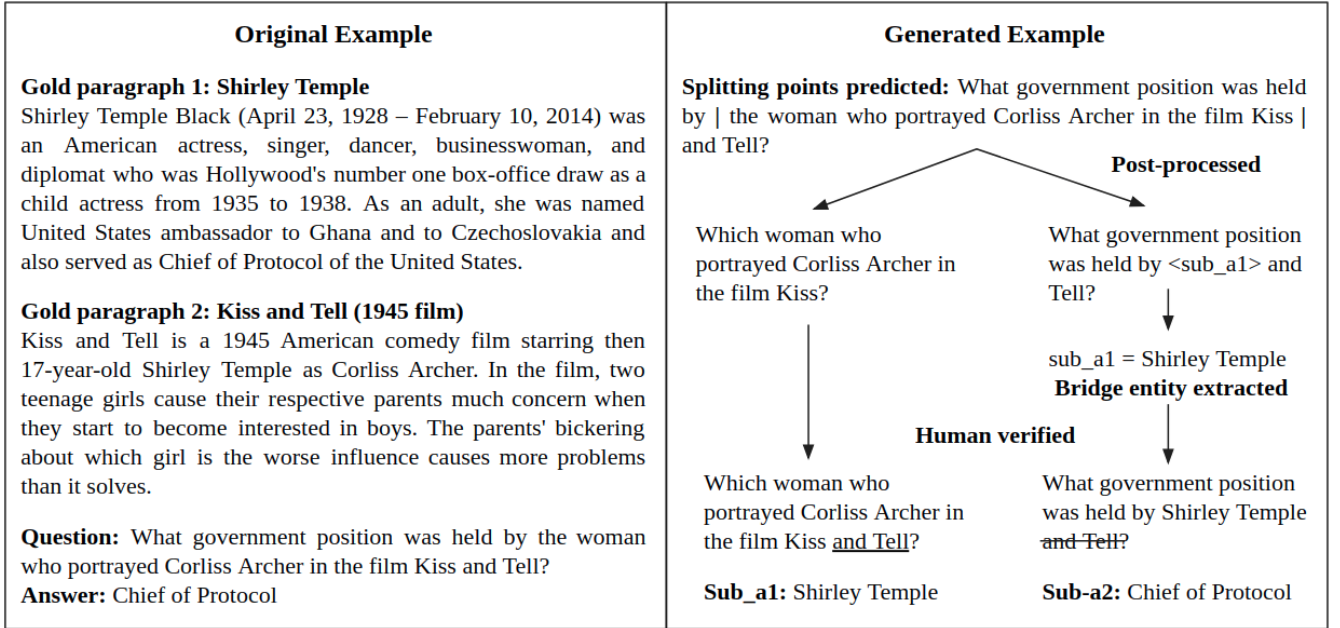


Figure 2: An illustration of our construction procedure to generate an evaluation example.

## 2.4 Human Verification

Sub-question generation and intermediate answer extraction help to efficiently generate candidate sub-questions and their answers for multi-hop questions from HotpotQA. To ensure the quality of the evaluation set, 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 corresponding answers, and two gold paragraphs, i.e.,  $\{q, a, sub\_q_1, sub\_a_1, sub\_q_2, sub\_a_2, p_{gold1}, p_{gold2}\}$ . The annotators are instructed to verify whether  $q$  is a two-hop question and whether  $a$  is the correct answer. Erroneous examples found in this step are filtered out. Then, the annotators are required to review whether  $sub\_q_1$  and  $sub\_q_2$  are two semantically and syntactically correct sub-questions of  $q$  and whether  $sub\_a_1$  and  $sub\_a_2$  are valid and to correct them if not. In total, 1,000 examples are generated from the HotpotQA official development set and manually verified for use in our evaluation.

## 3 Experiments

To determine whether existing top models understand the underlying reasoning path, we perform evaluation on three published top-performing QA models with publicly available open-source code: DFGN [Qiu *et al.*, 2019]<sup>2</sup>, DecompRC [Min *et al.*, 2019]<sup>3</sup>, and CogQA [Ding *et al.*, 2019]<sup>4</sup>.

