CMQA: A Dataset of Conditional Question Answering with Multiple-Span Answers

Yiming Ju[*,1,2], **Weikang Wang**[*,3], **Yuanzhe Zhang**[1,2],
Suncong Zheng[3], **Kang Liu**[1,2], **Jun Zhao**[1,2],
[1] National Laboratory of Pattern Recognition, Institute of Automation, CAS, Beijing, China
[2] School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing, China
[3] Machine Learning Platform Department, Tencent
{yiming.ju, yzzhang, kliu, jzhao}@nlpr.ia.ac.cn
{daxianwang, congzheng}@tencent.com

Abstract

Forcing the answer of the Question Answering (QA) task to be a single text span might be restrictive since the answer can be multiple spans in the context. Moreover, we found that multi-span answers often appear with two characteristics when building the QA system for a real-world application. First, multi-span answers might be caused by users lacking domain knowledge and asking ambiguous questions, which makes the question need to be answered with conditions. Second, there might be hierarchical relations among multiple answer spans. Some recent span-extraction QA datasets include multi-span samples, but they only contain unconditional and parallel answers, which cannot be used to tackle this problem. To bridge the gap, we propose a new task: conditional question answering with hierarchical multi-span answers, where both the hierarchical relations and the conditions need to be extracted. Correspondingly, we introduce CMQA, a **C**onditional **M**ultiple-span Chinese **Q**uestion **A**nswering dataset to study the new proposed task. The final release of CMQA consists of 7,861 QA pairs and 113,089 labels, where all samples contain multi-span answers, 50.4% of samples are conditional, and 56.6% of samples are hierarchical. CMQA can serve as a benchmark to study the new proposed task and help study building QA systems for real-world applications. The low performance of models drawn from related literature shows that the new proposed task is challenging for the community to solve. CMQA can be accessed at `https://github.com/juyiming/CMQA`.

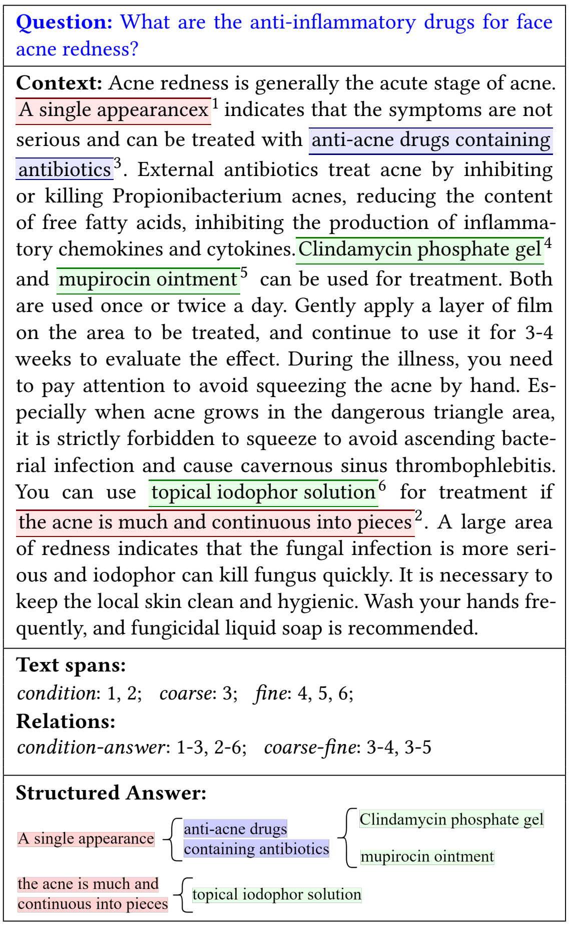
1 Introduction

Question answering (QA) is a challenging benchmark task, which can drive the development of natural language understanding (NLU) methods and has significant utility to users (Kwiatkowski et al., 2019). This research area has made significant progress with many sizable datasets and standard benchmarks. Notably, the span-extraction task is one of the most studied subtasks because of its wide applicability and easy evaluation characteristics.

Most existing span-extraction QA datasets (Rajpurkar et al., 2016; Yang et al., 2018; Choi et al., 2018; Kwiatkowski et al., 2019; Chen et al., 2020) only contain single-span samples, where the answer is a single text span in the context. However, limiting the answer to be a single span in the context might be restrictive since the answer of some questions can be multiple text spans in real-world QA scenarios. Take the case in Figure 1 for example, given the question asking about *'drugs for face acne redness'*, the corresponding drugs (green and blue highlighted text spans) appear in different parts of the context. A single span including all these drugs is extremely long, thus it is more suitable to use multi-span answers to answer this question. Moreover, when building the QA system for a real-world application, we found that the multi-span answers often appear with two characteristics:

First, multi-span answers can be caused by users lacking related domain knowledge and asking ambiguous questions. In such cases, questions might need multiple conditions to specify the circumstance. For example, in Figure 1, the user asked a brief question about *'drugs for acne redness'*. Since different drugs are needed according to the severity of symptoms:(*'A single appearance'* and *'the acne is much and continuous into pieces'*), the question is answered separately. It is misleading to give all answer spans without distinguishing the conditions. Moreover, answers of different conditions might be contradictory in some samples. For example, a user asked about the battery capacity of the iPhone. Since the battery capacity of the iPhone is different according to models, the multi-span an-

[*] means both authors contributed equally to this research.



