Understanding and Improving Zero-shot Multi-hop Reasoning in Generative Question Answering

Zhengbao Jiang[†], **Jun Araki**[‡], **Haibo Ding**[‡,*], **Graham Neubig**[†]
[†]Languages Technologies Institute, Carnegie Mellon University
[‡]Bosch Research
{zhengbaj,gneubig}@cs.cmu.edu
{jun.araki,haibo.ding}@us.bosch.com

Abstract

Generative question answering (QA) models generate answers to questions either solely based on the parameters of the model (the *closed-book* setting) or additionally retrieving relevant evidence (the *open-book* setting). Generative QA models can answer some relatively complex questions, but the mechanism through which they do so is still poorly understood. We perform several studies aimed at better understanding the multi-hop reasoning capabilities of generative QA models. First, we decompose multi-hop questions into multiple corresponding single-hop questions, and find marked inconsistency in QA models' answers on these pairs of ostensibly identical question chains. Second, we find that models lack zero-shot multi-hop reasoning ability: when trained only on single-hop questions, models generalize poorly to multi-hop questions. Finally, we demonstrate that it is possible to improve models' zero-shot multi-hop reasoning capacity through two methods that approximate real multi-hop natural language (NL) questions by training on either concatenation of single-hop questions or logical forms (SPARQL). In sum, these results demonstrate that multi-hop reasoning does not emerge naturally in generative QA models, but can be encouraged by advances in training or modeling techniques.[1]

1 Introduction

Empowered by large-scale pre-trained language models (LMs) (Devlin et al., 2019; Liu et al., 2019; Lewis et al., 2020a; Raffel et al., 2020), recent years have seen much progress on *generative question answering* (QA), where LMs generate answers given questions in an end-to-end fashion. While most works only demonstrate the performance of such generative QA models on simple questions (Joshi et al., 2017; Kwiatkowski et al., 2019), there has been some indication that these models can also answer complex questions that theoretically require multi-hop reasoning (Xiong et al., 2020), sometimes to an impressive degree. For example, Brown et al. (2020) demonstrate strong performance of LMs on multi-hop reasoning tasks such as DROP (Dua et al., 2019) which requires discrete reasoning and numeracy. On the other hand, many argue that LM-based QA models are not actually performing any reasoning, and rather performing (sophisticated) pattern matching and data memorization (Marcus and Davis, 2020). Simultaneously, in the context of extractive QA models that select answers from the provided context, several works have demonstrated that they can leverage superficial signals to return correct answers even when the context does not contain all the supporting facts (Chen and Durrett, 2019; Min et al., 2019a)

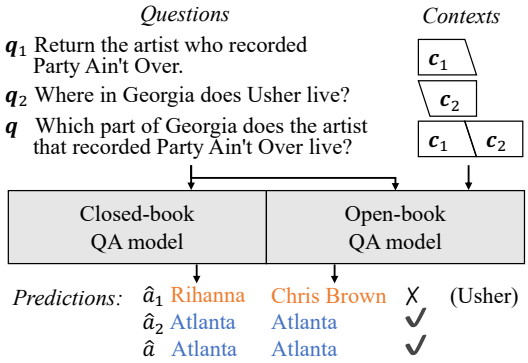

Figure 1: Probing generative closed- and open-book QA models with both multi-hop (q) and their component single-hop questions (q_1, q_2).

In this paper, we perform a closer examination of the multi-hop reasoning capabilities of generative QA models. To do so, we take multi-hop questions and their component single-hop questions to directly query generative QA models, studying their multi-hop reasoning ability. Specifically, we use multi-hop questions from the ComplexWebQuestions (Talmor and Berant, 2018) and HotpotQA (Yang et al., 2018; Tang et al., 2021) datasets as

*Haibo Ding is now at Amazon.
[1]Code is available at https://github.com/jzbjyb/multihop.

Proceedings of the 29th International Conference on Computational Linguistic, pages 1765–1775
October 12–17, 2022.

the testbed, and generate decomposed single-hop questions using heuristics (§ 2.2). We examine two types of generative QA models, namely *closed-book* (Roberts et al., 2020; Khashabi et al., 2020) and *open-book* (Guu et al., 2020; Lewis et al., 2020b; Izacard and Grave, 2021; Xiong et al., 2020) QA models that either do not or do refer to external knowledge when generating the answer respectively. Specifically, we use UnifiedQA (Khashabi et al., 2020) as a representative closed-book model, and RAG (Lewis et al., 2020b) as a representative open-book model (§ 2.1). We first ask:

RQ1 Is the correctness of decomposed single-hop questions a necessary and sufficient condition for correctness of multi-hop questions? (§ 3.2) Are answers to multi-hop and chains of decomposed questions consistent? (§ 3.3)

RQ2 Do models trained on single-hop questions demonstrate zero-shot generalization to multi-hop questions? (§ 4)

We find that generative QA models, even those close to the state-of-the-art, generally do not demonstrate robust multi-hop reasoning abilities, with success on multi-hop questions largely a result of taking shortcuts rather than true multi-hop reasoning. Zero-shot multi-hop reasoning ability does not emerge naturally from training on single-hop questions, which motivates our final question:

RQ3 Can we improve models' zero-shot multi-hop reasoning capacity by training on approximations of real multi-hop questions? (§ 4)

Motivated by the fact that pre-training on massive text endows LMs with the ability to identify semantically similar expressions, our first method uses concatenated decomposed single-hop questions to approximate real multi-hop questions. Our second method is inspired by recent work teaching LMs complex reasoning capabilities through neural execution of logical forms, e.g. by training neural models to execute SQL queries (Liu et al., 2021). We hypothesize that the ability to perform multi-hop reasoning can also be potentially learned from logical forms without reliance on NL questions. To this end, we propose to use SPARQL, a standard query language over knowledge bases, as our logical forms to endow generative QA models with the ability to perform multi-hop reasoning, and examine whether learning to execute SPARQL transfers to the ability to answer NL multi-hop questions. Both methods lead to significant improvement on zero-shot multi-hop reasoning performance, and further improvements are obtained when both are combined, opening possibilities for future work (§ 6).

