Generating Coherent Narratives with Subtopic Planning to Answer How-to Questions

Pengshan Cai,[1] **Mo Yu,**[2] **Fei Liu,**[3] **Hong Yu**[1,4]
[1]Manning College of Information & Computer Sciences, University of Massachusetts, Amherst
[2]Wechat AI, Tencent
[3]Department of Computer Science, Emory University
[4]CHORDS, University of Massachusetts, Lowell

Abstract

Answering *how-to* questions remains a major challenge in question answering research. A vast number of narrow, long-tail questions cannot be readily answered using a search engine. Moreover, there is little to no annotated data available to help develop such systems. This paper makes a first attempt at generating coherent, long-form answers for *how-to* questions. We propose new architectures, consisting of passage retrieval, subtopic planning and narrative generation, to consolidate multiple relevant passages into a coherent, explanatory answer. Our subtopic planning module aims to produce a set of relevant, diverse subtopics that serve as the backbone for answer generation to improve topic coherence. We present extensive experiments on a WikiHow dataset repurposed for long-form question answering. Empirical results demonstrate that generating narratives to answer *how-to* questions is a challenging task. Nevertheless, our architecture incorporated with subtopic planning can produce high-quality, diverse narratives evaluated using automatic metrics and human assessment.[1]

1 Introduction

How-to question (e.g., "How to turn off news notification on my phone?") is an important question type. To find answers, most people resort to internet search. However, the answers usually scatter in different web pages, making the search time-consuming and inefficient. To remedy this issue, Wikihow.com offers a platform for experts to share their answers to *how-to* questions. Despite its valuable content, the total volume of Wikihow articles is still limited. By far, Wikihow has only collected around 74K articles, too few when compared to other open-collaborative databases (e.g. over 6M articles in Wikipedia, over 90M items in Wikidata). As a result, it would be valuable to both editors and readers if NLP technology could be applied to automatically generate Wikihow entries to provide high quality answers to *how-to* questions.

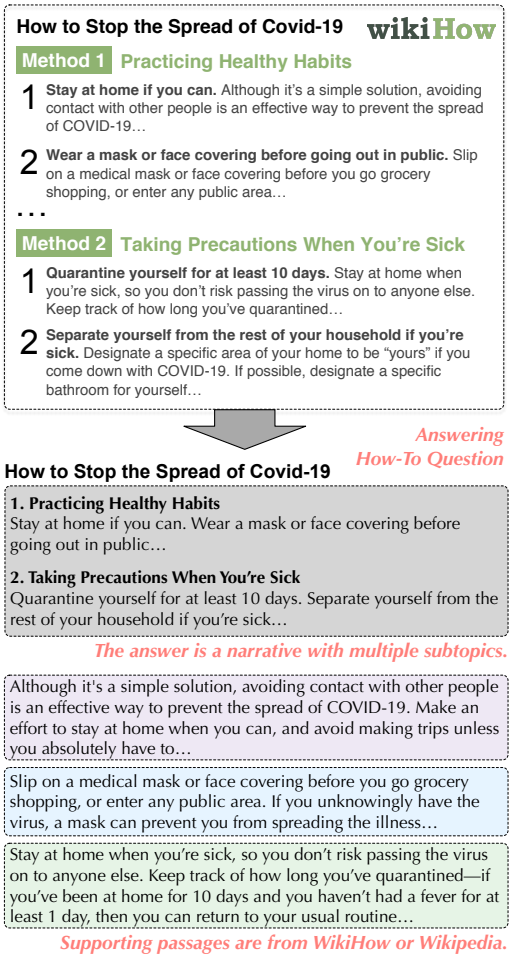

Figure 1: An example *how-to* question, its long-form answer and supporting passages.

Recent advances in generative QA researches have made it possible to generate answers to non-factoid questions (Tan et al., 2018; Nishida et al., 2019; Izacard and Grave, 2020). However, they have two limitations in generating an Wikihow entry: First, Wikihow presents answers in a hierarchical structure: an answer usually contains several sections, each led by a succinct subtopic. This

[1]We have made our dataset and source code available at https://github.com/pengshancai/how-to-QA

Proceedings of the 2nd Workshop on Natural Language Generation, Evaluation, and Metrics (GEM), pages 26 - 42

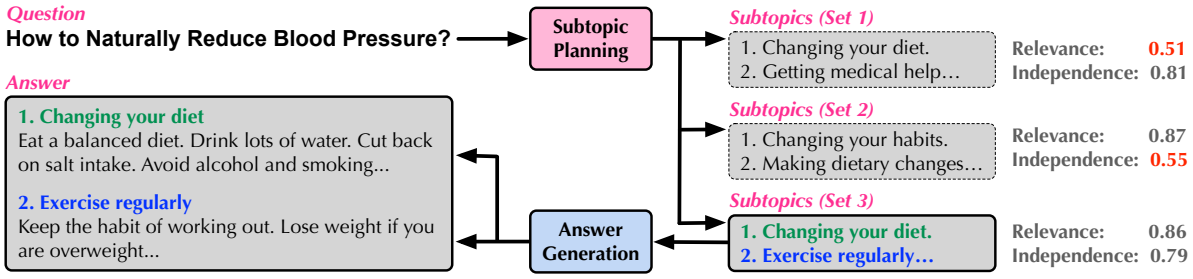

Figure 2: An overview of our *Arc-P* architecture for answering how-to questions. Given a question, our **subtopic planning module** generates multiple sets of subtopics using a sampling approach. It then automatically selects an optimal set of subtopics by measuring their *Relevance* and *Independence* (section 2.4). The selected subtopics are finally sent to the **answer generation module** to provide guidance to narrative generation.

structure helps readers with an overview of the answer scope before locating the details (Hearst and Pedersen, 1996). On the contrary, answers generated by current QA models are usually unstructured. Second, an informative and detailed answer is usually a long sequence of text. Due to exposure bias (He et al., 2019), long text generation suffers from the risk that the quality of generated text deteriorates as its length increases.

In this paper, we present a two-step approach to answering *how-to* questions. For each question, our **subtopic planning** module first generates several subtopics to cover different aspects of the question. Based on each subtopic, our **answer generation** module then generates a paragraph of text to elaborate the subtopic. Compared to previous generative question answering methods, our architecture has two advantages: (1) Subtopic planning allows presenting answers in a well-organized structure, which helps our readers to easily grasp the general idea of the answer, offering an easy-to-follow reading experience. Moreover, by generating answers from various subtopics, our architecture is able to provide answers with higher diversity in content. (2) Our architecture decreases the risk of exposure bias in the generated texts, by breaking down the answer into multiple shorter subsections instead of directly generating a long paragraph. In this way, it improves the generated answer's quality.

The quality of subtopics is crucial for the generation of long-form, explanatory answers. While a set of good subtopics can improve answer generation, a set of bad subtopics may digress the answer from the question. We observe two types of common mistakes in subtopic planning: (1) **Irrelevant subtopics**: a generated subtopic is not closely related to the question. For example, in Figure 2, the subtopic *Getting medical help* in the first subtopic set is not related to the question *how to naturally reduce blood pressure*; (2) **Redundant subtopics**: two or more generated subtopics to the same question have semantic overlap. For instance, in the second subtopic set in Figure 2, the subtopic *Making dietary changes* is included in the subtopic *Changing your habits*. This may further lead to repetition in subsequent answer generation.

We deal with these problems with an additional subtopic set selection step in the subtopic planning module. The key idea is to first generate multiple subtopic sets, and then evaluate their *relevance* and *independence* to select one optimal subtopic set for next-step processing. Specifically, we measure these two quantities in the vector space: We use a paragraph encoder (which in our work is the Dense Passage Retrieval model (Karpukhin et al., 2020)) to map the question and all the generated subtopics into a common embedding space. We then measure *relevance* and *independence* according to the subtopic vectors' relative distances and their distances to the question vector.