Question: What are the anti-inflammatory drugs for face acne redness?

Context: Acne redness is generally the acute stage of acne. A single appearancex[1] indicates that the symptoms are not serious and can be treated with anti-acne drugs containing antibiotics[3]. External antibiotics treat acne by inhibiting or killing Propionibacterium acnes, reducing the content of free fatty acids, inhibiting the production of inflammatory chemokines and cytokines. Clindamycin phosphate gel[4] and mupirocin ointment[5] can be used for treatment. Both are used once or twice a day. Gently apply a layer of film on the area to be treated, and continue to use it for 3-4 weeks to evaluate the effect. During the illness, you need to pay attention to avoid squeezing the acne by hand. Especially when acne grows in the dangerous triangle area, it is strictly forbidden to squeeze to avoid ascending bacterial infection and cause cavernous sinus thrombophlebitis. You can use topical iodophor solution[6] for treatment if the acne is much and continuous into pieces[2]. A large area of redness indicates that the fungal infection is more serious and iodophor can kill fungus quickly. It is necessary to keep the local skin clean and hygienic. Wash your hands frequently, and fungicidal liquid soap is recommended.

Text spans:
condition: 1, 2; *coarse*: 3; *fine*: 4, 5, 6;

Relations:
condition-answer: 1-3, 2-6; *coarse-fine*: 3-4, 3-5

Structured Answer:

- A single appearance
  - anti-acne drugs containing antibiotics
    - Clindamycin phosphate gel
    - mupirocin ointment
- the acne is much and continuous into pieces
  - topical iodophor solution

Figure 1: An example of labels in CMQA. Text spans: *condition* (red), *coarse* (blue) and *fine* (green). Relations: *condition-answer* and *coarse-fine*. The example is translated from Chinese.

swers will contradict each other without specifying the condition.

Second, multiple answers in a sample might belong to different granularities, and there are often hierarchical relations among them. For example, in Figure 1, the answer span *'topical anti-acne drugs containing antibiotics'* is a type of drug, while *'Clindamycin phosphate gel'* and *'mupirocin ointment'* are two specific drug names of this type. If we only give all these answer spans in parallel, the user cannot get the granularity and hierarchical information of these drug names. In this case, we need to provide the answer granularity and the hierarchical relations for accuracy.

The analysis results in Table 4 show that about half of the multi-span samples in our data are conditional, and half of them are multi-granularity. The high proportion of these samples demonstrates that the summarized two characteristics might be widespread in some real-world QA scenarios. Though some recent span-extraction QA datasets include multi-span samples (Dua et al., 2019; Dasigi et al., 2019; Zhu et al., 2020), they only contain unconditional and parallel answers, which cannot be used to tackle this problem. To bridge the gap, we propose a new task: conditional question answering with hierarchical multi-span answers, where both the hierarchical relations and the conditions need to be extracted.

In this paper, we introduce **C**onditional **M**ultiple-span Chinese **Q**uestion **A**nswering dataset (CMQA) to track the new proposed task. Specifically, we pick out samples need to be answered with multiple text spans and use a new label strategy to annotate them. We labeled both answers and conditions if the sample is conditional. Moreover, the answer spans are labeled with different granularity: *coarse* and *fine*. *fine* means the answer is a specific thing, such as a specific time, person, to name a few. *coarse* means the answer span is a general term for a class of things, such as foods containing certain nutrients and people with certain characteristics. The hierarchical relations among answers of different granularities are also provided. As shown in Figure 1, labeled text spans in CMQA consist of three types: *condition*, *coarse* and *fine* (highlighted in the context). Labeled relations consist of two types: *condition-answer* and *coarse-fine*. A structured answer can be easily derived from these labels, which is clear and accurate. Furthermore, such labels are very helpful to reduce the burden of users if they want to read the full context.

The final release of CMQA consists of 7,861 multi-span QA pairs and 113,089 labels, where 50.4% of samples are conditional, and 56.6% are multi-granularity. In addition, we establish models as the baseline of CMQA. Traditional single-span QA models which search for the most likely token as the start/end of the answer are unsuitable for extracting multi-span answers, thus, the span extraction is cast as a sequence tagging problem (Segal et al., 2020). And we use competitive relation extraction methods drawn from related literature (Wu and He, 2019; Zhong and Chen, 2021) for predicting relations. Experimental results show that it is very difficult to extract all spans correctly in a sample. The error analysis shows that the main challenge is to judge whether a span of the correct entity type is an answer. Moreover, though the relation model can achieve high performance on some traditional relation extraction datasets, the performance on our task is extremely poor, which makes giving structured answers like the case in Figure 1 difficult. The poor model performance demonstrates that the new proposed task is challenging for the community to solve.