2 Generative Question Answering

In this section, we briefly introduce generative QA models and multi-hop QA datasets. Then we elaborate on how we use multi-hop and decomposed questions to perform experiments.

2.1 Generative QA Models

There are two main classes of generative QA models: closed-book and open-book. Closed-book QA models usually consist of a sequence-to-sequence model that takes in a question $\boldsymbol{q}$ and calculates the probability of an answer $\boldsymbol{a}$ based on model parameters θ (Roberts et al., 2020; Khashabi et al., 2020):

$$P(\boldsymbol{a}|\boldsymbol{q};\theta) = \prod_{i=1}^{|\boldsymbol{a}|} P(a_i|\boldsymbol{q}, \boldsymbol{a}_{<i};\theta),$$

Because these models can only refer to model parameters, any relevant information must be stored in the parameters (Roberts et al., 2020). Open-book QA models first retrieve relevant context $\boldsymbol{c}$ from external resources, then generate answers using both questions and context (Guu et al., 2020; Lewis et al., 2020b; Izacard and Grave, 2021):

$$P(\boldsymbol{a}|\boldsymbol{c}, \boldsymbol{q};\theta) = \prod_{i=1}^{|\boldsymbol{a}|} P(a_i|\boldsymbol{c}, \boldsymbol{q}, \boldsymbol{a}_{<i};\theta),$$

We examine both types of models since we hypothesize that the difference in inputs might lead to different mechanisms of multi-hop reasoning.

Specifically, as our example of a closed-book model we use the UnifiedQA model of Khashabi et al. (2020). The UnifiedQA model is based on the T5 model (Raffel et al., 2020), which is an encoder-decoder model trained on the Colossal Clean Crawled Corpus (C4) by a denoising objective. It further fine-tunes on a variety of QA datasets by converting different QA formats into a unified sequence-to-sequence format.

We use the RAG model of Lewis et al. (2020b) as our example of an open-book QA model, which consists of a retriever for searching relevant passages $\boldsymbol{p}$, and a generator which generates answers $\boldsymbol{a}$ given both $\boldsymbol{p}$ and $\boldsymbol{q}$. The retriever is based on the dense passage retrieval model (DPR) (Karpukhin et al., 2020), and the generator is based on BART

Type	Questions (hop1, hop2, and multi-hop)	*Answers*
Composition	Return the country where Limonese Creole is spoken.	*Costa Rica*
	Which continent is Costa Rica located?	*North America*
	On which continent is Limonese Creole spoken?	*North America*
Conjunction	What team is Reggie Bush on 2011?	*Miami Dolphins, New Orleans Saints*
	Which one of the following is the team won the super bowl XLIV championship: Miami Dolphins, New Orleans Saints?	*New Orleans Saints*
	What team that won the super bowl XLIV championship was Reggie Bush in 2011?	*New Orleans Saints*
Superlative	What countries does the Niger River flow through?	*Benin, Guinea, Mali, Niger Nigeria*
	Which one of the following country calling code is smallest: Benin, Guinea, Mali, Niger, Nigeria?	*Mali*
	What country with the smallest calling code does the Niger River flow through?	*Mali*
Comparative	What were Hitler's parents names?	*Alois Hitler, Klara Hitler*
	Which one of the following person's date of death is after 1903-01-03: Alois Hitler, Klara Hitler?	*Klara Hitler*
	Which of Hitler's parents died after 3 January 1903?	*Klara Hitler*

Table 1: Each multi-hop question $\boldsymbol{q}$ from ComplexWebQuestions is decomposed into two single-hop questions $\boldsymbol{q}_1$ and $\boldsymbol{q}_2$. Underlined entities in the second single-hop questions are actually the answer to the first hop.

(Lewis et al., 2020a), which is also an encoder-decoder model that encodes both context and question, and generates answers autoregressively.

2.2 Multi-hop Questions and Decompositions

To understand multi-hop reasoning in generative QA models, we propose to query models using both multi-hop questions and their decompositions into multiple single-hop questions, and perform analysis based on the predictions.

To this end, we choose the **ComplexWebQuestions** dataset (Talmor and Berant, 2018) as our major testbed, as it contains multi-hop questions based on simple questions from the WebQuestionsSP dataset (Yih et al., 2016), and we can leverage simple heuristics to obtain decomposed single-hop questions and corresponding answers. Another advantage of ComplexWebQuestions is that it contains four types of questions: composition, conjunction, superlative, and comparative. This allows us to perform fine-grained analysis over these categories. Specifically, we follow heuristics in Talmor and Berant (2018) to generate decompositions. For the composition type, they use questions from WebQuestionsSP as the second hop, and replace an entity in it with a relational phrase to generate multi-hop questions. We revert this process to get the first-hop question. For the other three types, they use questions from WebQuestionsSP with multiple answers as the first hop, and add additional conditions to form the multi-hop questions. We extract those conditions and use the following template to generate the second hop question: "Which one of the following [condition]: [candidate answers]". Tab. 1 includes examples of multi-hop questions and their decompositions of four types.