### 3.1 Setup

For all experiments, we measure EM and F1 scores for  $q$ ,  $sub\_q_1$  and  $sub\_q_2$  on 1,000 human-verified examples. Since

<sup>2</sup><https://github.com/woshiyya/DFGN-pytorch>

<sup>3</sup><https://github.com/shmsw25/DecompRC>

<sup>4</sup><https://github.com/THUDM/CogQA>

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.

the objective of our evaluation is to determine whether models are able to correctly answer the decomposed single-hop sub-questions whose parent multi-hop questions are correctly answered. We also collect corresponding categorical statistics. 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 matched 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 are partially matched if either one of the following two requirements is satisfied.

$$f1 > 0.8$$

$$f1 > 0.6 \wedge \{(a_g \text{ contains } a_p) \vee (a_p \text{ contains } a_g)\} \quad (2)$$

### 3.2 Results

Table 2 shows the performance of DFGN, DecompRC, and CogQA on multi-hop questions and their single-hop sub-questions. Compared to multi-hop questions, the performance of DFGN and CogQA drops on simpler sub-questions,

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
Our Model	62.6	74.8	74	83.35	62.3	75.45

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

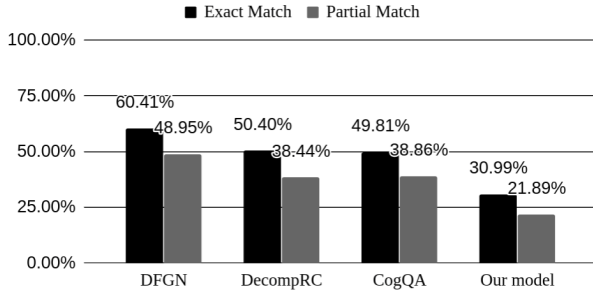


Figure 3: Model failure rates under EM and PM.

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. The EM and F1 scores are averaged over all examples. In order to understand whether models are able to answer the sub-questions of correctly answered multi-hop questions, we collect the correctness statistics with regards to each individual example. Table 3 presents the categorical statistics. The first four rows demonstrate 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. Figure 3 shows the results for all models. All three models tested have a high model failure rate, indicating that the models learn to answer the multi-hop questions without understanding the underlying reasoning chains. The same phenomenon appears under both exact match and the less strict partial match evaluation.

After analyzing the error examples, we observe one common characteristic shared by model failure cases: there is a high similarity between the words in the second sub-question and the words near the answer in a paragraph. The models are able to locate the correct answer by local pattern matching, instead of going through the reasoning steps. For the example presented in Figure 1, the information in second sub-question “What year did *Guns N Roses* perform ...” alone is enough for the model to retrieve the correct answer “1999”. With a distracting paragraph containing phrases “film ... starring Arnold Schwarzenegger ...”, the model is misled to answer the first sub-question wrongly.

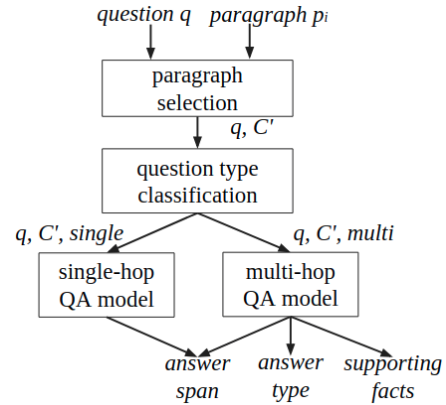


Figure 4: Model structure.

## 4 Proposed New Model

### 4.1 Training Data Creation

To prepare the training dataset, we apply the automatic procedure of sub-question generation and intermediate answer extraction to all bridge-type questions from the HotpotQA training set. The human verification step is *not* performed. In total, we produced 41,362 annotated training examples with sub-questions and their corresponding answers.

### 4.2 Model Structure

The sub-question evaluation experiments show that existing models trained on the HotpotQA dataset may answer the question correctly without understanding the underlying reasoning chain. To remedy this problem, we propose a new model to handle the sub-questions explicitly. As shown in Figure 4, our model consists of four components. Given a question, we first select related paragraphs from all ten paragraphs and concatenate them to form the context. Then a question type classifier is used to determine whether the question is a single-hop or multi-hop question. Finally, the example is sent to the corresponding QA model for answer prediction. Instead of end-to-end training, all four components are trained individually.

#### Gold Paragraph Selection

HotpotQA provides ten paragraphs in the distractor setting and only two of them contain information to answer the question. To remove unrelated context and ease the computational burden of subsequent steps, we first perform paragraph selection on the given context.

We feed a question  $q$  and each of the 10 paragraphs  $p_i$  to a BERT model [Devlin *et al.*, 2019] and get a softmaxed probability  $P(q, p_i)$  of  $p_i$  being a gold paragraph for  $q$ . A paragraph  $p_i$  is chosen as a gold paragraph for question  $q$  if  $P(q, p_i)$  is larger than a threshold  $\tau$ , or the probability is the highest or second highest among the 10 candidate paragraphs. To include most of the gold paragraphs for each question, we aim for high recall and acceptable precision. The threshold value for 98.6% recall and 68.7% precision is selected. The concatenation of all positive gold paragraphs predicted for each question is used as the new context  $C'$  for all subsequent steps.