The contributions of our paper include:

- A novel *how-to* question answering architecture based on subtopic planning;
- An efficient vector-space model to evaluate and select high-quality subtopics to a question in terms of *relevance* and *independence*;
- Extensive experiments with human study that both prove the effectiveness of our methods and explore factors affecting the quality of *how-to* question answering results.

2 Model

2.1 General Architectures

A **retrieve-generate paradigm** consists of a retriever and a generator. Given a query x, the retriever first collects K supporting paragraphs

$P = \{p_1, ..., p_K\}$ from a large text corpus. The supporting paragraphs are expected to include external knowledge relevant to to x. The generator then outputs the result y based on x and P. We consider the following two architectures:

1) **Arc-Direct** (*Arc-D*) directly generates the answer from the question by applying the *retrieve-generate paradigm*;

2) **Arc-Planning** (*Arc-P*) (Figure 2) is composed of a subtopic planning module and an answer generation module. It generates an answer in a two-step manner: First, the subtopic planning module decomposes q into a set of N subtopics $S = \{s_1, ..., s_N\}$, for each subtopic $s_i, i \in \{1, ..., N\}$, the answer generation module generates a paragraph of explanation to the subtopic. We present all the subtopics and their explanation as the final answer. Both subtopic planning and answer generation modules apply the *retrieve-generate paradigm*. Specifically, during subtopic planning, all the subtopics are generated in one single sequence, and divided by a special separation token. During answer generation, supporting paragraphs for each individual subtopic are retrieved based on the question and the subtopic.

2.2 Retriever

We use dense passage retriever (DPR) (Karpukhin et al., 2020), a neural retrieving model as our retriever. DPR is composed of a query encoder E_X and a paragraph encoder E_P. E_X and E_P respectively encodes the query and potential paragraphs into vectors in a common embedding space, the relevance score of x and p is calculated as the inner product between two vectors.

$$sim(x, p) = E_X(x)^T E_P(p) \tag{1}$$

After gaining the relevance score of x to each paragraph in the large text corpus, we rank the paragraphs according to their relevance score, the top K paragraphs are selected as supporting paragraphs to x. Note that we use the same DPR retriever to retrieve supporting paragraphs for both question queries and subtopic queries. This embeds questions and subtopics in the same space, thus enables measuring the semantic relevance of a question and its subtopics as will be described in Section 2.4.

2.3 Generator

We employ BART-LARGE (Lewis et al., 2020a), a seq2seq Transformer model (Vaswani et al., 2017) pretrained with a denoising objective as our generator. The generator takes as input a concatenation of the query x and its relevant paragraphs P and outputs a sequence of words $y = \{y_1, ..., y_L\}$, where L is the length of y. Formally,

$$p(y|x, P) = \prod_{l=1}^{L} p(y_l|x, P, y_{1:l-1}) \tag{2}$$

We leverage BART to decode multiple subtopics, conditioned on the question and its supporting paragraphs. All subtopics of a question are concatenated to form the target sequence. To predict the l-th word of the sequence, our generator computes a probability distribution over the vocabulary tokens $p(w|q, P, y_{1:l-1})$. Instead of the argmax inference

$$y_l = \arg\max_w p(w|x, P, y_{1:l-1}), \tag{3}$$

we perform sampling from the top-*k* most probable tokens to obtain

$$y_l \sim p(w|x, P, y_{1:l-1}). \tag{4}$$

Our subtopic planning module attempts to generate multiple sets of subtopics using a sampling method. The method is advantageous over greedy decoding, which tend to produce high-likelihood rather than diverse sequences (Ippolito et al., 2019). The sets of subtopics will be subsequently measured by their *relevance* and *independence* to identify an optimal set of subtopics.

2.4 Subtopics Selection

While the generators are able to generate locally fluent text, they do not guarantee global semantic optimality (Holtzman et al., 2020, 2018). Specifically, during experiments, we observed that for the same question, the quality of different subtopic sets generated using *top-k* sampling from the same question vary greatly (Section 3.4). In this section, we explore the following question: How to automatically single out a high quality subtopic set from various generated ones.

By observing the subtopic planning results, we realize there exist two types of common mistakes in subtopic planning: 1. The subtopics is irrelevant to the question. 2. Subtopics have semantic overlap with each other.

We present a simple yet effective method to filter subtopics which may lead to the above mistakes. To this end we reuse the trained DPR retriever to measure the generated subtopic sets from two perspectives: 1. *Relevance*: How relevant each subtopic is

to the question and 2. *Independence*: How much overlap the subtopics have with each other.

Formally, given a question q and a subtopic set S, we first transform q and each subtopic s_i in S into vectors $E_X(q)$ and $E_X(s_i)$ in the DPR embedding space. We define the neighbors of x, denoted as $\mathcal{N}_M(x)$, as the top M paragraphs whose DPR embeddings are closest to $E_X(x)$, where M is a human specified hyper-parameter.

Measuring relevance: We measure the relevance of a question q and a subtopic s_i as follows:

$$\text{relv}(q, S) = \frac{\sum_{s_i \in S} \#\left(\mathcal{N}_M(q) \cap \mathcal{N}_M(s_i)\right) / M}{|S|} \tag{5}$$

where $\#(\cdot)$ refers to the number of elements within a set. A higher relevance score indicates more paragraphs relevant to the question also relates to the subtopic, implying their semantic relevance.

Measuring Independence: We measure the independence of a set of subtopics as follows:

$$\text{indp}(S) = Avg\left(1 - \frac{\#(\mathcal{N}_M(s_i) \cap \mathcal{N}_M(s_j))}{M}\right) \tag{6}$$

A high independence score indicates a paragraph relevant to one subtopic may not be related to the other subtopics, implying low semantic overlap among subtopics.[2]

Selecting subtopics: A set of good subtopics needs to be balanced in both relevance and independence. As a result, we select a set of subtopics with the maximum independence subject to the minimum relevance of the subtopics greater than a human specified threshold τ.

2.5 Training

Retriever. When retrieving supporting paragraphs for a question q, we use the question itself as the query. When retrieving supporting paragraphs for a subtopic s, we use the concatenation of the q and s as the query. We train the retriever by optimizing the negative log likelihood loss:

$$L_R = -\log \frac{e^{\text{sim}(x,p^+)}}{e^{sim(x,p^+)} + \sum_{j=1}^{n} e^{\text{sim}(x,p_j^-)}} \tag{7}$$

where x is a query, positive paragraph p^+ is a gold supporting paragraph related to x, negative paragraphs $\{p_1^-, ..., p_n^-\}$ are randomly sampled from paragraphs unrelated to x.

Generator. After finished training the retriever, we use the retriever to collect supporting paragraphs as a part of training data to the generators[3]. When training the generator of both *Arc-D* and *Arc-P*'s subtopic planning module, we use the concatenation of the question and supporting paragraphs as input. When training the generator of *Arc-P*'s answer generation module, we use the concatenation of the question, subtopic and supporting paragraphs as input. We minimize the following maximum-likelihood objective function:

$$L_G = -\sum_{l=1}^{L} \log(p(y_l^*|x, P, y_{1:l-1}^*)) \tag{8}$$

where $y^* = \{y_1^*, ..., y_L^*\}$ is the ground-truth output sequence.