We also use another small dataset from Tang et al. (2021) to test the generality of models, where a subset of multi-hop questions from **HotpotQA** (Yang et al., 2018) are manually annotated with decompositions. This dataset only contains a single type of question, which is composition. ComplexWebQuestions has 27,639/3,519 questions in the training/development set, and HotpotQA has 1,000 questions in the development set.[2]

2.3 Answer Generation and Evaluation

We use $\boldsymbol{q}_t, t \in \{1, ..., T\}$ to denote the t-th decomposed single-hop question for a multi-hop question $\boldsymbol{q}$ with T hops. Correspondingly, we use $\boldsymbol{a}_t$ to denote answers and $\boldsymbol{c}_t$ to denote retrieved context for the single-hop question $\boldsymbol{q}_t$. Since the last single-hop question always has the same answer as the corresponding multi-hop question, $\boldsymbol{a}_T = \boldsymbol{a}$. We use $\hat{\boldsymbol{a}}_t$/$\hat{\boldsymbol{a}}$ to denote the predictions from single-/multi-hop questions generated with greedy decoding:

$$\begin{matrix}\hat{\boldsymbol{a}} \\ \hat{\boldsymbol{a}}_t\end{matrix} = \arg\max_{\boldsymbol{y}} P\Big(\boldsymbol{y}\Big|\begin{matrix}[\boldsymbol{c},]\boldsymbol{q} \\ [\boldsymbol{c}_t,]\boldsymbol{q}_t\end{matrix}; \theta\Big).$$

We query models using all decomposed questions $\boldsymbol{q}_t$ and multi-hop questions $\boldsymbol{q}$ which are concatenated with the corresponding context ($\boldsymbol{c}_t$ or $\boldsymbol{c}$) for open-book settings to get predicted answers. All questions from ComplexWebQuestions and HotpotQA have two hops (i.e., $T = 2$), thus in the following sections we always use $T = 2$.

Pseudo-gold context for oracle-book models Previous work clearly demonstrates that a better retrieval component usually implies higher open-book QA performance, as it results in more retrieved contexts with answers (Chen et al., 2017; Lee et al., 2019; Karpukhin et al., 2020). Therefore, we ablate out the influence of the retrieval

[2]Since the test sets of both datasets are hidden, we use development sets for evaluation purposes. Break (Wolfson et al., 2020) is another testbed with multi-hop questions and manually decomposed questions. However, the decomposed questions are not annotated with answers, making it less appropriate for our study.

component and focus on understanding the mechanism through which generative QA models parse multi-hop questions and generate answers.

We try to provide context that contains answers to the QA model so failure of answering the question can be mainly attributed to the generator instead of the retriever. Since gold context is not annotated in the datasets, we follow Karpukhin et al. (2020) to obtain pseudo-gold context. Specifically, we use the DPR model to retrieve the top-100 passages to each single-hop question $\boldsymbol{q}_t$, and find the first one containing the answer $\boldsymbol{a}_t$, which is denoted as the pseudo-gold passage $\boldsymbol{p}_t^{\checkmark}$. Only using pseudo-gold passages as the context might make the task too easy because no incorrect contexts are presented. Therefore, we concatenate the pseudo-gold passage with a negative passage $\boldsymbol{p}_t^{\times}$ which is the first retrieved passage not containing the answers: $\boldsymbol{c}_t = [\boldsymbol{p}_t^{\checkmark}, \boldsymbol{p}_t^{\times}]$.[3] For multi-hop questions $\boldsymbol{q}$, we concatenate all context of the decomposed questions: $\boldsymbol{c} = [\boldsymbol{c}_1, ..., \boldsymbol{c}_T]$. We fix the context for all of our experiments, and only use the generator of the RAG model. For clarity, instead of open-book we use *oracle-book* to refer to these QA models in the following sections.

Multi-answer generation Since some questions involve multiple answers, as shown in Tab. 1, we fine-tune generative QA models to generate multiple answers separated by a special symbol "#".

Evaluation metrics We follow previous works (Roberts et al., 2020; Khashabi et al., 2020; Lewis et al., 2020b) to use exact match (EM) as our major evaluation metric, which measures the percentage of predictions that match the ground truth answers exactly (Rajpurkar et al., 2016; Yang et al., 2018). Since we allow multi-answer generation, we split the prediction by the special symbol "#" and match each entry against all the answers. The prediction is judged as correct if all answers are included and no extra entry is predicted.

3 Probing Multi-hop Questions and Decompositions

To answer the first research question, we probe generative QA models on both multi-hop questions and their decompositions, examining the similarities and differences in models' behavior thereon. We hypothesize that if models answer multi-hop questions in a robust way, they should be able to perform multi-hop reasoning by following the chain of decompositions internally, which makes being able to answer decomposed questions a necessary and/or sufficient condition of being able to answer multi-hop questions. Motivated by this, we choose two probing angles to examine this question. The first angle evaluates the prediction correctness on decomposed and multi-hop questions, and investigates whether there is a correlation between them. The second angle generates predictions by answering multi-hop questions and the corresponding chain of decomposed single-hop questions in a sequence, and examining whether predictions are consistent.

[3]The concatenation order is randomized to avoid leaking superficial signals to QA models.

Model	Type	Hop1	Hop2	Multi-hop
UnifiedQA	overall	32.91	49.13	33.25
	composition	47.49	38.67	33.40
	conjunction	22.49	63.30	38.01
	superlative	16.23	48.69	21.99
	comparative	15.53	25.57	8.68
RAG	overall	58.72	65.11	60.32
	composition	76.23	61.24	60.51
	conjunction	25.12	78.82	66.50
	superlative	13.33	76.67	53.33
	comparative	17.65	35.29	26.47

Table 2: EM of two models on ComplexWebQuestions overall or each type separately.