Question Answered			Exact Match				Partial Match			
$q$	$q_{sub1}$	$q_{sub2}$	DFGN	DecompRC	CogQA	Our model	DFGN	DecompRC	CogQA	Our model
correctly	correctly	correctly	23	31.3	26.7	43.2	36.3	47.4	40.9	57.8
correctly	correctly	wrongly	9.7	7.2	5.8	5.2	11.9	8.5	6.1	5.3
correctly	wrongly	correctly	17.9	19.1	17.8	11.2	16.4	17.2	16.5	8.9
correctly	wrongly	wrongly	7.5	5.5	2.9	3	6.5	3.9	3.4	2
wrongly	correctly	correctly	4.9	3	3.6	5.4	4.2	4	4.5	5.9
wrongly	correctly	wrongly	17	18.6	22.5	20.2	12.1	11.1	15.2	13.6
wrongly	wrongly	correctly	3.5	3.4	5.9	2.5	3.1	1.9	5.6	2.1
wrongly	wrongly	wrongly	16.5	11.9	14.8	9.3	9.5	6	7.8	4.4

Table 3: Categorical statistics of sub-question evaluation based on EM and PM.

Gold \ Predicted	Multi-hop	Single-hop
Multi-hop	969	31
Single-hop	493	1507

Table 4: Results of question type classification.

Model	EM	F1
Baseline Model [Yang <i>et al.</i> , 2018]	45.60	59.02
QFE [Nishida <i>et al.</i> , 2019]	53.86	68.06
DecompRC [Min <i>et al.</i> , 2019]	55.20	69.63
KGNN [Ye <i>et al.</i> , 2019]	50.81	65.75
DFGN [Qiu <i>et al.</i> , 2019]	56.31	69.69
ChainEx [Chen <i>et al.</i> , 2019]	61.20	74.11
HGN [Fang <i>et al.</i> , 2019]	66.07	79.36
SAE-large [Tu <i>et al.</i> , 2019]	66.92	79.62
Our Model (SEval)	61.87	74.37

Table 5: Results comparison between our model with all published models on the HotpotQA leaderbaord in the distractor setting. Our model is named SEval.

### Question Type Classification

A question type classifier is constructed to explicitly predict whether a question is single-hop or multi-hop. We use a pre-trained BERT model for sequence classification as the classifier, and fine-tune it with the multi-hop questions and sub-questions generated. The model takes in a question and its new context as input and aims to predict the question type. The classification results on 1,000 evaluation examples are shown in Table 4. The accuracy is 82.5%. It is noteworthy that the recall for multi-hop questions is 96.9% and most of the error cases belong to misclassification of single-hop questions as multi-hop questions.

### Multi-Hop QA Model

We build our multi-hop QA model based on a BERT model for question answering. The question  $q$  and the new context  $C'$  are concatenated as one sequence  $q[SEP]C'$  and fed to the model, where  $[SEP]$  represents a separator token. Following the same strategy of BERT on SQuAD, we calculate the probability of each token in the context being the start position and end position of the answer span. The answer span with the highest sum of start and end probabilities is selected.

To apply on HotpotQA, we extend the prediction layer with answer type prediction and supporting facts prediction. An-

swer type prediction aims to predict whether an answer is “yes”, “no”, or an answer span. It is achieved by feeding the output vectors of BERT for the context to a linear layer, followed by a max-pooling layer over the whole sequence. For supporting facts prediction, we feed the BERT output of tokens in each sentence to a softmaxed linear layer and predict whether the sentence is a supporting fact. To realize this, we extract the start and end positions of sentences in the context and include them in the feature set during the pre-processing step. This model is fine-tuned on the official HotpotQA dataset, except that the context is replaced with the paragraphs obtained by gold paragraph selection.

### Single-Hop QA Model

We also exploit the pre-trained BERT model for our single-hop QA model component. The decomposed sub-questions and their extracted answers from the HotpotQA training set are used to fine-tune the model.

## 4.3 Results

Table 5 shows the performance of models on the HotpotQA blind test set under the distractor setting. Although our model emphasizes sub-question handling, it also achieves competitive performance on multi-hop questions. As shown in Table 2, our model outperforms two of the three QA systems on multi-hop questions in the human-verified dataset. It outperforms all other models on sub-questions. A large improvement is made on the first sub-questions. Table 3 and Figure 3 show that our model reduces the model failure rate significantly compared to the other three models. Both explicit single-hop QA modeling and the sub-question training data generated contribute to the improvement.