3 Experiments

3.1 Datasets

We build a dataset *HowQA* by reformatting *WikiSum* (Koupaee and Wang, 2018), a summarization collected from *Wikihow*. Overall, our *HowQA* contains 72.4K Wikihow articles, we randomly split them into training/validation/test sets. As shown in Figure 1, each Wikihow article contains one *How to* question and several subtopics, each subtopic is followed by a few description paragraphs. Each paragraph starts with a summary sentence which summarizes the meaning of the rest of the paragraph. We collect all the summary sentences as the *explanation* to the subtopic, and the rest of the paragraphs as *gold supporting paragraphs* to the subtopic. We consider two large text corpora as source of supporting paragraphs: A. Wikihow training set; B. Wikipedia[4]. *HowQA* is used for training and testing both retriever and generator. We show the statistics of our *HowQA* in Table 1[5]

[2]We use the overlap of neighbors instead of the vector cosine similarity to calculate *relevance* and *independence* as the former directly reflects the overlap of potential supporting paragraphs input to the generator.

[3]We use the retrieved supporting paragraphs instead of gold supporting paragraphs as this better mimics the situation in test time, where gold supporting paragraphs are unavailable.

[4]We use Wikipedia as it is a trustworthy information source for a knowledge intensive task like ours. By default, results reported in the experiment are based on using Wikihow training set as the supporting paragraph corpus unless stated otherwise.

[5]See Appendix for implementation and evaluation details.

Dataset	# Questions	# Subtopics (Explanations)	# Supporting Paragraphs
Train	69,990	214,549	1,074,129
Validation	1,200	3,692	18,423
Test	1,231	3,770	18,680

Avg Length of Subtitle	4.02
Avg Length of Explanation	35.14
Avg Length of Wikihow-train Paragraphs	68.89
Avg Length of Wikipedia Paragraphs	100.00
# Paragraphs in Wikihow-train	1.1M
# Paragraphs in Wikipedia	21.0M

Table 1: Statistics of our *HowQA*.

3.2 Evaluation Metrics

Automatic Metrics. We evaluate our overall performance with two sets of metrics: (1) Surface-form coverage metrics, including ROUGE-1, ROUGE-2, ROUGE-L and METEOR that are commonly used in automatic summarization and machine translation. These metrics measure how many words in the groundtruths are covered by the generated answers; (2) Diversity metrics (Li et al., 2016; Hua et al., 2019a) is calculated by the number of distinct unigrams (distinct-1), bigrams (distinct-2) and trigrams (distinct-3) in the generated answers. The values are scaled by the length of the answer to avoid favoring long text. The best-performed system should have both high coverage and diversity scores.

For the study of retrievers, we evaluate two common document retrieval metrics: (1) Hit@10 (Bordes et al., 2013), i.e., the proportion of gold supporting paragraphs ranked in the top 10; (2) Mean reciprocal rank (MRR) (Radev et al., 2002).

Human Evaluation Metrics. We note that surface-form coverage metric may not necessarily reflect the quality of generated answers. It is limited in a situation when there are multiple ways to decompose a *How-to* question into subtopics. For instance, for the question "How to deal with loneliness", one subtopic set may emphasize embracing and enjoying loneliness, another may focus on encouraging social interaction and improving social skills. Both subtopic sets make good sense, their contents vary greatly. We thus resort to human judges to evaluate the quality of our answers.

Specifically, we collected our evaluation results using Amazon Mechanical Turk[6]. We randomly sample 90 questions from our test set. To evaluate **answer generation**, we present judges with several answers generated by different methods to the same question, and let judges rank their preference for the answers from the best to the worst. We report the average ranking for each model and pairwise preference for each pair of model. To evaluate **subtopic selection**, we ask our judges to evaluate subtopic sets from three perspectives: (1) *Relevance:* How close is each subtopic set related to the question; (2) *Independence:* How independent is each subtopic to the other subtopics in the same subtopic set; (3) *Overall Preference:* How do judges like the subtopic set. The grading scale for each perspective is from 1 to 3. For both tasks, each result is evaluated by ten individual judges.

3.3 Overall Performance with and without Subtopic Planning

In this section, we explore the effects of subtopic planning by comparing the overall answer generation performance of *Arc-D* and *Arc-P*. Evaluation results show that our proposed *Arc-P* gives clear advantage in both automatic and human evaluation.

• **Surface-form Coverage.** From Table 2 we observed that *Arc-P* achieves better performance than *Arc-D* in ROUGE and METEOR. This demonstrates our *Arc-P* generates better answers than *Arc-D* in terms of coverage of the original answer.

• **Content Diversity.** Table 2 shows the diversity degree of *Arc-D*, *Arc-P* and the golden answers. *Arc-P* significantly outperforms *Arc-D* in all distinct-1, 2 and 3. This proves that by answering questions from various perspectives, *Arc-P* is able to generate answers that are more diversified in content. However, there still exists gaps between *Arc-P* and the gold answer, this implies our generated answers still do not match human written answers in content diversity.

• **Human Evaluation.**[7] Given a question, we ask human judges to rank their preference for the gold answer and answers generated by *Arc-D* and *Arc-P* respectively. From Table 3 we observed human judges prefer answers from *Arc-P* much more than *Arc-D*'s. Comparing to *Arc-D*, our *Arc-P* wins on more than 72% of the cases. This demonstrates the effectiveness of subtopic planning.

The results also show that neither *Arc-D* nor *Arc-P* is comparable to human performance. A closer investigation of the machine generated answers

[6]We require our judges to have at least 100 previous jobs and greater than 95% acceptance rate.

[7]We present more examples and analysis in the appendix.

Models	Coverage				Diversity		
	ROUGE-1	ROUGE-2	ROUGE-L	METEOR	Distinct-1	Distinct-2	Distinct-3
Arc-Direct	26.13	6.37	25.54	16.64	57.85	73.64	78.42
Arc-Planning	**28.3**	**7.11**	**25.63**	**17.47**	**62.45**	**83.60**	**89.88**
Gold Answers	n/a	n/a	n/a	n/a	68.51	90.61	94.52

Table 2: Performance of *Arc-D* and *Arc-P* in answer generation. *Arc-P* achieves higher ROUGE and METEOR score.

Single Model	Average Rank ↓
Arc-D	2.48
Arc-P	**1.87**
Gold	**1.65**
Model Pair	**Prefer Rate ↑**
Arc-P > *Arc-D*	**72.22%**
Arc-P > *Gold*	**40.66%**
Arc-D > *Gold*	23.55%

Table 3: Average ranking of each model and pairwise preference rate between each pair of models.

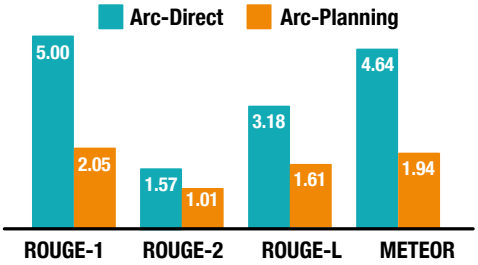

Figure 3: Score difference between part1 and part2.

Subtopic Set	R-1	R-2	R-L	METEOR
Low relv	27.53	6.59	24.83	17.27
Low indp	28.12	6.98	**25.64**	**17.57**
Selected	**28.3**	**7.11**	25.63	17.47
Oracle	28.93	8.11	25.96	19.87

Table 4: Automatic evaluation scores of answer generation when using different subtopic sets. *Selected* refers to the best subtopics set by our selecting metrics, *Low relv* and *Low indp* respectively refers to choosing the subtopic set with the lowest relevance and independence scores, *Oracle* refers to the oracle subtopics in Wikihow articles.

Subtopic Set	Relevance	Independence	Overall
Low relv	2.15	2.21	2.12
Low indp	2.30	2.07	2.02
Selected	**2.39**	**2.30**	**2.25**
Oracle	2.40	2.32	2.29

Table 5: Human evaluation scores for each subtopic set.

show that they sometimes make obvious logical mistakes. For example, for the question *How to Use Google Shopping Express*, the machine presents a generated fake sign-up URL.