Model	Type	Hop1	Hop2	Multi-hop
UnifiedQA	composition	1.70	1.30	1.20
RAG	composition	31.55	21.66	6.15

Table 3: EM of two models on HotpotQA.

3.1 Experimental Settings

We fine-tune the UnifiedQA and RAG model using both single- and multi-hop QA pairs from the training set of the ComplexWebQuestions dataset.[4] Then we generate predictions for both single- and multi-hop questions from the test set of the ComplexWebQuestions/HotpotQA datasets, and show their overall results in Tab. 2 and Tab. 3 respectively. We measure the EM metric on first-hop $\boldsymbol{q}_1$ (**Hop1**), second-hop $\boldsymbol{q}_2$ (**Hop2**), and multi-hop questions $\boldsymbol{q}$ (**Multi-hop**) separately. We also group examples by four types to investigate whether different types of reasoning exhibit different regularities.

To examine the correlation between success on decomposed and multi-hop questions, we bucket

[4]We follow the default hyperparameters of UnifiedQA for 100K steps and a batch size of 16 on a single TPU, and the default hyperparameters of RAG for 10 epochs with a batch size of 4 on a single V100 GPU.

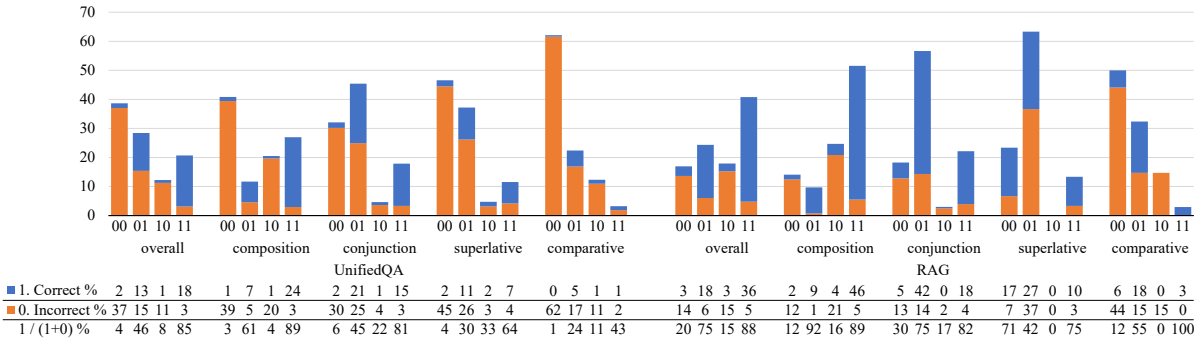

Figure 2: Correctness confusion matrices of two models on ComplexWebQuestions. Two binary codes on the X-axis indicates the correctness of the first/second single-hop question $s_1s_2 = \{00, 01, 10, 11\}$. In the table, the first/second row indicates the percentage (%) of examples of which the multi-hop question is correctly/incorrectly answered $P(s = \{1, 0\}, s_1s_2)$; the last row indicates the conditional success rate $P(s = 1|s_1s_2)$.

examples by their correctness. We use s_1, s_2 and s to denote correctness of predictions generated from the first/second single-hop and multi-hop questions, which is either 0 (incorrect) or 1 (correct). There are $8 = 2^3$ *configurations* of the correctness of a triple. We present the results using the correctness confusion matrices in Fig. 2, where examples are bucketed into 4 bins by correctness on single-hop questions (i.e., $s_1s_2 = \{00, 01, 10, 11\}$) and the inner blue/orange bars indicate the percentage of the corresponding configurations (i.e., $P(s = \{1, 0\}, s_1s_2 = \{00, 01, 10, 11\})$). To better reveal the correlation between decomposed and multi-hop questions, we compute the *conditional success rate* on multi-hop questions $P(s = 1|s_1s_2) = \frac{P(s=1,s_1s_2)}{P(s=1,s_1s_2)+P(s=0,s_1s_2)}$ in the last row of the table, which indicates how likely multi-hop questions are correctly answered given the correctness on single-hop decompositions.[5]

To examine the prediction consistency between multi-hop questions and chains of decompositions, we replace entities in the second single-hop questions $\boldsymbol{q}_2$ which correspond to answers to the first hop with a special placeholder "#1", and denoted it as $\boldsymbol{q}_2^*$. When answering a chain of decomposed questions, predictions from the first hop $\hat{\boldsymbol{a}}_1$ are used to replace the placeholder in the second hop: $\boldsymbol{q}_2^*(\hat{\boldsymbol{a}}_1)$, from which we generate the final answer denoted as $\hat{\boldsymbol{a}}_2^*$. Models fine-tuned in the normal setting only generate final answers from multi-hop questions, but not intermediate answers (i.e., answers to the hop1 question). To examine whether models can predict intermediate answers from the multi-hop question, and measure consistency on both, we append two prompts to multi-hop questions to instruct models to generate two predictions:

$$\hat{\boldsymbol{a}}_1^*\ \hat{\boldsymbol{a}} = \arg\max_{\boldsymbol{y}} P\Big(\boldsymbol{y}\Big|[\boldsymbol{c},]\boldsymbol{q},\ \text{“Intermediate answer:”}\ \text{“Final answer:”}\Big)$$

$$\hat{\boldsymbol{a}}_1\ \hat{\boldsymbol{a}}_2^* = \arg\max_{\boldsymbol{y}} P\Big(\boldsymbol{y}\Big|\begin{matrix}[\boldsymbol{c}_1,]\boldsymbol{q}_1\\ [\boldsymbol{c}_2,]\boldsymbol{q}_2^*(\hat{\boldsymbol{a}}_1)\end{matrix}\Big),$$

where $\hat{\boldsymbol{a}}_1^*$ denotes intermediate predictions. Predictions from multi-hop questions ($\hat{\boldsymbol{a}}_1^*$/$\hat{\boldsymbol{a}}$) are compared with predictions from decomposed questions in sequence ($\hat{\boldsymbol{a}}_1$/$\hat{\boldsymbol{a}}_2^*$) respectively to measure their consistency.

