Summarize-then-Answer: Generating Concise Explanations for Multi-hop Reading Comprehension

Naoya Inoue[♣,♠], **Harsh Trivedi**[♣], **Steven Sinha**[♣],
Niranjan Balasubramanian[♣], **Kentaro Inui**[♢,♠]
♣ Stony Brook University, ♠ RIKEN
♢ Tohoku University
{ninoue,hjtrivedi,stsinha,niranjan}@cs.stonybrook.edu
inui@tohoku.ac.jp

Abstract

How can we generate concise explanations for multi-hop Reading Comprehension (RC)? The current strategies of identifying supporting sentences can be seen as an extractive question-focused summarization of the input text. However, these extractive explanations are not necessarily concise i.e. not minimally sufficient for answering a question. Instead, we advocate for an abstractive approach, where we propose to generate a question-focused, abstractive summary of input paragraphs and then feed it to an RC system. Given a limited amount of human-annotated abstractive explanations, we train the abstractive explainer in a semi-supervised manner, where we start from the supervised model and then train it further through trial and error maximizing a conciseness-promoted reward function. Our experiments demonstrate that the proposed abstractive explainer can generate more compact explanations than an extractive explainer with limited supervision (only 2k instances) while maintaining sufficiency.

1 Introduction

Recent approaches to multi-hop Reading Comprehension (RC) have greatly improved its *explainability*, models ability to explain their own answers (Thayaparan et al., 2020). Some adopt a pipelined architecture, where they generate an explanation first and then use it to answer the question. This "faithful-by-construction" approach is aimed at ensuring that generated explanations are closer to the systems' internal reasoning (i.e. *faithfulness*). The explanation generation step is typically formulated as a sentence selection task over the input text — selecting a set of sentences which provide support for the answer output by the model (Yang et al., 2018; Groeneveld et al., 2020, etc.).

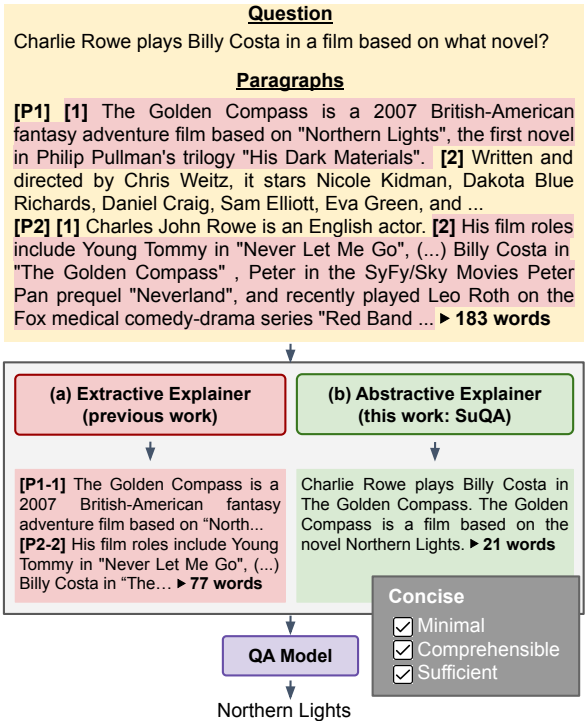

Figure 1: Summary of SUmmarizer-augmented QA (SuQA). To generate more concise (i.e. minimal, sufficient and comprehensible) explanations, SuQA augments QA module with an abstractive explainer.[1]

However, the main problem with these approaches is that the explanations obtained from the sentence selection tasks are not always minimal, sufficient, and comprehensible. The extractive explanations can include extraneous or superfluous texts which express information that is not necessary for answering questions. For example, as shown in Fig. 1 (a), the fragments such as *2007 British-American fantasy adventure* and *Young Tommy in "Never Let Me Go"* are not needed to explain the answer *Northern Lights*. Secondly, the extractive explanations may also not be sufficient: the interpretation of explanations may be dependent on its original paragraphs (e.g. pronouns). In Fig. 1 (a), *His film roles* means *Charles Rowe's*

[1]Our implementation is publicly available at https://github.com/StonyBrookNLP/suqa.

film, but this is not included in the extractive explanation. These types of gaps can also limit comprehensibility of the explanations.

In this work, we target *concise explanations* which provide minimal, sufficient and comprehensible information related to the answer. This can also be seen as targeting an abstractive question-focused summary. To this end, we propose *SUmmarizer-augmented QA (SuQA)*, an RC system augmented with an *abstractive explainer* component that generates an abstractive summary of explanations, which is then fed to a a separate QA module to produce an answer. An abstractive explainer can summarize longer sentences into short phrases and replace pronouns with their referent, leading to more compact and sufficient explanations compared to extractive explanations. For example, as shown in Fig. 1 (b), the abstractive explainer, unlike an extractive one, is allowed to remove unnecessary information such as *2007 British-American fantasy adventure*, and to generate context-independent sentences such as *Charlie Rowe plays Billy Costa in The Golden Compass*, instead of *His film roles includes....*

However, developing such an abstractive explainer imposes a significant challenge because of the limited amount of human-annotated abstractive explanations available and prohibitively high costs in extending these (Inoue et al., 2020). Given this limited supervision, how can we ensure that generated explanations are sufficient while promoting compression?

Our solution is to teach an abstractive explainer through trial and error maximizing a conciseness-promoting reward function in a reinforcement learning (RL) framework. The reward function assesses generated explanations against various criteria related to conciseness, such as linguistic acceptability, abstractiveness, and the accuracy of RC module's prediction on the generated explanations. By doing so, the model gradually learns to extract and summarize information from input texts so that they help the RC module arrive at the correct answers. Also, because the explainer aims to produce abstractive summaries, we can initialize the explainer with an abstractive summarizer that is *pretrained* on standard summarization datasets.

We evaluate the proposed approach on HotpotQA (Yang et al., 2018), one of the most popular multi-hop RC datasets. The findings of this paper can be summarized as follows:

- The semi-supervised abstractive explainer can generate more compact and sufficient explanations than extractive explanations while keeping explanations informative for answering questions. Compared to extractive ones, the abstractive explanations have a compression rate that is $\times 2.9$ higher, and improve human-judged sufficiency by 2.5 points, without incurring any significant drop in the QA accuracy.

- Even small amounts of human-annotated explanation supervision significantly improve the conciseness of generated explanations. For example, incorporating even 298 instances of annotated explanations makes the compression rate $\times 1.3$ higher and improves human-judged sufficiency by $+11.0$ points compared to the setting with no supervision for explanations.

2 Related work

Explainable NLP Three aspects of explainability have been explored (Jacovi and Goldberg, 2020): (i) comprehensibility to humans (Camburu et al., 2018; Rajani et al., 2019), (ii) faithfulness, correlation with systems' internal decision (Kumar and Talukdar, 2020; Glockner et al., 2020), (iii) conciseness, namely minimality, comprehensibility and sufficiency for solving an end task (Paranjape et al., 2020).

Earlier approaches to explainable NLP focus on comprehensibility (Camburu et al., 2018; Rajani et al., 2019), and then the community moves towards ensuring faithfulness by a system's architecture (*faithful by construction*), ranging from Natural Language Inference (Kumar and Talukdar, 2020), Fact Verification (Glockner et al., 2020) to Question Answering (Latcinnik and Berant, 2020; Groeneveld et al., 2020; Yadav et al., 2020).