Analysis of Degradation in Long Generation. We conduct analysis on *Arc-P*'s effectiveness. Our hypothesis is that by generating diverse answers, all parts of an answer from *Arc-P* convey useful information. To verify this, we design the following experiment: For each generated answer from *Arc-D* and *Arc-P*, we cut the answer into two parts of equal length from the middle. We then calculate each part's ROUGE and METEOR score to the golden standard. Figure 3 presents the score differences between part1 and part2. The performance gaps from *Arc-D* are significantly large, implying our *Arc-P* effectively avoids the performance degradation in long-form text generation, and generates answers of more consistent quality.

3.4 Effects of Subtopic Selection

In this section we analyze the effect of our subtopic selection algorithm in Section 2.4. We compare the qualities of our selected subtopics (*Selected*) with two variations: (1) selecting subtopic sets with lowest relevance (*Low relv*) and (2) selecting subtopic sets with lowest independence (*Low indp*). Similar to our overall evaluation, we conduct both the automatic evaluation and human evaluation as below:

Automatic Evaluation. Table 4 shows the performance of *Arc-P*'s answer when using different subtopic selections. We find that *Selected* gives better performance than *Low relv*. This implies irrelevant subtopics deteriorate the quality of subsequent answer generation. However, *Selected* shows similar performance to *Low indp* under the automatic metric. Considering the large performance gap from human evaluation, this confirms the problem of these coverage-based metrics that they overlook the semantic repetition of generation results.

Human Evaluation. Table 5 presents human evaluation results of the subtopic planning module of *Arc-P*. We observed score difference between *Selected*, *Low relv* and *Low indp*. This proves there exists quality discrepancy between different subtopic sets generated from the same question using *top-k* sampling. Specifically, *Low relv* and *Low indp* achieve the worst performance in *Relevance* and *Independence* respectively, this implies

	Category	R-1	R-2	R-L	METEOR	Portion (%)
WORST	*Work World*	25.17	4.27	22.1	16.52	1.72
	Health	25.72	5.97	25.24	16.94	14.21
	Holidays and Traditions	25.82	5.37	24.07	14.82	0.81
	Education and Communications	26.01	5.72	24.26	16.83	8.46
	Work World	26.01	3.93	26.56	15.64	0.7
BEST	*Cars & Other Vehicles*	30.15	8.11	26.17	18.32	1.7
	Youth	31.27	7.69	28.46	**21.22**	3.21
	Personal Care and Style	31.45	9.11	**28.67**	19.23	6.73
	Pets and Animals	32.52	9.96	26.99	20.13	6.6
	Food and Entertaining	**32.58**	**10.49**	27.17	19.49	8.03

Table 6: Performance of different categories. The performance between categories vary greatly (e.g. The highest difference in ROUGE-1 is over seven points). The training data portion for each category also varies greatly.

our vector space based metric is consistent with human rankings, and could distinguish unrelated and semantically overlapped subtopics. Moreover, *Selected* outperforms both *Low relv* and *Low indp* in all three metrics, and even achieves similar performance to *Gold*. This demonstrates our subtopic selection methods could effectively select subtopic sets with a higher quality.

3.5 Effects of Question Categories

Wikihow classifies all questions into 20 categories. We show in Table 6 the performance of 5 categories with the highest ROUGE-1 F1 score and 5 categories with the lowest[8]. From the table we have the following observations: 1. There exists a great discrepancy between performance of different categories. 2. Generally, if a category contains more data in the training set, it is more likely to demonstrates better performance. 3. However, a few categories with more training data (e.g. *Education and Communications*) achieved much lower performance than categories with less training data (e.g. *Cars and Other Vehicles*). We argue this is because in some categories, the answers to questions follow some specific routines. While in other categories, there is no answer routine to follow. For example, the category *Cars and Other Vehicles* contains many questions about installing car parts, e.g. *how to install a car starter*, *How to install a car stereo*, etc. The answer to these questions usually starts with removing the old car parts (Set the parking brake, stall the car, take out the old car parts, etc.) On the contrary, the category *Education and Communications* contains many primary/middle school math questions e.g. *How to calculate volume*, *how to divide double digits*, etc. The answers to these questions do not follow any pattern, and can not be answered routinely.

[8]We omit categories with <10 instances in the test set.

3.6 Effects of Other Factors

Effects of Retriever . We compare DPR with three other retrievers: 1) *TF-IDF* (Wu et al., 2008) measures the relevance between a query and a paragraph by the weighted sum of overlapped words between the query and the paragraph. 2) *None-RTV* use only the query as input to the generator; 3) *Oracle-RTV* uses the gold supporting paragraphs in oracle Wikihow article as supporting paragraphs; For each question, We rank all the supporting paragraphs in test set. We demonstrate the retrievers' performance in Table 7. We present in Table 8 the ROUGE and METEOR scores when using different retrievers to collect supporting paragraphs. For both architectures, *Oracle-RTV* outperforms *TF-IDF* and *DPR* by large margin, *TF-IDF* and *DRP* also outperforms None-RTV significantly. However, even *DPR* achieves much better performance than *TF-IDF* in retrieving evaluation, we do not notice prominent difference in answer generation. This implies the quality of supporting paragraphs has limited effects on generation results.

Retriever	Hits@10	MRR
TF-IDF	39.18	69.11
DPR	**54.67**	**87.76**
Oracle-RTV	100.00	100.00

Table 7: Performance of retrievers.

Effects of supporting paragraphs corpora. According to Figure 4, We do not observe explicit difference when using *Wikihow-train* and *Wikipedia* as corpus. However, we note that compared to Wikipedia, Wikihow-train is much smaller in volume. Thus in practice, Wikihow-train is a better choice for corpus as it takes less retrieving time.

Retriever	R-1	R-2	R-L	METEOR
Arc-Direct - Answer Generation				
TF-IDF	26.4	6.41	25.53	17.15
DPR	26.13	6.37	25.54	16.64
None-RTV	17.65	4.53	18.52	8.02
Oracle-RTV	41.92	16.51	39/63	27.61
Arc-Planning - Answer Generation				
TF-IDF	27.89	6.71	25.02	17.3
DPR	28.3	7.11	25.63	17.47
None-RTV	21.11	3.97	19.33	13.2
Oracle-RTV	46.45	18.94	43.05	32.32
Arc-Planning - Subtopic Planning				
TF-IDF	23.0	7.89	23.79	17.13
DPR	23.04	7.8	23.57	16.59
None-RTV	16.36	4.68	17.17	11.39
Oracle-RTV	27.13	10.75	28.23	19.24

Table 8: Performance of answer generation when using different retrievers. Even though *DPR* outperforms *TF-IDF* by over 10 points in both Hit10 and MRR (Table7), their answer generation results are not prominently different.

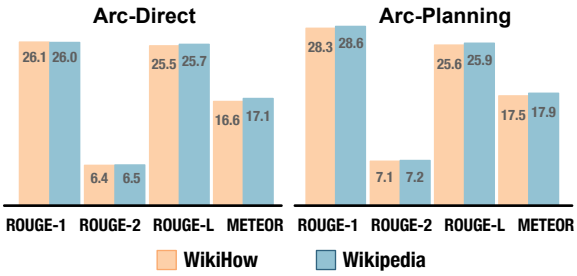

Figure 4: Performance of answer generation when using different supporting paragraph corpora. We find that the difference is not prominent.

4 Related Works

Question Answering Datasets Recent years has witnessed great advances in open domain question answering. However, most question answering datasets (Yang et al., 2018; Joshi et al., 2017; Rajpurkar et al., 2016; Dunn et al., 2017) are designed for factoid questions, i.e., questions that start with 'what', 'when', 'who', 'where', 'which', etc. and they require only extractive answers, i.e., the answers are spans of source text.