3.2 Correlation of Correctness

Multi-hop performance is unexpectedly high Given the hypothesis that being able to answer decomposed questions is a prerequisite of being able to answer multi-hop questions, we expect *a priori* that the performance on multi-hop questions will be much lower than the performance on all single-hop questions due to error propagation. However, what we observe on ComplexWebQuestions is the opposite: overall, the multi-hop performance is slightly higher than the hop1 performance, and the gap between hop2 and multi-hop performance is much smaller than may be expected, especially for the oracle-book RAG model. This indicates that generative QA models somehow manage to take shortcuts when answering multi-hop questions, i.e., being able to answer the multi-hop question without correctly answering its component questions.

Success on decompositions does not always imply success on multi-hop questions Looking at the overall percentage, we can see that indeed the success rate on multi-hop questions is highest if both decomposed questions are correctly answered, i.e., $P(s = 1|s_1s_2 = 11) = 85\%/88\%$ for the UnifiedQA/RAG model respectively, indicating that generative QA models are more likely to answer

[5] For robust models, $P(s = 1|s_1s_2 = 11)$ should be close to 1, $P(s = 1|s_1s_2 = \{00, 01, 10\})$ should be close to 0.

Type	Questions (hop1, hop2, multi-hop)	Answers	Predictions
Compo.	Return the country where Cerveceria Modelo Corona light beer is made.	Mexico	Mexico
	Who is Mexico's president right now 2011?	Felipe Calderón	Felipe Calderón
	Who was the president in 2011 in the country where Cerveceria Modelo Corona light beer is made?	Felipe Calderón	Juan Manuel Santos
Conj.	What year did Detroit Pistons win the championship?	2004, 1990, 1989 NBA Finals	2004, 1990, 1989 NBA Finals
	Which one of the following sports championship results were 4-1: 2004, 1990, 1989 NBA Finals?	2004, 1990 NBA Finals	2004, 1990 NBA Finals
	In what year did the Detroit Pistons win the sports championship where the results were 4-1?	2004, 1990 NBA Finals	2002 NBA Finals
Compo.	Return the team won the 2006 NFC championship	Seattle Seahawks	Indianapolis Colts
	Where do the Seattle Seahawks play?	CenturyLink Field	CenturyLink Field
	Which Stadium does the team that claimed the 2006 NFC championship play in?	CenturyLink Field	CenturyLink Field
Compa.	Who is the leader of France 2012?	Nicolas Sarkozy, François Hollande	Nicolas sarkozy
	Which one of the following started tenure after 1979: Nicolas Sarkozy, François Hollande?	Nicolas Sarkozy	Nicolas Sarkozy
	Who was the leader of France from 1979 until 2012?	Nicolas Sarkozy	Nicolas Sarkozy

Table 4: Cases of predictions generated from single/multi-hop questions of different types. Correct/Incorrect predictions are indicated in blue/orange.

multi-hop questions if they can answer all decomposed single-hop questions. However, there are still 15%/12% examples where correctness on both decomposed questions does not imply correctness on multi-hop questions, as shown by the first two examples in Tab. 4. The predictions generated from the multi-hop questions are usually of the correct type, but they diverge from the predictions generated from decomposed questions, indicating that models do not necessarily follow the decomposed components when answering multi-hop questions.

Multi-hop success is most correlated with success on the last hop Even when models fail on decomposed questions, they can still answer some percentage of multi-hop questions correctly (4-46%/15-75%) depending on which of the decomposed hops fails. Success on hop2 questions is more correlated with success on multi-hop questions than hop1 questions (i.e., $P(s = 1|s_1s_2 = 01) > P(s = 1|s_1s_2 = 10)$), especially for the oracle-book RAG model. When the model is only able to answer the second single-hop questions, there is still $46\%/75\%$ chance that the model can answer the multi-hop questions in closed/oracle-book settings respectively, indicating that generative QA models manage to take shortcuts instead of performing real reasoning. The shortcuts could be some superficial signals in the context or parameters that generative QA models can take advantage of to bypass the requirement of the first hop, as shown by the third example in Tab. 4. Or for multi-hop questions with multiple intermediate answers, generative QA models might not need to know all of them in order to answer the multi-hop questions, as shown by the fourth example in Tab. 4.

Other observations Overall, the oracle-book RAG model performs significantly better than the closed-book UnifiedQA model on both datasets, indicating that knowledge stored in parametric generative QA models is still limited and it is beneficial to provide external evidence. Hop2 performance is significantly higher than the hop1 performance on conjunction, superlative, and comparative questions, which is because hop1 questions usually have more answers than hop2 questions as shown in Tab. 1, thus being harder.[6] Both models generalize poorly to the unseen HotpotQA dataset (Tab. 3), indicating that the learned multi-hop reasoning capability cannot generalize across datasets.