Conciseness, in contrast, has been relatively unexplored. One exception is Paranjape et al. (2020), who propose to learn to extract a minimal set of input sentences that are useful for solving downstream tasks by imposing information bottleneck on the NLP framework. Although our work shares the similar spirit with their work, unlike our work, their explainer is extractive. Our work is the first to incorporate abstractive explainers into RC systems.

To date, more NLP datasets are being annotated with explanations (Wiegreffe and Marasović, 2021), but most of them are based on extractive

explanations (Yang et al., 2018; DeYoung et al., 2020, etc.). For abstractive explanations, there are a few resources: textual entailment dataset (Camburu et al., 2018), and question answering dataset in non-RC settings (i.e. input paragraphs are not given) (Jansen et al., 2018; Rajani et al., 2019). As for RC, Inoue et al. (2020) annotate HotpotQA (Yang et al., 2018) with abstractive explanations, but only 2k of them (i.e. 3% of the whole dataset) are annotated.

Abstractive explainer A similar pipeline model has been proposed for textual entailment (Camburu et al., 2018) and commonsense QA (Rajani et al., 2019), where the model first generates an explanation, and then the downstream classifier consumes it to predict a task label. Although the architecture is the same as ours, the training process is different: they train the explainer in a fully supervised manner using input-explanation pairs, while our work additionally leverages a signal from the downstream QA model in RL. As demonstrated in §5.5, we show that this additional training is crucial when few annotated explanations are available.

Generating abstractive explanations is closely related to query-focused summarization (QFS), where a few datasets are publicly available (Dang, 2006; Baumel et al., 2016; Nema et al., 2017; Pasunuru et al., 2021). However, the task setting of QFS is radically different from our problem setting, which makes it difficult to leverage the datasets and models in a straightforward manner. The QFS task typically consists of non-question queries (e.g. keywords or complex sentences) or opinion-oriented questions (e.g. *Is X a good idea?*), and gold summaries are not guaranteed to contain all information required for answering questions. We leave it the future work to explore how to effectively use their datasets and models in our task.

3 SuQA: SUmmarizer-augmented QA

Extractive explanations may contain superfluous information that is not necessary for answering questions or may not be sufficient for answering questions. We address this issue by generating concise explanations defined as follows.

Definition 1. *An explanation is* concise *if it is (i) minimal, (ii) comprehensible, and (iii) sufficient for answering the question.*

Fig. 1 summarizes the overall architecture. To ensure the faithfulness of explanations, we use a pipeline architecture consisting of two main components: (i) an *abstractive explainer* (AX) and (ii) *QA module* (QAM) (§3.1). The AX takes a question and paragraph as inputs and is responsible for generating a question-focused, abstractive summary of input paragraphs. The QAM then answers the question solely based on the generated summary. This summary is supposed to contain information necessary for answering questions and is the only factor that the QAM relies on. Thus, the generated summary can be interpreted as a faithful explanation of the model.

3.1 Architecture

First, we formalize the overall pipeline. Given a question q and paragraphs p, we first generate the most-likely explanation e as follows:

$$e = \arg\max_{e'} p_\pi(e'|q, p), \tag{1}$$

where p_π is the AX. We then answer the question q *solely* based on the generated explanation e:

$$a = \arg\max_{a'} p_\phi(a'|q, e), \tag{2}$$

where p_ϕ is the QAM. Our architecture is agnostic to the implementation of AX and QAM as long as they are differentiable.

From the viewpoint of probabilistic models, this formulation is a special case of a probabilistic latent variable model of $p(a|q, p)$ where explanations are treated as latent variables, similar to retrieval-augmented language models (Guu et al., 2020; Lewis et al., 2020b). Specifically, we have $p(a|q, p) = \sum_e p_\phi(a|q, e)p_\pi(e|q, p)$, assuming $p_\phi(a|q, e, p) = p_\phi(a|q, e)$. Replacing the sum with $\arg\max$ yields Equation 2. The main challenge is that $p_\pi(e|q, p)$ is not a retriever but a text generator.

Abstractive explainer (AX) It takes a paragraph p and a question q as an input, and outputs an explanation e. We implement the AX using a sequence-to-sequence generation model as follows:

$$p_\pi(e|q, p) = \prod_t^n p_\pi(e_t|e_{<t}, q, p) \tag{3}$$

In our experiments, we use BART (Lewis et al., 2020a). We simply concatenate q and p into one text with a separator token to generate a question-focused summary of the paragraph.

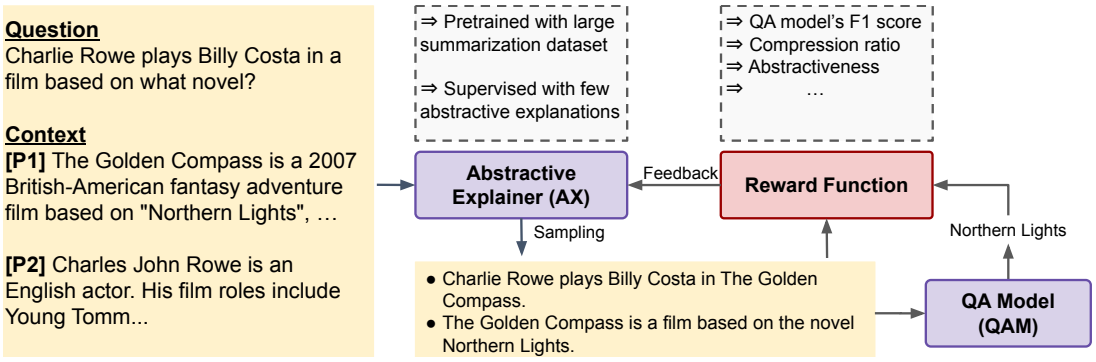

Figure 2: Training regime of the proposed method. We pretrain the AX with a large summarization dataset and finetune it on a limited amount of human-annotated explanations (§4.1). We then train it further through indirect supervision from the QAM using Reinforcement Learning (§4.2).

QA module (QAM) It takes a question q an explanation e generated by the AX as an input, and outputs an answer a. We implement the QAM as a generation-based question answering module.

$$p_\phi(a|q,e) = \prod_t^n p_\phi(a_t|a_{<t},q,e) \tag{4}$$

4 Training

Fig. 2 shows an overview of our training regime. The main challenge of training the AX is that human-annotated explanations are rarely available for question-answer pairs, though the conciseness of explanations heavily relies on human judgement. To address this issue, we train the AX in a semi-supervised manner.

4.1 Supervised training with summarization and explanation generation

Because the AX aims to produce abstractive summaries, we initialize the AX with an abstractive summarizer that is pretrained on standard summarization datasets. As we will see later (§5.6.2), this initialization is one of the key ingredients for the AX.

Given a training dataset consisting of QA pairs annotated with its gold explanations, we train the AX with a standard teacher forcing approach. Specifically, we minimize the following loss:

$$L_{\mathrm{ML}} = \sum_{t=1}^{n} \log p_\pi(y_t^*|y_{<t}^*,q), \tag{5}$$

where q is a question, and $(y_1^*, y_2^*, ..., y_n^*)$ is a human-annotated explanation for the QA pair.

4.2 Semi-supervised training

Although the fully supervised training provides the AX with direct signals, large-scale annotation of such abstractive explanation is prohibitively costly (Inoue et al., 2020). Thus, after training the AX in a supervised fashion, we further train the AX through indirect supervision from answers, which are much cheaper to annotate.