There exist several non-factoid question answering datasets (Dos Santos et al., 2015; Yang et al., 2015; Cohen et al., 2018; Nakov et al., 2017; Hashemi et al., 2019; Fan et al., 2019). Most of these datasets are proposed for the purpose of passage retrieval and re-ranking, i.e., the proposed tasks are ranking the provided evidence passages according to their relevance to a given question. No answer generation is involved. An exception is (Fan et al., 2019), which focus on non-factoid questions, i.e. questions start with "why," "how," and long-form answer generation, i.e. the answer is an elaborate and in-depth passage. Our work differs from ELI5 with a specific emphasis on *how-to* questions, and subtopic structures of their answers.

Question Answering Models As open domain question answering is a knowledge intensive task, most state of art models (Chen et al., 2017; Guu et al., 2020; Lewis et al., 2020b; Izacard and Grave, 2020; Asai et al., 2021) apply a "retrieve-generate" paradigm, where the retriever collects supporting paragraphs related to the question from a large external corpus, the generator then generates an answer based on the question and the supporting paragraphs. Compared to previous works which apply the "retrieve-generate" paradigm, we use the retriever not only to retrieve supporting paragraphs, but also to evaluate and select subtopics.

Planning Based Text Generation Content planning is widely used to improve diversity and coherence in various text generation tasks including story generation (Yao et al., 2019; Goldfarb-Tarrant et al., 2019), argument generation (Hua et al., 2019b), Wikipedia article generation (Hua and Wang, 2019) and abstractive summarization (Liu et al., 2015; Huang et al., 2020) etc. While most research studies resort to key words or knowledge graph entities for content planning, we use subtopics for planning our answers.

Procedural Text Our work is also related to procedural text understanding such as recipes (Tandon et al., 2020; Rajagopal et al., 2020; Du et al., 2019; Dalvi et al., 2019; Tandon et al., 2019; Chu et al., 2017). However, instead of tracking state changes in procedure text, we focus on generating subtopics to improve the coherence of answers.

Wikihow In previous researches, Wikihow data (Koupaee and Wang, 2018) was mostly used in summarization (Zhang et al., 2019; Kryscinski et al., 2019) or step reasoning (Zhang et al., 2020), The task we propose aims at different goals, i.e, decomposing a question into subtopics and generating answer based on the question and subtopics.

5 Conclusion and Future Work

We proposed a novel subtopic planning based architecture for answering *How-to* questions. Our architecture is able to generate answers with better structure, higher diversity and more consistent

quality. Moreover, our subtopic selection method effectively singles out high quality subtopics with relevance and independence. Both automatic and human evaluation proved the effectiveness of our methods. We consider the two directions for future research: 1) Improving the answer's quality by applying end-to-end retrieval-generation models, e.g. (Lewis et al., 2020b). 2) Developing precise metrics to evaluate long-form and non-factoid answers.

Acknowledgement

We are grateful to the reviewers for their insightful comments. Fei Liu is supported in part by National Science Foundation grant IIS-2303678. Hong Yu is supported in part by UMass Lowell startup fund.

References

Akari Asai, Jungo Kasai, Jonathan H. Clark, Kenton Lee, Eunsol Choi, and Hannaneh Hajishirzi. 2021. XOR QA: Cross-lingual open-retrieval question answering. In *Proceedings of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Advances in neural information processing systems*, pages 2787–2795.

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 (Volume 1: Long Papers)*, pages 1870–1879, Vancouver, Canada. Association for Computational Linguistics.

Cuong Xuan Chu, Niket Tandon, and Gerhard Weikum. 2017. Distilling task knowledge from how-to communities. In *Proceedings of the 26th International Conference on World Wide Web*, pages 805–814.

Daniel Cohen, Liu Yang, and W Bruce Croft. 2018. Wikipassageqa: A benchmark collection for research on non-factoid answer passage retrieval. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 1165–1168.

Bhavana Dalvi, Niket Tandon, Antoine Bosselut, Wen-tau Yih, and Peter Clark. 2019. Everything happens for a reason: Discovering the purpose of actions in procedural text. 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)*, pages 4496–4505, Hong Kong, China. Association for Computational Linguistics.

Cicero Dos Santos, Luciano Barbosa, Dasha Bogdanova, and Bianca Zadrozny. 2015. Learning hybrid representations to retrieve semantically equivalent questions. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 694–699.

Xinya Du, Bhavana Dalvi, Niket Tandon, Antoine Bosselut, Wen-tau Yih, Peter Clark, and Claire Cardie. 2019. Be consistent! improving procedural text comprehension using label consistency. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2347–2356, Minneapolis, Minnesota. Association for Computational Linguistics.

Matthew Dunn, Levent Sagun, Mike Higgins, V Ugur Guney, Volkan Cirik, and Kyunghyun Cho. 2017. Searchqa: A new q&a dataset augmented with context from a search engine. *arXiv preprint arXiv:1704.05179*.

Angela Fan, Yacine Jernite, Ethan Perez, David Grangier, Jason Weston, and Michael Auli. 2019. ELI5: Long form question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3558–3567, Florence, Italy. Association for Computational Linguistics.

Seraphina Goldfarb-Tarrant, Haining Feng, and Nanyun Peng. 2019. Plan, write, and revise: an interactive system for open-domain story generation.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. Realm: Retrieval-augmented language model pre-training. *arXiv preprint arXiv:2002.08909*.

Helia Hashemi, Mohammad Aliannejadi, Hamed Zamani, and W Bruce Croft. 2019. Antique: A non-factoid question answering benchmark. *arXiv preprint arXiv:1905.08957*.

Tianxing He, Jingzhao Zhang, Zhiming Zhou, and James Glass. 2019. Quantifying exposure bias for neural language generation. *arXiv preprint arXiv:1905.10617*.

Marti A Hearst and Jan O Pedersen. 1996. Reexamining the cluster hypothesis: scatter/gather on retrieval results. In *Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 76–84.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. In *International Conference on Learning Representations*.

Ari Holtzman, Jan Buys, Maxwell Forbes, Antoine Bosselut, David Golub, and Yejin Choi. 2018. Learning to write with cooperative discriminators. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1638–1649, Melbourne, Australia. Association for Computational Linguistics.

Xinyu Hua, Zhe Hu, and Lu Wang. 2019a. Argument generation with retrieval, planning, and realization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2661–2672, Florence, Italy. Association for Computational Linguistics.

Xinyu Hua, Zhe Hu, and Lu Wang. 2019b. Argument generation with retrieval, planning, and realization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2661–2672, Florence, Italy. Association for Computational Linguistics.

Xinyu Hua and Lu Wang. 2019. Sentence-level content planning and style specification for neural text 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)*, pages 591–602, Hong Kong, China. Association for Computational Linguistics.

Luyang Huang, Lingfei Wu, and Lu Wang. 2020. Knowledge graph-augmented abstractive summarization with semantic-driven cloze reward. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5094–5107, Online. Association for Computational Linguistics.

Daphne Ippolito, Reno Kriz, João Sedoc, Maria Kustikova, and Chris Callison-Burch. 2019. Comparison of diverse decoding methods from conditional language models. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3752–3762, Florence, Italy. Association for Computational Linguistics.

Gautier Izacard and Edouard Grave. 2020. Leveraging passage retrieval with generative models for open domain question answering. *arXiv preprint arXiv:2007.01282*.

Mandar Joshi, Eunsol Choi, Daniel 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 (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada. Association for Computational Linguistics.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick 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)*, pages 6769–6781, Online. Association for Computational Linguistics.

Mahnaz Koupaee and William Yang Wang. 2018. Wikihow: A large scale text summarization dataset. *arXiv preprint arXiv:1810.09305*.