	Type	EM				Consistency	
		Decompose		Multi-hop			
		Hop1	Hop2	Hop1	Hop2	Hop1	Hop2
UnifiedQA	overall	32.48	32.23	30.78	31.40	50.81	36.12
	compo.	51.87	33.97	48.51	32.13	58.92	43.24
	conj.	17.73	34.68	17.54	34.68	44.46	33.51
	super.	13.09	24.08	11.52	23.04	38.74	26.70
	compa.	13.24	9.59	12.79	10.50	47.49	11.42
RAG	overall	56.51	62.65	61.92	58.11	79.61	65.48
	compo.	73.86	60.88	76.78	54.48	86.47	67.46
	conj.	23.65	74.38	30.05	72.41	67.98	68.47
	super.	13.33	60.00	33.33	56.67	60.00	43.33
	compa.	11.76	23.53	38.24	32.35	55.88	35.29

Table 5: EM of predictions from answering chains of decomposed questions and multi-hop questions on ComplexWebQuestions and their consistency (%).

3.3 Prediction Consistency

Predictions are not consistent between multi-hop questions and chains of decompositions As shown in Tab. 5, consistency is relatively low for both models and on both hop1 and hop2, indicating that generative QA models answer multi-hop questions not necessarily in the same way as they answer decomposed questions in sequence. The consistency of the UnifiedQA model is lower than the consistency of the RAG model, which is because knowledge is limited in closed-book QA

[6] Note that the difference in the difficulty of hop1 and hop2 questions does *not* invalidate our previous conclusion about correctness correlation since we use conditional success rate.

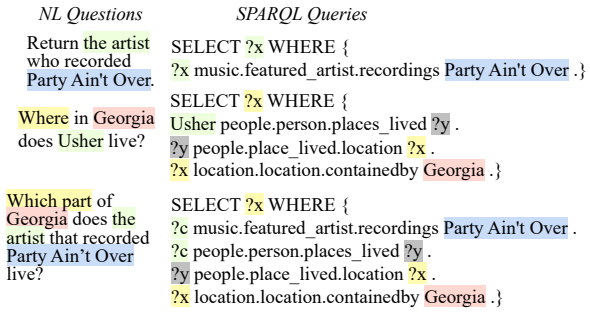

Figure 3: NL questions and corresponding SPARQL queries. Mentions of the same entity are in the same color.

models, and navigating in parameters implicitly is probably harder than searching chains of evidence in context explicitly. Consistency on the first hop is usually higher than consistency on the second hop, which is because inconsistent intermediate predictions ($\hat{a}_1$) will propagate to the second hop, leading to accumulated inconsistency.

4 Improving Zero-shot Multi-hop Reasoning Capability

In this section, we first examine LMs' zero-shot capacity for multi-hop reasoning when they are *not* trained on multi-hop NL questions. Compositional generalization ability (Lake and Baroni, 2018) is required in this case to generalize from single-hop to multi-hop questions. Unsurprisingly, generative QA models perform poorly in this setting, with almost half performance degradation. Since multi-hop NL questions are expensive to obtain, one natural question is "is it possible to improve the multi-hop reasoning ability using only single-hop NL questions, or even without any NL questions?"

We design two methods to achieve this goal. Motivated by the fact that UnifiedQA and RAG models are initialized with language models pre-trained on massive text corpora, which endows them with the ability to identify semantically similar expressions, our first method uses concatenated decomposed NL questions (i.e., $[\boldsymbol{q}_1, \boldsymbol{q}_2^*]$) to approximate the real NL multi-hop question $\boldsymbol{q}$, and fine-tunes models on them.

The second is inspired by recent progress in teaching LMs complex reasoning capabilities by executing logical forms neurally. For example, Liu et al. (2021) formulate the execution of SQL over tables as a seq2seq task where the input is a logical form string associated with a table and the output is answers (Liu et al., 2021). We hypothesize that in our multi-hop QA setting, the ability to perform multi-hop reasoning can also be potentially learned from logical forms without reliance on any NL question. To this end, we propose to use SPARQL, which is a standard query language over knowledge bases, as our logical forms. We then examine whether the ability to answer questions expressed in these SPARQL queries is transferable to NL multi-hop questions. The advantage of using SPARQL for training is that SPARQL queries can be easier and cheaper to obtain or generate than NL. For example we can use existing query logs,[7] or use manual SPARQL queries as templates and replace entities/relation to generate more queries.[8] Our observation sheds light on potential improvement on multi-hop reasoning using many SPARQL query-answer pairs.

4.1 Experimental Settings

Each NL multi-hop question in the ComplexWebQuestions dataset is associated with a SPARQL query based on the Freebase schema. We follow similar heuristics described in § 2.2 to generate SPARQL queries for the first- and second-hop NL questions. Each single- and multi-hop SPARQL query is used as a pseudo input question after replacing entity identifiers with their names, as shown in Fig. 3. In order to answer the above research questions, we design the following settings:

- No fine-tuning (**Default**): This setting uses the original model without fine-tuning.
- **S**ingle- and **M**ulti-hop **NL** (**SM-NL**): The normal setting discussed in § 3.2 where we train the model using both single- and multi-hop NL questions. This serves as the upper bound of the zero-shot performance.
- **S**ingle-hop **NL** (**S-NL**): Only use decomposed single-hop NL questions for training.
- **S**ingle- and **M**ulti-hop **SPARQL** (**SM-SPARQL**): Only use SPARQL queries.
- **S-NL** with **Concat**enation (**S-NL+Concat**): Use concatenated decomposed NL questions in addition to **S-NL**.
- **SM-NL+Concat+SM-SPARQL** (**Combo.**): Combine the previous two settings to leverage both NL and SPARQL for training.