We use the RL framework and design a reward function that assesses the goodness of generated explanations based on answers and sentence-level supporting facts. A state here is a sequence of explanation tokens generated so far $y_{<t}$, an action is to generate a token, and the policy function is a probability distribution $p_\pi(y_t|y_{<t},q)$ of tokens given by the AX, as with previous work on RL-based language generation (Rennie et al., 2017, etc.). Given a reward function $r(\cdot)$ which we describe later, we optimize the policy function $p_\pi(y_t|y_{<t},q)$ via self-critical training (Rennie et al., 2017) as follows:

$$L_{\mathrm{RL}} = -\frac{1}{n}\sum_{t=1}^{n}(r(y') - r(\hat{y}))\log p_\pi(y'_t|y'_{<t},q), \tag{6}$$

where y' is a sampled explanation according to the current policy, and $\hat{y}$ is an explanation generated by a greedy decoding. $r(\hat{y})$ is called a baseline reward that stabilizes the training process by reducing the variance in the gradient. To prevent generated explanations from deviating too much from gold explanations, we jointly optimize the RL loss with the supervised loss: our final loss is $L_{\mathrm{RL}} + \lambda L_{\mathrm{ML}}$, where λ is a weight of the ML loss. In our experiments, we used $\lambda = 0.1$.

4.3 Reward function

Given question q, input paragraphs c, and explanation e, we define the reward function as a geometric mean of N elemental reward functions:

$$r(e) = \text{gmean}(\{r_i(q, c, e)\}_{i=1}^{N}) \quad (7)$$

The intuition here is that we combine elemental reward functions with "AND" operator: if one of elemental reward functions gives zero, the explanation must not be rewarded. We introduce three types of elemental reward functions as follows.

Summarization rewards promote the AX to generate more compact summaries. To keep the summary relevant to the question, we also incorporate the relevance of generated explanations to input paragraphs and questions. Let P, Q be a set of tokens, and the P's coverage of Q be $\text{cov}(P, Q) = |P \cap Q|/|Q|$. Let $\text{ng}(X, i)$ be a set of i-grams in X, and $w(X) = \text{ng}(X, 1)$.

- Compression ratio of e w.r.t. input paragraphs: $1 - (\#\text{ tokens in } e / \#\text{ tokens in } c)$
- Abstractiveness of e w.r.t. input paragraphs: $1/4 \sum_i^4 (1 - \text{cov}(\text{ng}(c, i), \text{ng}(e, i)))$.
- Relevance of e to input paragraphs based on unigrams: $\text{cov}(w(c), w(e))$
- e's coverage of question: $\text{cov}(w(e), w(q))$

Sufficiency rewards ensure that generated explanations are sufficient, i.e. useful for answering questions.

- F1 score of the QAM's predicted answer: we feed e into the QAM and calculate the answer F1 score of the predicted answer.
- Existence of gold answer span: 1 if e contains the gold answer span; 0 otherwise.

Comprehensibility rewards ensure the comprehensibility of generated explanations to humans.

- Linguistic acceptability: we feed e into a pretrained CoLA (Warstadt et al., 2018) scorer. In our experiments, we use RoBERTa-base finetuned on the CoLA dataset.[2]
- Sampling noisiness: 1 if $\log p_\pi(e|q, p) > T$; 0 otherwise. This is to prevent noisy explanations from being rewarded. We use $T = -50$.
- Well-formedness: 1 if e has repetition or too long words, starts from pronouns, or ends without period; 0 otherwise.

5 Evaluation

5.1 Dataset

We use HotpotQA (Yang et al., 2018), which consists of 90,564 training and 7,405 development instances.[3] All instances are annotated with extractive explanations called *supporting facts, or SFs*, sentences that are required to answer questions from input documents. We use the distractor setting in our experiments.

For human-annotated explanations, we use $\mathcal{R}^4\mathcal{C}$ (Inoue et al., 2020),[4] which annotates 2,379 training instances (3% of the training instances) and 2,541 development instances from HotpotQA with reasoning steps. The reasoning steps are abstractive explanations that describe information necessary for deriving answers, consisting of entity relation triplets in natural language texts (e.g. *(Biden, is a president of, US)*). We concatenate entities and its relation into one sentence for training the AX.

5.2 Relevant paragraph prediction

To select relevant paragraphs for the AX, we trained a ranker that ranks paragraphs according to its relevance to questions. The ranker takes a question and one paragraph as an input and outputs a relevance score. To train the ranker, we used a binary cross entropy loss, where paragraphs containing gold SFs (henceforth, *supporting paragraphs*) are used as positive instances and the other distractor paragraphs are negative instances. Following Kim et al. (2020), we also randomly sample one supporting paragraph from other questions for each question and used them as negative instances.

At test time, we retain top-k paragraphs and give them to the AX. We use $k = 3$ because HotpotQA has two supporting paragraphs always. Our evaluation shows that all supporting paragraphs are included at top-k ranked paragraphs in 97.4% of dev instances on HotpotQA. When training the AX, we gave gold supporting paragraphs and randomly selected distractor paragraphs to the AX. To implement the ranker, we use a standard sequence classifier on top of RoBERTa-large (Liu et al., 2019).

[2] https://huggingface.co/textattack/roberta-base-CoLA

[3] https://hotpotqa.github.io/

[4] http://naoya-i.github.io/r4c

5.3 Setup

Models We create *Extr*, a simple baseline model that resembles a typical extraction-based explainable NLP architecture (Glockner et al., 2020; Paranjape et al., 2020). Here, we train the AX using Eq. 5 only, where we use SFs as supervision.

We denote our proposed model as *SuQA*. To see the effectiveness of RL, we have *SuQA-NoRL*, a model trained with annotated explanations using Eq. (5) *without additional RL training*. SuQA-NoRL resembles fully-supervised, generation-based explain-then-predict models by Camburu et al. (2018); Rajani et al. (2019).

AX We initialize the AX with DistilBART fine-tuned on CNN/Daily Mail, one of large, standard datasets of summarization (Shleifer and Rush, 2020). During training, we feed supporting paragraphs as an input to the model. At test time, we use predicted relevant paragraphs from §5.2 as an input. For hyperparameter tuning, we reserve 500 training instances as a validation dataset. See §A in Appendix for further details.

QAM We use UnifiedQA-base (Khashabi et al., 2020) as the QAM and freezed it during training. Ideally, the AX should learn from a "perfect" QA model that does not perform disconnected reasoning (Trivedi et al., 2020). However, such a QA model is not available at the moment. We thus simulate it by using UnifiedQA (Khashabi et al., 2020), a T5 (Raffel et al., 2020)-based QA model fine-tuned on a diverse set of QA datasets (e.g. SQuAD, NarrativeQA, RACE) *excluding* HotpotQA. We expect this to discourage the QAM from giving correct answers for insufficient explanations by disconnected reasoning, which improves the quality of reward function of RL. At test time, we use UnifiedQA finetuned on HotpotQA, whose performance is shown in Table 2 (see QAM w/o AX).

5.4 Evaluation measures

Conciseness To assess the *compactness* of generated explanations, we calculate (i) a compression ratio (*Cm*), # tokens in an input paragraph divided by # tokens in a generated explanation, and (ii) abstractiveness (*Abs*) with respect to a given paragraphs selected by the paragraph ranker, calculated by the equation from §4.3.

To assess the *sufficiency* of generated explanations, we use crowdsourcing. Given a generated explanation and its original question, five crowdworkers are asked to judge if generated explanations alone provide sufficient information for answering the question in a 3-point Likert scale (yes, likely, no) plus "unsure". To reliably estimate the quality of explanations, we additionally ask them answers that they inferred from the given explanations.