Wojciech Kryscinski, Nitish Shirish Keskar, Bryan McCann, Caiming Xiong, and Richard Socher. 2019. Neural text summarization: A critical evaluation. 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)*, pages 540–551, Hong Kong, China. 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*, pages 7871–7880, Online. Association for Computational Linguistics.

Patrick Lewis, Ethan Perez, Aleksandara Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020b. Retrieval-augmented generation for knowledge-intensive nlp tasks. *arXiv preprint arXiv:2005.11401*.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016. A diversity-promoting objective function for neural conversation models. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 110–119, San Diego, California. Association for Computational Linguistics.

Fei Liu, Jeffrey Flanigan, Sam Thomson, Norman Sadeh, and Noah A. Smith. 2015. Toward abstractive summarization using semantic representations. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1077–1086, Denver, Colorado. Association for Computational Linguistics.

Preslav Nakov, Doris Hoogeveen, Lluís Màrquez, Alessandro Moschitti, Hamdy Mubarak, Timothy Baldwin, and Karin Verspoor. 2017. SemEval-2017 task 3: Community question answering. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 27–48, Vancouver, Canada. Association for Computational Linguistics.

Kyosuke Nishida, Itsumi Saito, Kosuke Nishida, Kazutoshi Shinoda, Atsushi Otsuka, Hisako Asano, and Junji Tomita. 2019. Multi-style generative reading comprehension. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2273–2284, Florence, Italy. Association for Computational Linguistics.

Dragomir R Radev, Hong Qi, Harris Wu, and Weiguo Fan. 2002. Evaluating web-based question answering systems. In *LREC*.

Dheeraj Rajagopal, Niket Tandon, Peter Clark, Bhavana Dalvi, and Eduard Hovy. 2020. What-if I ask you to explain: Explaining the effects of perturbations in procedural text. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3345–3355, Online. Association for Computational Linguistics.

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*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Chuanqi Tan, Furu Wei, Nan Yang, Bowen Du, Weifeng Lv, and Ming Zhou. 2018. S-net: From answer extraction to answer synthesis for machine reading comprehension. In *AAAI*.

Niket Tandon, Bhavana Dalvi, Keisuke Sakaguchi, Peter Clark, and Antoine Bosselut. 2019. WIQA: A dataset for "what if..." reasoning over procedural text. 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)*, pages 6076–6085, Hong Kong, China. Association for Computational Linguistics.

Niket Tandon, Keisuke Sakaguchi, Bhavana Dalvi, Dheeraj Rajagopal, Peter Clark, Michal Guerquin, Kyle Richardson, and Eduard Hovy. 2020. A dataset for tracking entities in open domain procedural text. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6408–6417, Online. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Ho Chung Wu, Robert Wing Pong Luk, Kam Fai Wong, and Kui Lam Kwok. 2008. Interpreting tf-idf term weights as making relevance decisions. *ACM Transactions on Information Systems (TOIS)*, 26(3):1–37.

Yi Yang, Wen-tau Yih, and Christopher Meek. 2015. Wikiqa: A challenge dataset for open-domain question answering. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 2013–2018.

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.

Lili Yao, Nanyun Peng, Ralph Weischedel, Kevin Knight, Dongyan Zhao, and Rui Yan. 2019. Plan-and-write: Towards better automatic storytelling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 7378–7385.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter J. Liu. 2019. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization.

Li Zhang, Qing Lyu, and Chris Callison-Burch. 2020. Reasoning about goals, steps, and temporal ordering with WikiHow. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4630–4639, Online. Association for Computational Linguistics.

A Appendices

A.1 Example Subtopic Sets

We observe that some system-generated subtopic sets are flawed, but a large portion of the subtopic sets are of good quality. Some of them are close to human-written subtopics. Examples are provided in Table 9 (examples 1–6). Moreover, we come up with imaginary questions that are quite different from those observed in Wikihow (examples 5–6) and we find the model is able to produce reasonable answers for those questions.

Irrelevant subtopics generated by our model tend to have the following characteristics:

1. **A later generated subtopic in the subtopic set tends to be less relevant**: Note that we generate a set of subtopics in a seq2seq manner, with the question and supporting documents as the source sequence, and the concatenation of all subtopics as the target sequence. Due to exposure bias (He et al., 2019), subtopic generated in the later part of the sequence are more likely to be off-topic. Examples of this kind are included in Table 9 (examples 7–10). While the subtopics in the front part are all related to the question, the later generated subtopics, marked in red, tend to deviate from the question.

2. **Commonsense mistakes**: We observe that when the question is about a problem that requires commonsense, the model tends to generate unrelated or erroneous subtopics. E.g. in Table 9 example 9, the location services have nothing to do with facebook like notification.

It suggests that incorporating commonsense knowledge may improve the performance of a subtopic decomposition model.

Overlapped subtopics generated by our model tend to have the following characteristics:

- Thanks to a repetition penalty, the generated subtopics of the same subtopic set are usually different from each other. I.e., there are only very few cases where multiple subtopics of the same subtopic set are identical. However, there is no guarantee that those generated subtopics are semantically independent. E.g., in Table 9 example 11–12, the orange subtopics have semantic overlap. According to our experimental results, the vector-space based metrics could effectively identify those semantically overlapped subtopics.

A.2 Example Long-Form Answers

Table 10 shows example answers generated by **Arc-D** and **Arc-P** models, respectively. Notable findings include the following observations:

1. Subtopics make a long answer more organized. The answers generated by **Arc-P** are split into multiple paragraphs, each paragraph focuses on a subtopic. In contrast, answers generated by **Arc-D** are quite general and sometimes disordered. Additionally, answers generated by **Arc-P** are easier to read than those of **Arc-D**, as the model leverages subtopics to generate structured answers.

2. As the target sequence gets longer, the quality of the generated sequence tends to deteriorate due to exposure bias. This is demonstrated in Table 10 example 1, where the model repeatedly generates the red part in the later stage of the generation process. Instead of generating a long paragraph as the answer, **Arc-P** generates several shorter answer paragraphs, thus reducing the risk of exposure bias.

A.3 Implementation and Hyperparameters

Both *Arc-D* and *Arc-P* are based on a retrieval-composing paradigm, which consists of a retrieval module (DPR) and a composing module (BART). We discuss the model implementation details and hyperparameters below.

A.3.1 Retrieval module (DPR) key parameters and implementation details

- Batch size: 80 (which means one positive instance corresponds to 79 negative, including 1 hard negative instance);
- Max sequence length: 256;
- Training epochs: 12 (during testing, we use the model that gives the best validation loss);
- The other hyper-parameters are based on the default parameter of DPR: `https://github.com/facebookresearch/DPR`
- DPR is trained on 4 M40 GPUs;
- Given a question, we search for a similar paragraph that has high TF-IDF similarity with the question, but is not among the question's gold supporting paragraphs as a **hard negative instance**;
- DPR is pretrained on *fairseq/roberta-base*;
- When measuring *relevance* and *independence*, we choose the top $M = 500$ paragraphs whose DPR embeddings are closest to $E_X(x)$.

A.3.2 Composing module (BART) key parameters and implementation details

- BART-large pre-trained on *yjernite/bart_eli5*;
- Max number of tokens input: 1024 (for *Arc-D*), 512 (For *Arc-P*);
- Max number of tokens output: 384 (for *Arc-D*), 128 (For *Arc-P*);
- Max number of training epochs: 9 (during testing, we use the model that gives the best validation loss);
- Learning rate: 3e-5;
- The other hyper-parameters are based on the default parameters of huggingface's BART model: `https://huggingface.co/transformers/model_doc/bart.html`
- For each question, we use *top-k* sampling to generate 10 different subtopic sets.
- The BART is trained on a single M40 GPU, the average training time is up to 4-5 days.