4.2 Experimental Results

Tab. 6 includes results for all the above experimental settings. Compared to the oracle multi-hop performance, performance of only using single-hop NL questions (**S-NL**) drops by almost half on

[7] https://bit.ly/3wRRIPZ
[8] https://bit.ly/3ciOFqy

	Setting	Supervision Single	Supervision Multi	Hop1	Hop2	Multi-hop
UnifiedQA	Default			0.71	15.37	6.56
	S-NL	•		33.28	49.33	17.02
	+Concat	•	○	31.91	48.25	25.69
	SM-SPARQL	□	□	19.04	34.67	24.84
	Combo.	•□	○□	32.76	48.51	**27.14**
	SM-NL	•	•	32.91	49.13	33.25
RAG	Default			7.99	12.65	7.62
	S-NL	•		59.83	68.55	34.03
	+Concat	•	○	61.06	64.13	**53.93**
	SM-SPARQL	□	□	49.51	58.48	51.60
	Combo.	•□	○□	57.37	62.53	53.07
	SM-NL	•	•	58.72	65.11	60.32

Table 6: EM on NL questions in zero-shot multi-hop evaluation, where •, ○, □ denotes NL, concatenation, and SPARQL respectively. Oracle performance using multi-hop NL questions has a gray background. Best zero-shot multi-hop performance is in bold.

both UnifiedQA (33.25 → 17.02) and RAG models (60.32 → 34.03), indicating that without learning on multi-hop questions, compositional generalization does not naturally emerge in generative QA.

Single-hop concatenation is a good approximation of multi-hop questions Surprisingly, by simply concatenating single-hop NL questions and fine-tuning on them, multi-hop performance increases by a large margin (17.02 → 25.69/34.03 → 53.93), indicating that simple concatenation is an effective approximation for multi-hop questions. We hypothesize that LMs pre-trained on noisy text have the paraphrasing ability to generalize from concatenated simple sentences to complex sentences at least to some degree.

Models generalize from SPARQL to NL questions SPARQL queries explicitly specify compositional structure using pre-defined grammar and canonicalized entities/relations, while NL questions express this process in a more flexible way. Despite this gap, models trained solely on SPARQL queries are able to generalize to NL questions at test time on both single- and multi-hop questions, with a performance drop of 7-15 on both single- and multi-hop questions compared to (**SM-SPARQL** vs. **SM-NL**), which is far better than no fine-tuning (**Default**). This indicates that when answering NL questions, the ability learned from mapping SPARQL queries to answers can be reused by the model, similar to the observation on table-based QA (Liu et al., 2021). As demonstrated in other tasks such as table-based QA (Jiang et al., 2022) and text-to-SQL (Wu et al., 2021), converting the SPARQL queries into NL questions and training models on them can potentially mitigate the gap and further improve the performance, which we plan to explore in future works.

Combining concatenation and SPARQL improves further In this setting, we attempt to combine the merits of using concatenated single-hop NL questions, which are more natural, and SPARQL queries, which are more explicit with respect to the reasoning process. Compared to training on two types of supervision separately (**S-NL+Concat** and **SM-SPARQL**), training on both jointly (**Combo.**) improves the multi-hop performance of UnifiedQA (25.69→27.14) while slightly hurting the performance of RAG (53.93→53.07). We hypothesize that closed-book models are less constrained compared to oracle-book models due to the existence of the additional context, therefore closed-book models can benefit from the stronger supervision from a combination of two methods. Note that there is still a large gap between fine-tuning on multi-hop NL questions (**SM-NL**) and zero-shot settings, which indicates the potential for better approximations or modeling techniques.

5 Related Work

Multi-hop QA models Most multi-hop QA models proposed so far are pipeline methods that generate sub-questions to retrieve evidence iteratively (Qi et al., 2019; Ding et al., 2019; Qiu et al., 2019; Das et al., 2019; Asai et al., 2020; Min et al., 2019b; Perez et al., 2020; Xiong et al., 2020). The final answers are generated either by reading each retrieved evidence independently and recomposing the generated intermediate answers (Min et al., 2019b; Perez et al., 2020), or by taking all evidence as input at once (Qi et al., 2019; Das et al., 2019; Asai et al., 2020). Instead, we focus on understanding the multi-hop reasoning capabilities of end-to-end generative QA models in this paper.

Analysis of multi-hop reasoning Several works studying multi-hop reasoning in extractive QA models found that they can leverage superficial signals to extract answers even when the context does not contain all supporting facts (Chen and Durrett, 2019; Min et al., 2019a; Trivedi et al., 2020; Jiang and Bansal, 2019; Niu et al., 2020; Lee et al., 2021). While they examine from the perspective of dataset bias, we directly query models with both multi-hop and component single-hop questions, using both closed- and open-book generative QA models. Another work that studied multi-hop QA models using

both multi-hop and single-hop questions is Tang et al. (2021). While they use pipeline extractive QA models, we focus on end-to-end generative QA models and investigate correctness, consistency, and compositional generalization ability.

6 Conclusion

In this paper, we examined the multi-hop reasoning capabilities of generative QA models, finding that overall models take shortcuts when answering multi-hop questions, not demonstrating convincing multi-hop reasoning capability. When trained only on single-hop questions, models generalize poorly to multi-hop questions, while approximation using the concatenation of single-hop questions and SPARQL queries improves the multi-hop performance significantly. Further directions include better approximations of multi-hop questions and advanced modeling techniques that encourage compositional ability.

References

Akari Asai, Kazuma Hashimoto, Hannaneh Hajishirzi, Richard Socher, and Caiming Xiong. 2020. Learning to retrieve reasoning paths over wikipedia graph for question answering. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 1870–1879. Association for Computational Linguistics.

Jifan Chen and Greg Durrett. 2019. Understanding dataset design choices for multi-hop reasoning. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4026–4032. Association for Computational Linguistics.