### 4.4 Ablation Test

Table 6 presents the ablation studies of our system. “- SingleQA” refers to the model with the question type classifier and single-hop QA model removed. This model performs slightly better on multi-hop questions, while the model failure rate is higher. The result suggests that having an explicit model to handle sub-questions is indeed helpful for intermediate answer extraction. On the other hand, our model performs much worse on multi-hop questions when the question type classifier and multi-hop QA models are removed, although it achieves better performance on the first sub-questions.

Model	$q$		$q_{sub1}$		$q_{sub2}$		Model failure rate	
	EM	F1	EM	F1	EM	F1	EM	PM
Our Model	62.6	74.8	74	83.35	62.3	75.45	30.99%	21.89%
– SingleQA	64	76.35	67	81.27	64.2	76.67	36.41%	23.61%
– MultiQA	19	24.87	78.9	85.56	61.1	74.42	28.95%	25.75%

Table 6: Performance of ablation test on handling sub-questions.

## 5 Related Work

### 5.1 Multi-hop QA Datasets

Earlier document-based QA datasets [Hermann *et al.*, 2015; Rajpurkar *et al.*, 2016; Trischler *et al.*, 2017] mainly contain single-hop questions, with emphasis on evaluating models’ ability on local pattern matching. Existing models [Lan *et al.*, 2019; Zhang *et al.*, 2019] have achieved super-human performance. To address the ability of performing complex reasoning, several multi-hop QA datasets [Khashabi *et al.*, 2018; Welbl *et al.*, 2018; Yang *et al.*, 2018] have been proposed. MultiRC [Khashabi *et al.*, 2018] contains multiple-choice questions which can only be answered by integrating information from multiple sentences. They ensure this property by excluding the questions which can be answered given incomplete context. QAngaroo WikiHop [Welbl *et al.*, 2018] constructs multi-hop questions by transforming Wikipedia facts into questions and retrieving related Wikipedia articles using a bipartite graph connecting entities and documents.

Existing QA datasets suffer from a lack of interpretability. It is a good start for HotpotQA [Yang *et al.*, 2018] to provide annotations for supporting facts. However, our work show that models jointly trained with supporting facts prediction still fail to answer the sub-questions along the reasoning path. Therefore, answering complex multi-hop reading comprehension questions in an end-to-end manner without explicit modeling of the reasoning chain has severe drawbacks, resulting in non-explainable QA systems.

### 5.2 Multi-hop QA Models on HotpotQA

Some neural models [Yang *et al.*, 2018; Nishida *et al.*, 2019; Feldman and El-Yaniv, 2019; Tu *et al.*, 2019] on HotpotQA adopt the architecture of top-performing single-hop QA models and enhance it for the multi-hop setting. They first transform a question and its context into vector representations via pre-trained word embeddings, then encode them via several layers of recurrent neural networks. Next, they update the vector representations for tokens in the context by making interaction with the question vector in a multi-hop manner. Attention mechanism [Hermann *et al.*, 2015] is commonly used to retrieve related evidence. The final representations for the context are sent to an answer prediction layer. Another group of successful models [Qiu *et al.*, 2019; Ye *et al.*, 2019; Fang *et al.*, 2019] on HotpotQA focuses on constructing graphs based on named entities extracted from a question and its context. They perform reasoning over the entity graph using graph neural networks [Scarselli *et al.*, 2009] and pass information back to the document representation for answer prediction. Although some models aim to provide explain-

able intermediate answers, their performance on sub-question evaluation is still unsatisfactory.

### 5.3 Adversarial Evaluation for QA Datasets

Jia and Liang [2017] first apply adversarial evaluation to QA models on SQuAD [Rajpurkar *et al.*, 2016]. They show that the performance of state-of-the-art models drops significantly when an additional distracting sentence is added to the context. Gan and Ng [2019] evaluate the robustness of models trained on SQuAD by asking them to answer paraphrased questions. They also find that paraphrased test questions lead to significant decrease in performance on multiple state-of-the-art models. On HotpotQA dataset, Jiang and Bansal [2019] demonstrate that existing models often answer a multi-hop question via exploiting some reasoning shortcut. To remedy the problem, they propose a new model using a control unit to guide the multi-hop reasoning process by dynamically attending to the question. All these works analyze the deficiencies of QA datasets by inserting distracting information to questions or contexts. Our work addresses this issue in a novel way. Without modifying the original data, we show the lack of reasoning ability of the existing multi-hop QA models by constructing an additional set of sub-questions for evaluation.