ID		Question & Subtopics
Fine Subtopic Sets Produced by the Model		
1	**Q**	How to be glamorous?
	G	1.Caring for your body; 2. Applying makeup; 3.Dressing glamorous
	H	1.Looking glamorous; 2.Dressing glamorous; 3.Acting glamorous
2	**Q**	How to maintain a relationship?
	G	1.Communicating your needs; 2.Keeping the romance alive; 3.Avoiding conflict
	H	1.Love them for who they are; 2.Be a good listener; 3.Be nice
3	**Q**	How to lose extreme weight?
	G	1.Eating the right foods; 2.Exercising the right way; 3.Managing emotional and mental health
	H	1.Setting a weight loss goal; 2.Designing a weight loss diet; 3.Using exercise to support weight loss; 4.Staying motivated
4	**Q**	How to coach youth basketball?
	G	1.Finding players; 2.Preparing for practice; 3.Developing effective coaching skills
	H	1.Preparing yourself; 2.Meeting your players; 3.Putting your knowledge to work
5	**Q**	How to train a dragon
	G	1.Training your dragon; 2.Mastering the dragon's flight
6	**Q**	How to master Shinobi kill
	G	1.Mastering the technique; 2. Mastering the art of killing
Flawed Subtopic Sets Produced by the Model		
7	**Q**	How to fold jeans?
	G	1. Making a triangle fold; 2.Making a rectangle fold; 3.Folding a t-shirt
	H	1. Folding jeans; 2.Storing folded jeans
8	**Q**	How to coach youth basketball?
	G	1. Becoming a good coach; 2.Developing your skills; 3.Finding a job
	H	1.Preparing yourself; 2.Meeting your players; 3.Putting your knowledge to work
9	**Q**	How to turn off facebook like notifications on iphone?
	G	1. Disabling facebook like notifications; 2. Disabling location services
	H	1. disabling notifications for a single post; 2.Disabling all facebook notifications
10	**Q**	How to switch keyboard input languages in windows 8?
	G	1. Switching keyboard input language; 2. Using the task manager
	H	1. Switching through the settings charm; 2.Switching through the taskbar on the desktop; 3. Switching through keyboard shortcuts
11	**Q**	How to maintain a relationship?
	G	1. Maintaining a healthy relationship; 2. Keeping the relationship strong; 3. Avoiding toxic relationships
	H	1.Love them for who they are; 2.Be a good listener; 3.Be nice
12	**Q**	How to start a business in New York city
	G	1. Developing your business plan; 2.Creating a marketing plan; 3. ...
	H	1.Using city services to get your business started 2.Determining your market

Table 9: Examples of system-generated subtopics. **Q** refers to a question, **G** refers to a subtopic set generated by our model, **H** refers to the subtopic set provided by human editors from Wikihow. The red subtopics are unrelated to the question, the orange subtopics have semantic overlap among themselves, the blue questions are imaginative questions made-up by the authors.

ID	Question & Answers	
1	**Q**	**How to Give Yourself Insulin?**
	Arc-D	Determine if you need insulin. Prepare the insulin. Place the syringe in your arm. Use a needle for the injection. Inject the insulin into your skin. Do not remove the needle. Check for bleeding. Remove the needle. Prepare the insulin. Place the syringe in your arm. ... (repeated generation)
	Arc-P	**1.Getting Ready to Give Yourself Insulin:** Prepare the insulin for injection. Check your bloodsugar. Know when to give yourself insulin. Find a good needle and syringe. Clean your syringe thoroughly . Decide where to give yourself insulin. **2.Administering the Insulin Injection:** Wash your hands thoroughly with soap and water. Determine the injection site. Position the needle at a 45-degree angle to the skin. Inject the insulin into the skin. Wait for the insulin to release. Remove the needle from the skin. **3.Knowing when to Avoid Insulin Injections:** Know the risks of insulin injections. Recognize the symptoms of diabetes. Watch for signs of complications. See your doctor.
2	**Q**	**How to Not Worry About Something?**
	Arc-D	Think about what you're worried about. Make a worry list. Put your worries into perspective. Take some time to be in the moment. Practice mindfulness. Talk to someone. Write down your worries. Do something you enjoy. Get enough sleep. Focus on the positive. Avoid rumination. Recognize when you need professional help. See a therapist. Join a support group.
	Arc-P	**1.Facing Your Worries:** Practice mindfulness. Accept uncertainty. Set realistic goals. Don't compare yourself to others. **2.Making Time for Relaxation:** Make time for exercise. Get enough sleep. Don't procrastinate.Set aside time to think about the future **3.Taking Care of Yourself:** Get enough sleep. Eat a healthy diet. Exercise regularly. Find ways to relax. Take care of your body.
3	**Q**	**How to Get Your Band Noticed?**
	Arc-D	Make a website. Create a Facebook page. Create a Twitter account. Create an Instagram account. Create a YouTube account. Create a blog. Play gigs. Find a manager. Contact music labels. Perform locally. Get your music out there. Write songs. Record your music. Publish your music. Distribute your music.
	Arc-P	**1.Developing Your Band's Music:** Record your songs. Make a demo tape. Put your music on YouTube. Start a blog. Get an agent. **2.Promoting Your Band Online:** Promote your band on social media sites. Create a blog. Start a YouTube account. Get listed in music publications. **3.Promoting Your Band in Person:** Play in front of friends and family. Go to local shows. Promote your music on social media. Make a scrapbook.
4	**Q**	**How to Buy a Racehorse?**
	Arc-D	Determine what you want in a horse. Decide how much money you can afford to spend on a horse. Consider your finances. Find a good breeder. Research the condition of the horse. Check the horse's pedigree. Assess the horse's temperament. Look at the horse's past. Ask about the horse's health. Make sure that you have enough money before you buy the horse. Buy the horse from a reputable breeder. Purchase the horse from a trustworthy seller. Do not buy a horse from a pet store.
	Arc-P	**1.Deciding to Buy a Racehorse:** Consider the horse's pedigree. Think about the horse's age. Consider the horse's breed. Consider the horse's physical condition. Consider the horse's temperament. **2.Buying a Racehorse in Person:** Decide what type of horse you want to buy. Visit several barns and showmen. Make an appointment with the salesperson. **3.Buying a Racehorse Online:** Find a horse to buy. Search for the horse online. Purchase your horse.

Table 10: Examples of system-generated answers with or without subtopics.

- We balance equation 5 and 6 using a hyper-parameter τ, specifically, the final score of a subtopic set S to a question q is calculated as

$$\text{Score}(S, q) = \tau \text{Relv}(S, q) + (1 - \tau)\text{Indp}(S)$$

We determine τ by heuristically tuning on validation set (MMR). We did so because the quality of high-quality subtopics may not be reflected by automatic coverage-based metrics. During experiment, we set $\tau = 0.3$.

A.4 Evaluation

A.4.1 Automatic Evaluation

We use the python library rouge 1.0.0 to calculate ROUGE: https://pypi.org/project/rouge/; METEOR: https://www.nltk.org/_modules/nltk/translate/meteor_score.html

A.4.2 Human Evaluation

We recruit Amazon Turks to work on two tasks: (a) **1. Answer Preference**, where we ask the Turks to rank three answers to a given question according to their preference; (b) **2. Subtopic Selection**, where we ask our judges to grade subtopic sets from three perspectives: Relevance, Independence and Overall Preference, on the scale of 1-3 points. The grading guidelines are given below:

1. **Relevance**: How close is each subtopic set related to the question; (1-point: One or more subtopic is clearly not relevant to the question; 2-points: One or more subtopics may not relate to the question so well; 3-points: All subtopics are related to the question.)

2. **Independence**: How independent is each subtopic to the other subtopics in the same subtopic set; (1-point: There is a clear meaning overlap (or repetition) between a few subtopics; 2-points: There is a weak meaning overlap between a few subtopics; 3-points: There is no meaning overlap between subtopics.)