Rajarshi Das, Shehzaad Dhuliawala, Manzil Zaheer, and Andrew McCallum. 2019. Multi-step retriever-reader interaction for scalable open-domain question answering. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Ming Ding, Chang Zhou, Qibin Chen, Hongxia Yang, and Jie Tang. 2019. Cognitive graph for multi-hop reading comprehension at scale. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 2694–2703. Association for Computational Linguistics.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 2368–2378. Association for Computational Linguistics.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. REALM: retrieval-augmented language model pre-training. *CoRR*, abs/2002.08909.

Gautier Izacard and Edouard Grave. 2021. Leveraging passage retrieval with generative models for open domain question answering. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, EACL 2021, Online, April 19 - 23, 2021*, pages 874–880. Association for Computational Linguistics.

Yichen Jiang and Mohit Bansal. 2019. Avoiding reasoning shortcuts: Adversarial evaluation, training, and model development for multi-hop QA. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2726–2736, Florence, Italy. Association for Computational Linguistics.

Zhengbao Jiang, Yi Mao, Pengcheng He, Graham Neubig, and Weizhu Chen. 2022. OmniTab: Pretraining with natural and synthetic data for few-shot table-based question answering. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 932–942, Seattle, United States. Association for Computational Linguistics.

Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 1601–1611. Association for Computational Linguistics.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick S. H. Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 6769–6781. Association for Computational Linguistics.

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 *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings, EMNLP 2020, Online Event, 16-20 November 2020*, pages 1896–1907. Association for Computational Linguistics.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur P. Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: a benchmark for question answering research. *Trans. Assoc. Comput. Linguistics*, 7:452–466.

Brenden M. Lake and Marco Baroni. 2018. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 2879–2888. PMLR.

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent retrieval for weakly supervised open domain question answering. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 6086–6096. Association for Computational Linguistics.

Kyungjae Lee, Seung-won Hwang, Sang-eun Han, and Dohyeon Lee. 2021. Robustifying multi-hop QA through pseudo-evidentiality training. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6110–6119, Online. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020a. 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, ACL 2020, Online, July 5-10, 2020*, pages 7871–7880. Association for Computational Linguistics.

Patrick S. H. Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020b. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Qian Liu, Bei Chen, Jiaqi Guo, Zeqi Lin, and Jian-Guang Lou. 2021. TAPEX: table pre-training via learning a neural SQL executor. *CoRR*, abs/2107.07653.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Gary Marcus and Ernest Davis. 2020. Gpt-3, bloviator: Openai's language generator has no idea what it's talking about. *Technology Review*.

Sewon Min, Eric Wallace, Sameer Singh, Matt Gardner, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2019a. Compositional questions do not necessitate multi-hop reasoning. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 4249–4257. Association for Computational Linguistics.

Sewon Min, Victor Zhong, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2019b. Multi-hop reading comprehension through question decomposition and rescoring. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 6097–6109. Association for Computational Linguistics.

Yilin Niu, Fangkai Jiao, Mantong Zhou, Ting Yao, Jingfang Xu, and Minlie Huang. 2020. A self-training

method for machine reading comprehension with soft evidence extraction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3916–3927, Online. Association for Computational Linguistics.

Ethan Perez, Patrick S. H. Lewis, Wen-tau Yih, Kyunghyun Cho, and Douwe Kiela. 2020. Unsupervised question decomposition for question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 8864–8880. Association for Computational Linguistics.

Peng Qi, Xiaowen Lin, Leo Mehr, Zijian Wang, and Christopher D. Manning. 2019. Answering complex open-domain questions through iterative query generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 2590–2602. Association for Computational Linguistics.

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 Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 6140–6150. Association for Computational Linguistics.

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. *J. Mach. Learn. Res.*, 21:140:1–140:67.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100, 000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 2383–2392. The Association for Computational Linguistics.

Adam Roberts, Colin Raffel, and Noam Shazeer. 2020. How much knowledge can you pack into the parameters of a language model? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 5418–5426. Association for Computational Linguistics.

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, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 641–651. Association for Computational Linguistics.

Yixuan Tang, Hwee Tou Ng, and Anthony K. H. Tung. 2021. Do multi-hop question answering systems know how to answer the single-hop sub-questions? In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, EACL 2021, Online, April 19 - 23, 2021*, pages 3244–3249. Association for Computational Linguistics.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2020. Is multihop QA in dire condition? measuring and reducing disconnected reasoning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 8846–8863. Association for Computational Linguistics.

Tomer Wolfson, Mor Geva, Ankit Gupta, Yoav Goldberg, Matt Gardner, Daniel Deutch, and Jonathan Berant. 2020. Break it down: A question understanding benchmark. *Trans. Assoc. Comput. Linguistics*, 8:183–198.

Kun Wu, Lijie Wang, Zhenghua Li, Ao Zhang, Xinyan Xiao, Hua Wu, Min Zhang, and Haifeng Wang. 2021. Data augmentation with hierarchical SQL-to-question generation for cross-domain text-to-SQL parsing. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8974–8983, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Wenhan Xiong, Xiang Lorraine Li, Srinivasan Iyer, Jingfei Du, Patrick S. H. Lewis, William Yang Wang, Yashar Mehdad, Wen-tau Yih, Sebastian Riedel, Douwe Kiela, and Barlas Oguz. 2020. Answering complex open-domain questions with multi-hop dense retrieval. *CoRR*, abs/2009.12756.

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 *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 2369–2380. Association for Computational Linguistics.

Wen-tau Yih, Matthew Richardson, Christopher Meek, Ming-Wei Chang, and Jina Suh. 2016. The value of semantic parse labeling for knowledge base question answering. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 2: Short Papers*. The Association for Computer Linguistics.