To aggregate each annotator's judgement, we first replace crowdworker's submission with 'no' when (i) the answer is different from the gold standard answer, or (ii) the judgement is unsure, and replace 'likely' with 'yes'. We then used MACE (Hovy et al., 2013) to aggregate all the judgements (*Suf*). Due to the cost,[5] we evaluate 100 gold explanations and 200 generated explanations for each configuration. We obtained Krippendorff's α of 0.298 on average, indicating a fair agreement. See §D in Appendix for further details of crowdsourced judgement.

In some experiments, we report the similarity between generated explanations and human-annotated explanations as a proxy for sufficiency, due to the cost of human evaluation. We employ ROUGE-2 (Lin, 2004) (*RG2*), which is proven a high correlation between human ratings on several summarization datasets (Bhandari et al., 2020).

QA performance We report *F1*, one of the official evaluation measures of HotpotQA.

Given that our ultimate goal is to create an explainable RC system, we also introduce *XF1*, new evaluation measure:

$$\text{XF1} = \frac{1}{N}\sum_{i}^{N} \text{suf}(i) \cdot \text{F1}(i), \tag{8}$$

where N is the number of instances in the dataset, $\text{suf}(i)$ is a crowdsourced sufficiency label (yes=1, no=0), and $\text{F1}(i)$ is a F1 score of i-th instance. This captures how well the system generates sufficient explanations *and* predicts the correct answer.

5.5 Results and discussion

Abstractive explanations are more concise (i.e. compact and sufficient) than extractive ones. To understand the advantage of abstractive explanations, we compare gold extractive explanations (Gold SF) with gold abstractive explanations (Gold XP) in Table 1. It clearly indicates that abstractive explanations are more abstract and compact than extractive ones. Surprisingly, it also shows that extractive explanations are much less sufficient than

[5] We paid the workers $9/hr.

Input	Abs	Cm	Suf‡	F1
Gold SF†	1.1	4.4	72.0	79.7
Gold SF	1.2	4.3	68.0	74.9
Gold XP†	**51.0**	**11.1**	**90.0**	**85.2**

Table 1: Upper bound study on HotpotQA (HQ) dev set. †: evaluated only on 2,541 dev instances annotated with explanations. ‡: manually evaluated on 100 instances.

Model	Abs	Cm	Suf†	F1	XF1†
QAM w/o AX	0.0	1.0	-	64.2	-
Extr (baseline)	0.3	4.2	70.0	**69.4**	60.5
SuQA-NoRL	40.1	11.2	71.5	65.6	62.6
SuQA	**42.6**	**12.2**	**72.5**	67.6	**63.7**

Table 2: Main results on HotpotQA dev set. †: evaluated on 200 instances with human-judged sufficiency.

abstractive ones. Our manual inspection of insufficient explanations reveals that 100% of the explanations do contain gold answer spans, but the interpretation of them depends on the context of input paragraphs that is not included in the explanations (e.g. pronoun referents). On the one hand, pronouns in abstractive explanations can be replaced with the actual referent, which allows explanations to be more self-contained and compressed. F1 also improved given more sufficient explanations.

The abstractive explainer generates more concise explanations. Now we turn to the proposed models. The results are shown in Table 2. As consistent with Table 1, it shows that SuQA generates more abstractive, compact and sufficient explanations than the extractive baseline model. Examples of sufficient explanations generated by SuQA are shown in Table 4 (see §E in Appendix for more outputs with full input paragraphs). It shows that the abstractive explainer successfully captures information about important entities in question (e.g. bridging entity *World War II* in (b)).

One may think why F1 of SuQA is lower than that of the extractive baseline (-1.8 point) given more sufficient and compressed explanations, which is inconsistent with Table 1. To obtain further insights, we investigated the relation between the sufficiency of explanations and the correctness of answers in Table 3, where "Correct" here means the number of instances with > 0.5 Answer F1.

Table 3 shows that the extractive baseline got 27 correct answers *even when explanations are insufficient* (27/151=17.9%), while SuQA got 17 correct answers for insufficient explanations

	Correct	Wrong
Suf.	124	16
Insuf.	27	33
Total	151	49

(a) Extr (baseline)

	Correct	Wrong
Suf.	128	17
Insuf.	17	38
Total	145	55

(b) SuQA

Table 3: Sufficiency-Answer correctness matrix. SuQA gets more correct answers with sufficient explanations (128/145=88%) than Extr (124/151=82%).

(17/145=11.7%). This suggests that that the QA module relies on task-unrelated lexical cues — so-called disconnected reasoning (Trivedi et al., 2020), and such task-unrelated cues become unavailable in SuQA's more compressed explanations, which undesirably degrades the QA performance. We also experimented with SAE-large (Tu et al., 2020), one of the strong QA models in HotpotQA, but got a similar trend. See §B in Appendix for further details. We believe that QA performance will improve if one can successfully develop a QA model that performs less shortcut reasoning, which is an emerging research topic in the QA community.

The proposed model generates more correct answers with sufficient explanations. Our ultimate goal is to predict correct answers *and* to genereate sufficient explanations. Here we investigate how many instances we generate sufficient explanations *and* predict the correct answer for. Table 3 show that SuQA gets more correct answers with sufficient explanations (128/145=88%) than the extractive baseline (124/151=82%). XF1 in Table 2 reflects this tendency and now tells a different story from conventional F1: the extractive baseline is now behind the proposed model.

RL helps generate concise explanations. As described in §5.3, we pretrain the AX with explanations before applying RL. How much does the additional RL help the AX generate more concise explanations? The results are shown in Table 2 (SuQA-NoRL v.s. SuQA). It indicates that RL is important to obtain more concice explanations in all the aspects of conciseness.

5.6 Analysis

5.6.1 Role of explanation supervision

It is costly to manually annotate QA datasets with abstractive explanations (Inoue et al., 2020). The natural question is then: how much supervision do we need to generate concise explanations?

Question	Generated explanation	Gold answer
(a) Who was born first Burton Cummings or Sharleen Spiteri?	Burton Cummings is born on December 31, 1947. Sharleen Spiteri is born on 7 November 1967.	Burton Lorne Cummings
(b) The Livesey Hal War Memorial commemorates the fallen of which war, that had over 60 million casualties?	Livesey Hall War Memorial commemorates the fallen of World War II. World War II had over 60 million casualties.	World War II
(c) Charles Barton "Chuck" Kendall, Jr. was reportedly interested in purchasing the Los Angeles Clippers from which Jewish-American businessman?	Charles Kendall, Jr. was reportedly interested in purchasing the Los Angeles Clippers from owner Donald Sterling. Donald Sterling is a Jewish-American businessman.	Donald Sterling

Table 4: Sufficient explanations from SuQA. Important entities are gray-highlighted by the author.

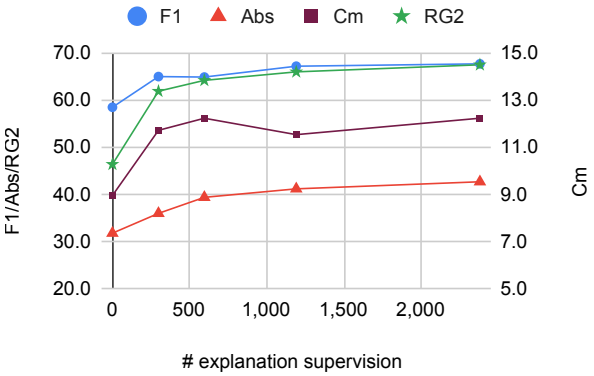

Figure 3: Effect of size of explanation supervision. Our human-judged sufficiency shows 55.0 at size 0 and 66.0 at size 298, indicating the importance of explanation supervision.