3. **Overall Preference** :How do judges like the subtopic set. The grading scale for each perspective is from 1 to 3. (1: A bad set of subtopics for the question; 2: An acceptable but not good subtopics for the question; 3: A good subtopics for the question)

Answer preference	47%
Subtopic-Relevance	36%
Subtopic-Independence	45%
Subtopic-Overall	39%

Table 11: Inter-annotator agreement of human evaluations

We present the screenshots of our AMTurk pages in Figure 5 and 6. We measure inter-annotator agreement using Cohen's Kappa (k), the results are presented in Table 11.

A.5 Dataset

We re-purpose the WikihowSum datasetfor our task. The source code for processing the data has been included in our project repository.

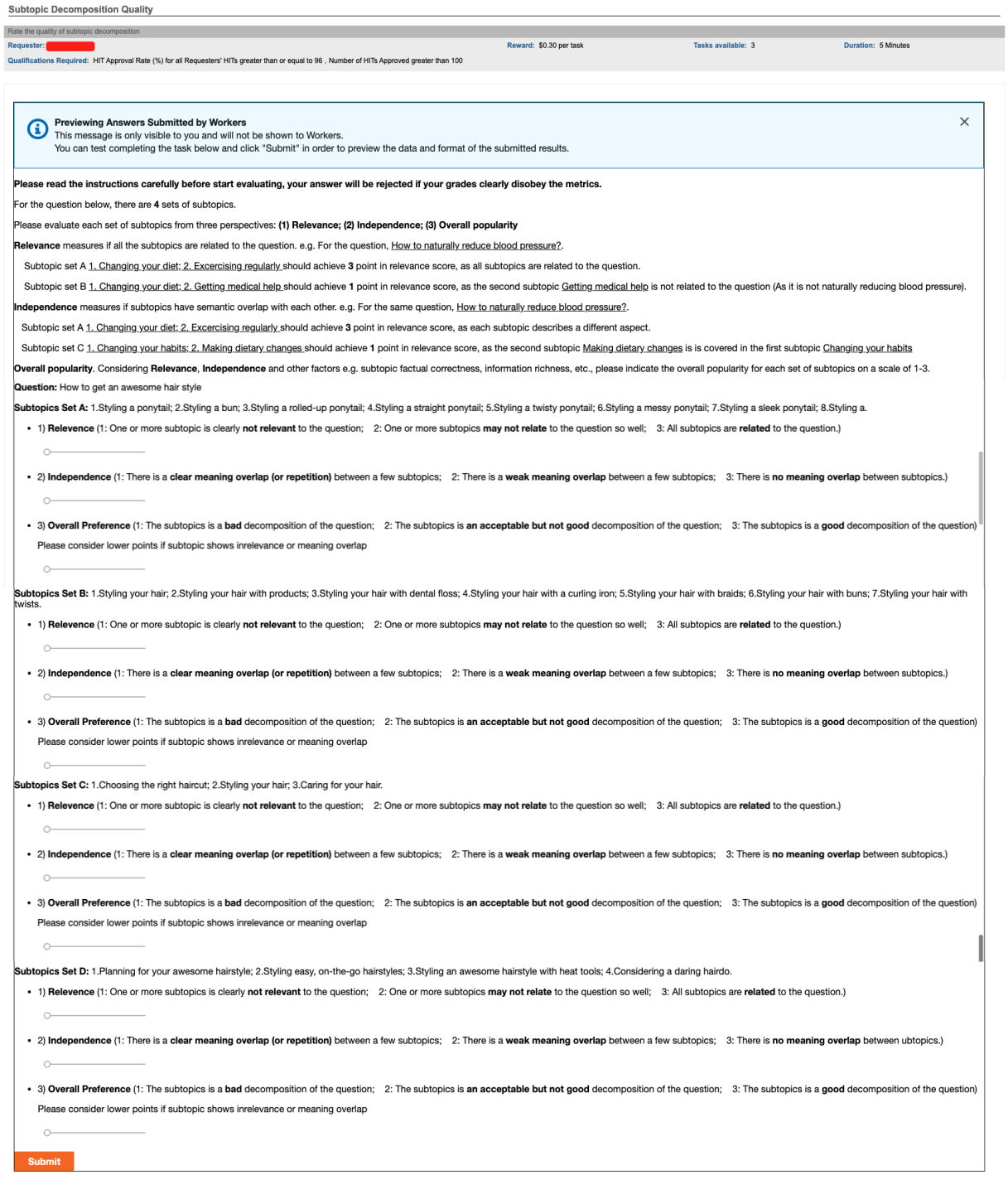

Figure 5: A screenshot of subtopic evaluation performed by mechanical turkers.

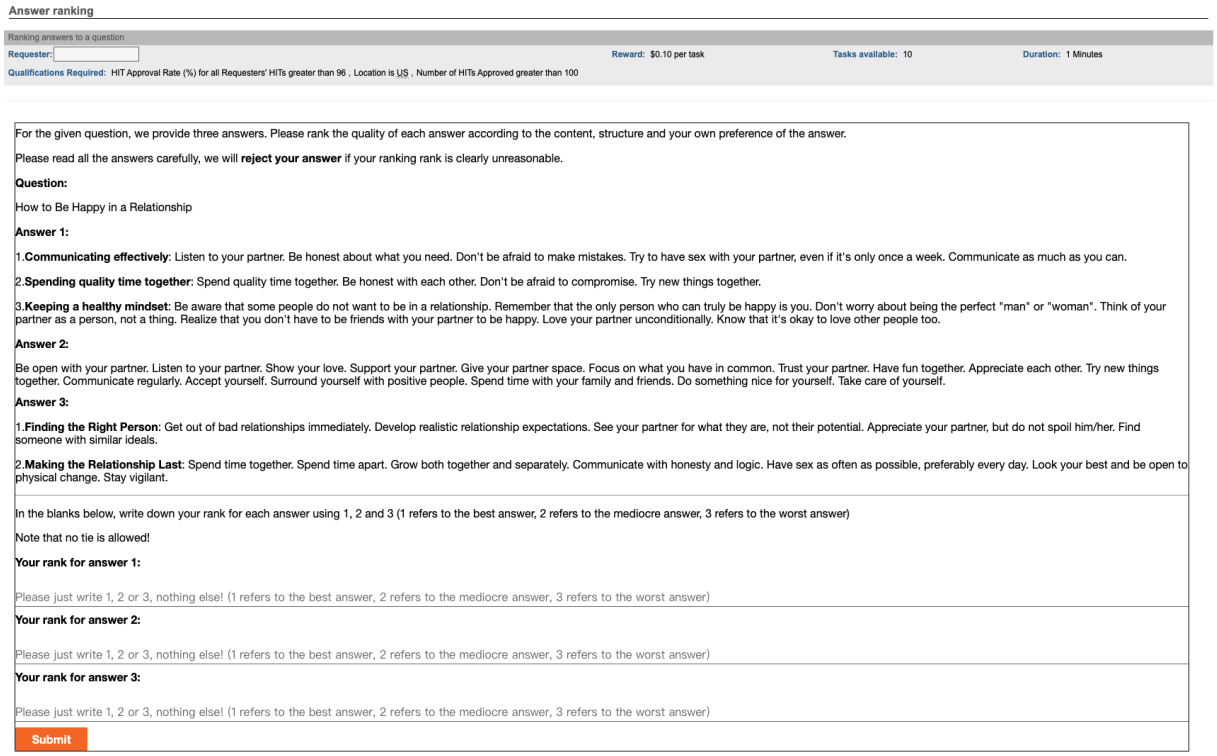

Figure 6: A screenshot of answer ranking evaluation performed by mechanical turkers.