## 6 Conclusion

We propose a new way to evaluate whether multi-hop QA systems have learned the ability to perform reasoning over multiple documents by asking sub-questions. An automatic approach is designed to generate sub-questions for a multi-hop question. On a human-verified test set, we show that all three existing top models give worse performance on the sub-questions compared to our proposed model with an explicit question type classification component and a single-hop QA component. As an initial step towards a more explainable QA system, we hope our work could 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, Shih-ting Lin, and Greg Durrett. Multi-hop question answering via reasoning chains. *arXiv e-prints*, page arXiv:1910.02610, 2019.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, pages 4171–4186, 2019.



- Ming Ding, Chang Zhou, Qibin Chen, Hongxia Yang, and Jie Tang. Cognitive graph for multi-hop reading comprehension at scale. In *ACL*, pages 2694–2703, 2019.
- Yuwei Fang, Siqi Sun, Zhe Gan, Rohit Pillai, Shuohang Wang, and Jingjing Liu. Hierarchical graph network for multi-hop question answering. *arXiv e-prints*, page arXiv:1911.03631, 2019.
- Yair Feldman and Ran El-Yaniv. Multi-hop paragraph retrieval for open-domain question answering. In *ACL*, pages 2296–2309, 2019.
- Wee Chung Gan and Hwee Tou Ng. Improving the robustness of question answering systems to question paraphrasing. In *ACL*, pages 6065–6075, 2019.
- Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. Teaching machines to read and comprehend. In *NIPS*, pages 1693–1701, 2015.
- Robin Jia and Percy Liang. Adversarial examples for evaluating reading comprehension systems. In *EMNLP*, pages 2021–2031, 2017.
- Yichen Jiang and Mohit Bansal. Avoiding reasoning shortcuts: Adversarial evaluation, training, and model development for multi-hop QA. In *ACL*, pages 2726–2736, 2019.
- Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *ACL*, pages 1601–1611, 2017.
- Daniel Khashabi, Snigdha Chaturvedi, Michael Roth, Shyam Upadhyay, and Dan Roth. Looking beyond the surface: A challenge set for reading comprehension over multiple sentences. In *NAACL-HLT*, pages 252–262, 2018.
- Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. ALBERT: A lite BERT for self-supervised learning of language representations. *arXiv e-prints*, page arXiv:1909.11942, 2019.
- Sewon Min, Victor Zhong, Luke Zettlemoyer, and Hannaneh Hajishirzi. Multi-hop reading comprehension through question decomposition and rescoring. In *ACL*, pages 6097–6109, 2019.
- Kosuke Nishida, Kyosuke Nishida, Masaaki Nagata, Atsushi Otsuka, Itsumi Saito, Hisako Asano, and Junji Tomita. Answering while summarizing: Multi-task learning for multi-hop QA with evidence extraction. In *ACL*, pages 2335–2345, 2019.
- Lin Qiu, Yunxuan Xiao, Yanru Qu, Hao Zhou, Lei Li, Weinan Zhang, and Yong Yu. Dynamically fused graph network for multi-hop reasoning. In *ACL*, pages 6140–6150, 2019.
- Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. SQuAD: 100, 000+ questions for machine comprehension of text. In *EMNLP*, pages 2383–2392, 2016.
- Pranav Rajpurkar, Robin Jia, and Percy Liang. Know what you don’t know: Unanswerable questions for SQuAD. In *ACL*, pages 784–789, 2018.
- Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE Trans. Neural Networks*, 20(1):61–80, 2009.
- Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordani, Philip Bachman, and Kaheer Suleman. NewsQA: A machine comprehension dataset. In *Rep4NLP@ACL*, pages 191–200, 2017.
- Ming Tu, Kevin Huang, Guangtao Wang, Jing Huang, Xiaodong He, and Bowen Zhou. Select, answer and explain: Interpretable multi-hop reading comprehension over multiple documents. *arXiv e-prints*, page arXiv:1911.00484, 2019.
- Johannes Welbl, Pontus Stenetorp, and Sebastian Riedel. Constructing datasets for multi-hop reading comprehension across documents. *TACL*, 6:287–302, 2018.
- Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *EMNLP*, pages 2369–2380, 2018.
- Deming Ye, Yankai Lin, Zhenghao Liu, Zhiyuan Liu, and Maosong Sun. Multi-paragraph reasoning with knowledge-enhanced graph neural network. *arXiv e-prints*, page arXiv:1911.02170, 2019.
- Zhuosheng Zhang, Yuwei Wu, Junru Zhou, Sufeng Duan, and Hai Zhao. Sg-net: Syntax-guided machine reading comprehension. *arXiv e-prints*, page arXiv:1908.05147, 2019.