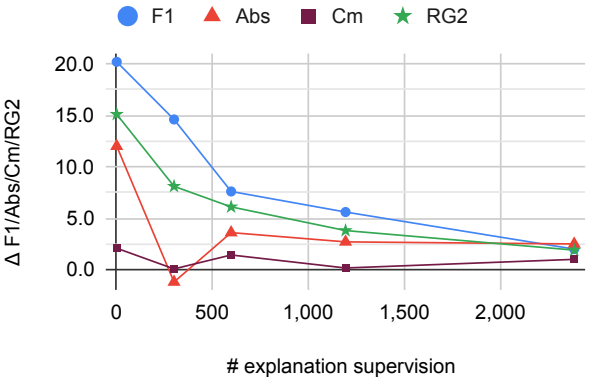

Figure 4: Effect of RL. Y axis indicates the benefit of each evaluation measure from RL (i.e. the difference from SuQA-NoRL to SuQA). The benefit of RL is more pronounced in low-resource settings.

Pretrain?	L_{ML}**?**	**Abs**	**Cm**	**RG2**†	**F1**
SUM	Y	37.5	11.9	64.7	65.3
XG	Y	**47.0**	**13.7**	55.7	54.3
SUM,XG	Y	42.6	12.2	**67.4**	**67.6**
SUM,XG		46.3	12.7	52.1	62.7

Table 5: Ablation of training strategy. Pretraining on the summarization task plays an important role in generating concise explanations. Using seq2seq loss L_{ML} during RL prevents generated explanations from deviating too much from gold explanations. †: evaluated only on 2,541 dev instances annotated with explanations.

We pretrain and apply RL, using various sizes of explanation supervision (0, 298, 595, 1190, 2379) and plotted each result in Fig. 3. Due to the cost of human evaluation, we evaluated 100 generated explanations at size 0 and 298 only, and plotted RG2 as a proxy for human-judged sufficiency.

The results indicate that incorporating even 298 explanations has a large impact on both the conciseness of explanations and the QA performance. Our human-judged sufficiency shows 55.0 for size 0, and 66.0 for size 298. Even with zero explanation supervision, the explainer still generates concise explanations to some extent. This indicates that the task of generating abstractive explanations matches with the pretrained summarizer's original task. Thus, even with such small amounts of data, the AX can learn to produce question-focused summaries that are useful for answering questions.

To see the benefit of RL in low-resource settings, we also repeated the same procedure with SuQA-NoRL and plotted how each evaluation measure changes from SuQA-NoRL to SuQA in Fig. 4. We observe that the benefit of F1 and RG2 is more pronounced in lower resource settings, which indicates the importance of RL for generating concise explanations. See §C in Appendix for the absolute performance of SuQA-NoRL.

5.6.2 Training strategy

Pretraining tasks We pretrain the AX on the summarization task (SUM) and the explanation generation task (XG) (§4.1). To investigate the contribution of each factor, we conduct ablation experiments in Table 5. It shows that the summarization task is the most contributing factor: without the pretraining, we obtain more compact explana-

Insufficiency type	Question	Generated explanation	Gold answer	Freq.
No answer span	In which city was this band formed, whose rhythm guitarist featured in "Cupid's Chokehold?"	Cupid's Chokehold is performed by Gym Class Heroes. Fall Out Boy is formed in Wilmette, Illinois.	Chicago	13
Partially missing	Creed features the boxer who held what WBC title from 2016 to 2017?	Creed (film) features (*missing: the boxer*) Tony Bellew. Tony Bellew held the WBC cruiserweight title from 2016 to 2017.	cruiserweight	8
Bridge fact missing	Where does the descendant of the Red Setter originate?	James Andrew Hanna is known as Red Setter. Scotch Collie originated from the highland regions of Scotland. *Missing: Scotch Collie is the descendant of Red Setter.*	Scotland	3
Fact invented	Which game was released first, Icehouse pieces or Kill Doctor Lucky?	Icehouse pieces was released in 1996 (*correct: 1987*). Kill Doctor Lucky was released in 1996.	Icehouse pieces	1
Dataset flaw	Which Walt Disney film was released earlier, The Rescuers or The Muppets?	The Rescuers was released on June 22, 1977. The Muppets was released in 2011.	The Muppets	3
Worker error	Does Lucozade pre-date Hires Root Beer?	Hires Root Beer is introduced in 1876. Lucozade is created in 1927.	no	2

Table 6: Manual analysis of 30 insufficient explanations from SuQA.

tions, but fatally, they are less similar to the gold explanations and lead to more incorrect answers.

Seq2seq loss We incur the seq2seq loss (L_{ML}) along with the RL loss (§4.2). To see the effect of this, we conduct ablation experiments in Table 5. Without the seq2seq loss, the generated explanations get more compact, but dissimilar to the gold standard explanations. We speculate that the seq2seq loss is important in keeping the search space of the AX closer to gold explanations.

5.6.3 Error analysis

When model's prediction is wrong, we have two possibilities: (A) generated explanations are insufficient, or (B) generated explanations are sufficient, but the QAM fails to find the correct answer. Table 3 indicates that case A is more frequent (69.1% (38/55)) than case B (30.9% (17/55)).

We thus randomly sampled and manually analyzed 30 insufficient explanations generated by SuQA in Table 6. First of all, we found that 43.3% (13/30) of generated explanations have no gold answer spans ('No answer span'). Among the rest of explanations, the AX successfully mentions important entities, but fails to generate some related information such as entity type ('Partially missing', 26.7% (8/30)). We also observed that the AX fails to provide important information bridging two entities such as a family relation ('Bridge fact missing', 10.0% (3/30)), and sometimes the AX invents new fact that is not mentioned in the original input paragraph ('Fact invented', 3.3% (1/30)).

The remaining explanations are wrongly judged as insufficient (16.7% (5/30)) in 2 cases: (i) crowdworkers' answers were wrongly judged as incorrect due to wrong gold answers ('Dataset flaw'); (ii) the crowdworkers' judgement was wrong, and they are actually sufficient ('Worker error').

The error analysis highlighted that a major source of errors is the explainer failing to include answer spans in generated explanations. One can possibly enhance our architecture with one more pass: before generating explanations, the QAM predicts candidate answers based on questions and input paragraphs, and feeds them into the explainer.

6 Conclusions

We have proposed SuQA, an RC system augmented with an abstractive explainer component. Our experiments have demonstrated that the abstractive explainer can generate more concise explanations than an extractive explainer with limited supervison, while keeping explanations sufficient for QA.

One limitation of our work is that the QA module is trained separately from the explainer. One can jointly optimize the AX and QAM by extending our framework. Finally, our abstractive explainer explains what facts were used for answering questions, but does not explain the inference process. It would be an interesting research direction to extend our work by explaining how these facts are combined to arrive at the answer.

Acknowledgements

This work was supported in part by the National Science Foundation under grant No. IIS-1815358 and JST CREST Grant Number JPMJCR20D2, Japan. We thank the anonymous reviewers for the insightful feedback.

Ethical concerns

To make fair compensation for Mechanical Turk workers in human evaluation (§5.4), we setup a reward based on a minimum hourly wage in the United States. Our preliminary experiments show that it takes about one minute to finish one HIT, so we rewarded crowdworkers with $0.15 per HIT. This amounts to $9.00 per hour, which is above $7.25, a minimum wage in the United States.

References

Tal Baumel, Raphael Cohen, and Michael Elhadad. 2016. Topic concentration in query focused summarization datasets. In *AAAI*, pages 2573–2579.

Manik Bhandari, Pranav Narayan Gour, Atabak Ashfaq, Pengfei Liu, and Graham Neubig. 2020. Re-evaluating evaluation in text summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9347–9359, Online. Association for Computational Linguistics.

Oana-Maria Camburu, Tim Rocktäschel, Thomas Lukasiewicz, and Phil Blunsom. 2018. e-snli: Natural language inference with natural language explanations. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

Hoa Trang Dang. 2006. DUC 2005: Evaluation of question-focused summarization systems. In *Proceedings of the Workshop on Task-Focused Summarization and Question Answering*, pages 48–55, Sydney, Australia. Association for Computational Linguistics.

Jay DeYoung, Sarthak Jain, Nazneen Fatema Rajani, Eric Lehman, Caiming Xiong, Richard Socher, and Byron C. Wallace. 2020. ERASER: A benchmark to evaluate rationalized NLP models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4443–4458, Online. Association for Computational Linguistics.

Max Glockner, Ivan Habernal, and Iryna Gurevych. 2020. Why do you think that? exploring faithful sentence-level rationales without supervision. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1080–1095, Online. Association for Computational Linguistics.

Dirk Groeneveld, Tushar Khot, Mausam, and Ashish Sabharwal. 2020. A simple yet strong pipeline for HotpotQA. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8839–8845, Online. Association for Computational Linguistics.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. 2020. Retrieval augmented language model pre-training. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 3929–3938. PMLR.

Dirk Hovy, Taylor Berg-Kirkpatrick, Ashish Vaswani, and Eduard Hovy. 2013. Learning whom to trust with MACE. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1120–1130, Atlanta, Georgia. Association for Computational Linguistics.

Naoya Inoue, Pontus Stenetorp, and Kentaro Inui. 2020. R4C: A benchmark for evaluating RC systems to get the right answer for the right reason. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6740–6750, Online. Association for Computational Linguistics.

Alon Jacovi and Yoav Goldberg. 2020. Towards faithfully interpretable NLP systems: How should we define and evaluate faithfulness? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4198–4205, Online. Association for Computational Linguistics.

Peter A. Jansen, Elizabeth Wainwright, Steven Marmorstein, and Clayton T. Morrison. 2018. WorldTree: A Corpus of Explanation Graphs for Elementary Science Questions supporting Multi-Hop Inference. In *Proc. of LREC*, pages 2732–2740.

Daniel Khashabi, Sewon Min, Tushar Khot, Ashish Sabharwal, Oyvind Tafjord, Peter Clark, and Hannaneh Hajishirzi. 2020. UNIFIEDQA: Crossing format boundaries with a single QA system. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1896–1907, Online. Association for Computational Linguistics.

Hyounghun Kim, Zineng Tang, and Mohit Bansal. 2020. Dense-caption matching and frame-selection gating for temporal localization in VideoQA. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4812–4822, Online. Association for Computational Linguistics.

Sawan Kumar and Partha Talukdar. 2020. NILE : Natural language inference with faithful natural language explanations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8730–8742, Online. Association for Computational Linguistics.

Veronica Latcinnik and Jonathan Berant. 2020. Explaining question answering models through text generation. *arXiv preprint:2004.05569*.

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*, pages 7871–7880, Online. Association for Computational Linguistics.

Patrick 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*, volume 33, pages 9459–9474. Curran Associates, Inc.

Chin-Yew Lin. 2004. ROUGE: A Package for Automatic Evaluation of Summaries. In *Proc. of the Workshop on Text Summarization Branches Out*, pages 74–81.

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. *arXiv preprint:1907.11692*.

Preksha Nema, Mitesh M. Khapra, Anirban Laha, and Balaraman Ravindran. 2017. Diversity driven attention model for query-based abstractive summarization. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1063–1072, Vancouver, Canada. Association for Computational Linguistics.

Bhargavi Paranjape, Mandar Joshi, John Thickstun, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2020. An information bottleneck approach for controlling conciseness in rationale extraction. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1938–1952, Online. Association for Computational Linguistics.

Ramakanth Pasunuru, Asli Celikyilmaz, Michel Galley, Chenyan Xiong, Yizhe Zhang, Mohit Bansal, and Jianfeng Gao. 2021. Data augmentation for abstractive query-focused multi-document summarization. In *AAAI*, pages 13666–13674.

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. *Journal of Machine Learning Research*, 21(140):1–67.

Nazneen Fatema Rajani, Bryan McCann, Caiming Xiong, and Richard Socher. 2019. Explain yourself! leveraging language models for commonsense reasoning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4932–4942, Florence, Italy. Association for Computational Linguistics.

Steven J. Rennie, Etienne Marcheret, Youssef Mroueh, Jerret Ross, and Vaibhava Goel. 2017. Self-critical sequence training for image captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Sam Shleifer and Alexander M. Rush. 2020. Pre-trained summarization distillation. *arXiv preprint:2010.13002*.

Mokanarangan Thayaparan, Marco Valentino, and André Freitas. 2020. A survey on explainability in machine reading comprehension. *arXiv preprint:2010.00389*.

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)*, pages 8846–8863, Online. Association for Computational Linguistics.

Ming Tu, Kevin Huang, Guangtao Wang, Jing Huang, Xiaodong He, and Bowen Zhou. 2020. Select, answer and explain: Interpretable multi-hop reading comprehension over multiple documents. In *AAAI*, pages 9073–9080.

Alex Warstadt, Amanpreet Singh, and Samuel R Bowman. 2018. Neural network acceptability judgments. *arXiv preprint:1805.12471*.

Sarah Wiegreffe and Ana Marasović. 2021. Teach me to explain: A review of datasets for explainable nlp. *arXiv preprint:2102.12060*.

Vikas Yadav, Steven Bethard, and Mihai Surdeanu. 2020. Unsupervised alignment-based iterative evidence retrieval for multi-hop question answering. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4514–4525, Online. Association for Computational Linguistics.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

A Training detail

For all experiments, we used public implementations from huggingface's transformers library available at https://huggingface.co/. We used `roberta-large` for the paragraph ranker, `distilbart-cnn-12-6` for AX, and `unifiedqa-t5-base` for UnifiedQA-base.

For Reinforcement Learning, we used AdamW with the learning rate of 2e-6 and the batch size of 8. We clipped the minimum reward to -0.001. For sampling, we used a temperature of 0.4. To prevent overfitting, we used early stopping with a patience of 5. Specifically, we monitor the Answer F1 on the validation set every 4096 training steps and stopped training if the best F1 is not updated for five times. The RL training took 10h31m on a single GPU (DGXA-100).

For pretraining the AX, we used AdamW with the learning rate of 8e-6 and the batch size of 16. In all experiments, we used a linear learning rate scheduler with 10% warm up and trained the models with 5 epochs. For the learning curve, we monitored the Answer F1 every 128 steps for size 298, 256 steps for size 595, 512 steps for size 1,190 & 2,379 and used early stopping with a patience of 5. We used 512 as a maximum length of input subwords for both the AX and QAM. We used 256 as a maximum length of generation outputs for the AX. We used greedy decoding for both the AX and QAM.

B Experiments with stronger QA model

We conducted additional analysis with SAE-large (Tu et al., 2020), one of the large QA models top-ranked at the leaderboard.[6] We downloaded a publicly available pretrained model[7] and ran the exactly same experiments in Table 1, 2, and 3, where we used SAE-large as the QAM *at test time only*. Note that during training, we used UnifiedQA-base *not* finetuned on HotpotQA (see §5.3 for further details).

The results are shown in Table 7 and Table 8. Overall, they show the same trend as Table 1, 2, and 3: (i) gold abstractive explanations yields higher F1; (ii) SuQA achieved better XF1 than the extractive baseline; and (iii) there are more correct answers led by insufficient explanations in the extractive baseline.

[6] https://hotpotqa.github.io/

[7] https://github.com/JD-AI-Research-Silicon-Valley/SAE

Model	F1	XF1‡
Gold SF†	80.1	-
Gold SF	77.7	-
Gold XP†	**84.4**	-
QAM w/o AX	70.7	-
Extr	**71.5**	59.4
SuQA-NoRL	64.9	58.5
SuQA	66.8	**60.4**

Table 7: Larger QA models on HotpotQA (HQ) dev set. †: evaluated only on 2,541 dev instances annotated with explanations. ‡: evaluated on 200 instances with human-judged sufficiency.

	Correct	Wrong
Suf.	122	14
Insuf.	28	30
Total	150	44

(a) Extr (baseline)

	Correct	Wrong
Suf.	120	18
Insuf.	14	41
Total	134	59

(b) SuQA

Table 8: Sufficiency-Answer correctness matrix. SuQA gets more correct answers with sufficient explanations (120/134=90%) than Extr (122/150=81%).

C Learning curve of SuQA-NoRL

To see the effectiveness of RL in low-resource settings, we investigated the performance change from SuQA-NoRL to SuQA in Fig. 4. Here we plot the absolute performance of SuQA-NoRL in Fig. 5.

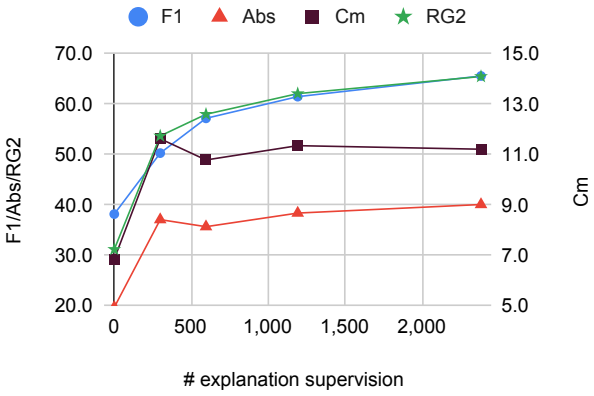

Figure 5: Size of explanation supervision v.s. QA performance and conciseness for SuQA-NoRL.

D Human evaluation

We use Mechanical Turk as a crowdsourcing platform for human evaluation. We hired five annotators per Human Intelligence Task (HIT) and rewarded them with $0.15. Our preliminary experiments show that it takes about one minute to finish one HIT, so it is $9.00 per hour, which is above

$7.25, a minimum wage in the United States. To ensure the quality of annotations, we used crowdworkers with $\geq 5,000$ HITs experiences and $\geq 99\%$ approval rates. Among them, we manually find the pool of high-quality workers and used the same pool throughout the experiments.

The instruction to crowdworkers is shown in Fig. 6 and Fig. 7, and the task interface is shown in Fig 8.

E Example of generated explanations with full inputs

Examples of generated explanations and predicted answers along with their full input paragraphs retrieved by the paragraph ranker are shown in Table 9, Table 10 and Table 11.

1.1 Input and your task

You will be given two pieces of information:

- AI-generated Sentences
- Question

Your first task is the following:

- Judge if you can answer the question solely based on the AI-generated sentences. Your answer choices are **"Yes", "Likely", "No"**, and **"Unsure"**.

The meaning of each choice is as follows:

- **Yes**: I'm sure I can answer the question. All information needed is there.
- **Likely**: I can answer the question, but I have to guess something.
- **No**: I cannot answer the question because important information is missing.
- **Unsure**: It's difficult to judge.

General Guildines:

- **Select Yes**: When all information is present, and you have to make no assumptions or very reasonable assumptions.
- **Select Likely**: When most information is present, but you have to make a reasonable assumption
- **Select No**: When critical information is missing, and you must make an unreasonable assumption.
- **Select Unsure**: When it's difficult to judge.

You will have to use your best judgement in determining what is an unreasonable versus reasonable vs very reasonable assumption.

Your second task is the following:

1. If you select **"Yes"** or **"Likely"**: tell us your answer.
2. If you select **"Likely"** or **"No"**: please tell us what information is missing for answering the question in a text box.
3. If you select **"Unsure"**: tell us the reason in a text box.

Figure 6: Instruction for crowdworkers (general guidelines).

1.2 Examples

Sentences (input)	Question (input)	Your answer should be...
Krzysztof Zanussi was born on 17 June 1939. Thom Andersen was born in 1943.	Who was born first, Krzysztof Zanussi or Thom Andersen?	**Judge**: Yes **Answer**: Krzysztof Zanussi **Exp**: All necessary information is present, **no assumptions** required.
Raj Kapoor was a noted Indian film actor. Mike Cahill is an American film director and screenwriter.	What profession do Raj Kapoor and Mike Cahill share?	**Judge**: Yes **Answer**: Nothing **Exp**: All necessary information is present, **no assumptions** required.
K-Y Jelly is a water-based, water-soluble personal lubricant. Johnson & Johnson is founded in 1886.	What company founded in 1886 has owned the K-Y brand?	**Judge**: No **Missing information**: which company owns the K-Y brand. **Exp**: It is an **unreasonable assumption** to assume that J and J owns this brand just because it was founded in 1886.
Mexico is a small mill town for the papermaking industry. Mexico is a town in Oxford County, Maine, United States.	What is the main industry of this town with a population of around 2,600 as of 2010 that has Mountain Valley High School attendees as residents?	**Judge**: No **Missing information**: The population of the town as of 2010. The fact that Mountain Valley High School atendees are residents. **Exp**: It is an **unreasonable assumption** to assume that Mexico has a certain population or high school.
I Love You was written by Chris White. I Love You was covered by People! and The Carnabeats.	Who covered the song I Love You by Chris Write?	**Judge**: Likely **Answer**: People! and The Carnabeats. **Missing information**: "I Love You" is a song. **Exp**: It is **reasonable** to assume that "I Love You" is a song, since the word cover is usually used for songs.
Johnny Angel (film) stars George Raft. George Raft was an American film actor and dancer.	Which American film actor and dancer starred in the 1945 film Johnny Angel?	**Judge**: Likely **Answer**: George Raft. **Missing information**: Johnny Angel is a 1945 film. **Exp**: It is **reasonable** to assume that Johny Angel is the same film as Johnny Angel (1945), since films usually don't have the same title.

Figure 7: Instruction for crowdworkers (examples).

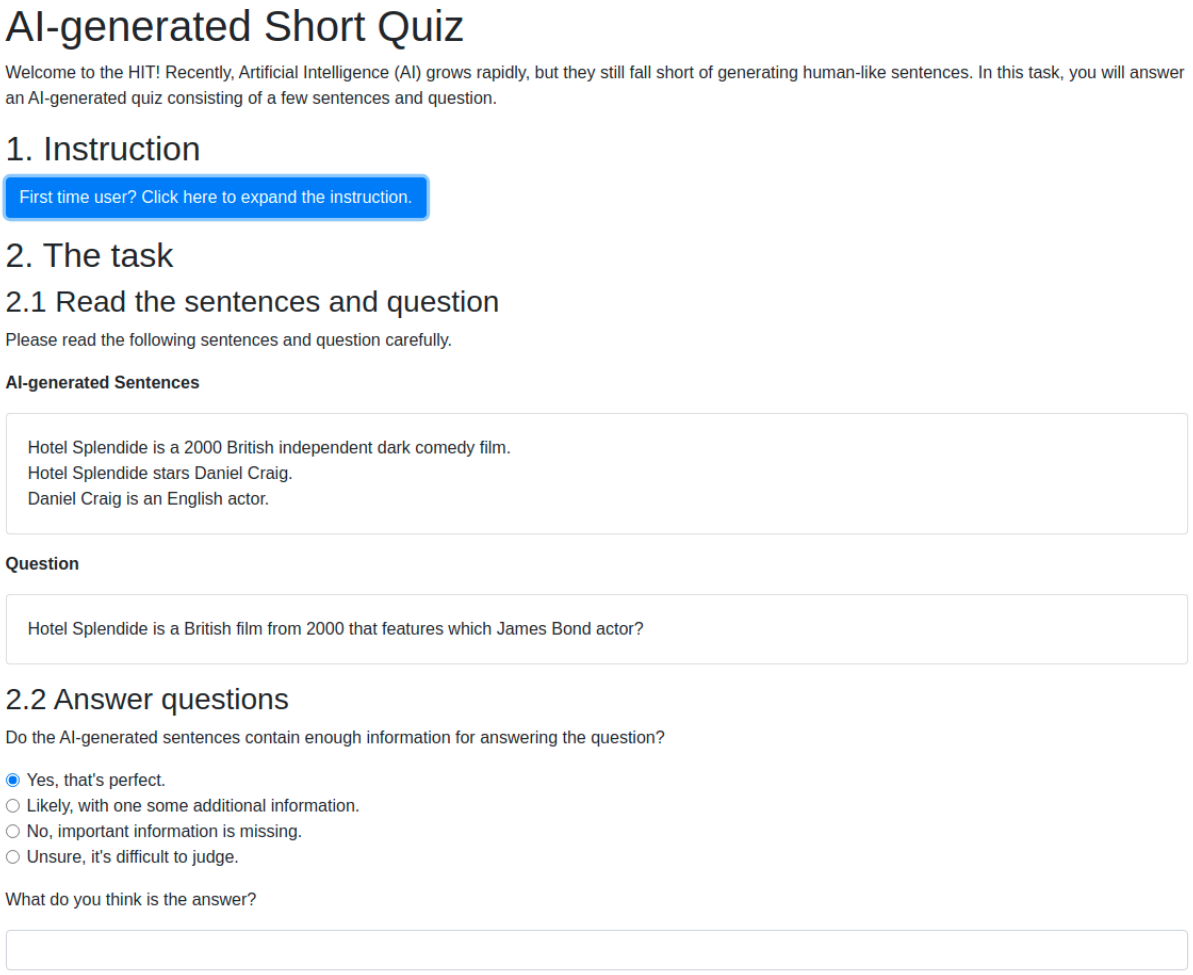

Figure 8: Crowdsourcing interface for sufficiency evaluation.

Question	Input paragraphs from the ranker	Generated explanation	Predicted answer
Charlie Rowe plays Billy Costa in a film based on what novel?	[P1] Charles John Rowe is an English actor. His film roles include Young Tommy in "Never Let Me Go", James in "The Boat That Rocked", Billy Costa in "The Golden Compass" , Peter in the SyFy/Sky Movies Peter Pan prequel "Neverland", and recently played Leo Roth on the Fox medical comedy-drama series "Red Band Society" alongside Octavia Spencer and Dave Annable. [P2] Cloud is a 2010 American drama film based on Ben Sherwood's best-selling novel, "The Death and Life of Charlie St. Cloud" published in 2004 by Bantam Books. The film is directed by Burr Steers and stars Zac Efron and Amanda Crew. The story is of Charlie St. Cloud's choice between keeping a promise he made to his brother, who died in a car accident, or going after the girl he loves. In some markets the film used the complete title of the book. [P3] The Golden Compass is a 2007 British-American fantasy adventure film based on "Northern Lights", the first novel in Philip Pullman's trilogy "His Dark Materials". Written and directed by Chris Weitz, it stars Nicole Kidman, Dakota Blue Richards, Daniel Craig, Sam Elliott, Eva Green, and Ian McKellen. The project was announced in February 2002, but difficulties over the script and the selection of a director caused significant delays. At US$180 million, it was one of New Line Cinema's most expensive projects ever, and its disappointing results in the USA contributed to New Line's February 2008 restructuring.	Charlie Rowe plays Billy Costa in The Golden Compass. The Golden Compass is based on Northern Lights.	Northern Lights

Table 9: Full input and output example (1).

Question	Input paragraphs from the ranker	Generated explanation	Predicted answer
Who was born first, Krzysztof Zanussi or Thom Andersen?	[P1] Krzysztof Zanussi, (born 17 June 1939) is a Polish film and theatre director, producer and screenwriter. He is a professor of European film at the European Graduate School in Saas-Fee, Switzerland where he conducts a summer workshop. He is also a professor at the Silesian University in Katowice. [P2] Thom Andersen (born 1943, Chicago) is a filmmaker, film critic and teacher. [P3] Weronika Anna Rosati (] ; born 9 January 1984) is a Polish actress and a member of European Film Academy. She began her acting career in Polish soaps. In 2005, she starred as Dżemma in her first theatrical feature film "Pitbull". A year later, she launched her international career with a small uncredited role in "Inland Empire" directed by David Lynch. Since then she has appeared in many critically acclaimed Polish and international productions. In 2013, she has received her first Polish Academy Award nomination for Best Actress for her role in "Obława" (2012). A year later, she starred alongside Agnieszka Grochowska in "Obce ciało" directed by Krzysztof Zanussi. She also had a recurring role in the HBO TV series "Luck" (2012).	Krzysztof Zanussi is born on 17 June 1939. Thom Andersen is born on 1943.	Krzysztof Zanussi

Table 10: Full input and output example (2).

Question	Input paragraphs from the ranker	Generated explanation	Predicted answer
Kalle Mäkinen is nicknamed after a Spanish football that captains what two teams?	[P1] Kalle Mäkinen (born 1 February 1989) is a Finnish footballer who last played for the Finnish Veikkausliiga club Maskun Palloseura. He was nicknamed "Sergio" because his style of play was similar to that of Spanish fullback Sergio Ramos. [P2] Sergio Ramos García () ; born 30 March 1986) is a Spanish professional footballer who plays for and captains both Real Madrid and the Spain national team. Primarily a central defender, he can also play as a right back. [P3] The Primera División, commonly known as La Liga and as La Liga Santander for sponsorship reasons with Santander, is the top professional association football division of the Spanish football league system. Administrated by the Liga de Fútbol Profesional (LFP), La Liga is contested by 20 teams, with the three lowest-placed teams relegated to the Segunda División and replaced by the top two teams in that division plus the winner of a play-off.	Kalle Mäkinen is nicknamed Sergio Ramos. Sergio Ramos captains both Real Madrid and the Spain national team.	Real Madrid and the Spain national team

Table 11: Full input and output example (3).