Our contributions can be summarized as follows:

- We propose a challenge when solving questions from real-world users: conditional question answering with hierarchical multi-span answers.
- We introduce a new annotated Chinese question answering dataset: CMQA, which consists of questions that need to be answered with conditions and hierarchically answer spans.
- We establish models drawn from related literature as the baselines of CMQA. Experimental results show that the new proposed task is challenging.

2 Related Work

Most existing span-extraction QA datasets only contain single-span samples, which might be restrictive in some real-world QA scenarios. An important reason for this gap is the question collection methods when building the datasets. Question collection methods of QA datasets can generally be classified into two categories: creating questions by annotators and collecting questions from real-world users.

2.1 QA datasets with questions created by annotators.

Most span-extraction QA datasets consist of questions created by annotators. Some span-extraction QA datasets' questions are written by annotators who have first read the context containing the answer. For example, **SQuAD** (Rajpurkar et al., 2016) tasks crowdworkers with asking up to 5 questions about each paragraph and highlighting the corresponding answers in the paragraph. **HotpotQA** (Yang et al., 2018) tasks crowdworkers with asking a question about two given paragraphs from different Wikipedia pages and providing the answers. Tasking annotators with questioning and answering at the same time is suboptimal, which might cause questioners to ask questions based on a text span in the context. The asked questions are often simple reformulations of sentences in the context.

Field	Ratio
healthcare	77.6%
education	13.5%
government affair	4.0%
food	1.4%
digital product	0.3%
planting	0.2%
others	2.1%

Table 1: The fields of questions in the QA community.

To avoid this problem and create more natural and challenging questions, **NewsQA** (Trischler et al., 2017) only provides news article's headlines and its summary points to questioners. The full context is unseeable. Similarly, **QuAC** (Choi et al., 2018) also prevents the questioner from seeing the full context. Though effort has been made, there are inevitably differences between the human-created questions and the questions asked by users in real-world scenarios. The concerns are as follows: First, the user's question is often not based on a certain context in real-world QA scenarios. Second, crowdworkers might already know the form of the answer (single text span), which hints that they should ask questions with a single text span answer. Last, the problems created are often high quality, while a real-world question might sometimes be ambiguous.

2.2 QA datasets with questions collected from real-world users.

Besides manually created questions, some QA datasets use questions collected from queries from search engines. The answer form of these datasets is usually free text instead of text spans. For example, **MS Marco** (Nguyen et al., 2016) contains queries sampled from the Bing search engine. **DuReader** (He et al., 2018), which contains queries from Baidu search logs, chooses the free-text answer form, both use the free-text answer form and BLEU (Papineni et al., 2002) score as the evaluation metric.

Though there are datasets with questions from search engines that use text span as the answer form, the span length is usually longer than phrase level. Moreover, a screening mechanism is often used to filter questions. For example, **WIKIQA** (Yang et al., 2015) uses a single sentence as the answer form, and questions that cannot be answered with a single sentence are abandoned during the construction. **Natural Questions** (Kwiatkowski et al., 2019) filter out questions cannot be answered with an entity or explanation. Moreover, Natural Questions provide long answers (paragraphs, tables, list items, or whole lists) besides short answers. A question might have a long answer but no short answer. These datasets show that a single span, especially a phrase-level text span, is not enough to answer real-world questions.

3 Dataset Construction

In this section, we describe the data construction process of CMQA.

3.1 Data Source

Samples in CMQA are collected from an authoritative Chinese QA community, where experts in related fields provide answers to questions from users. The community covers a lot of fields, as shown in Table 1, in which the healthcare field has received the most questions from users.

Since the provided answers often contain much secondary information such as descriptions and supplements, such as the example in Figure 1, the length of some answers is rather long (64.7% are longer than 100 words). The long answer texts are inconvenient to users who want to browse quickly and get information. Thus, we want to provide a concise answer (short text span) besides the full text. However, due to the existence of multi-span samples and the two characteristics summarized in Section 1, many questions are not suitable to be answered with a single text span. Thus, we let the annotators pick out samples that cannot be answered with a single text span to tackle the problem. We get 8,864 filtered samples from 25,000 samples and build our dataset based on these samples. Note that we have discard samples that may reveal the personal information of the questioner or containing offensive content during this phase.

3.2 Annotation Scheme

The annotation process is realized by three experts and three full-time annotators. As shown in Figure 2, the annotation scheme consists of 4 steps. Experts first formulate the annotation guidelines after pilot annotation. Then we train the annotators until all annotators think they can work independently. Then, we conduct an inter-annotator agreement (IAA) study to examine the annotation quality. We decide whether to continue the training phase according to the IAA score. In the last phase, we assign different parts of the samples to annotators for annotation.

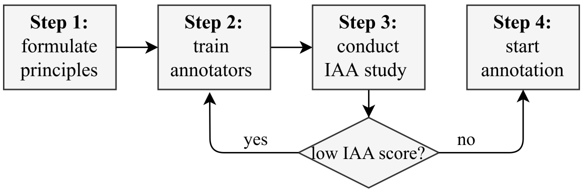


Figure 2: Pipeline of the annotation scheme.

Annotation Principles The main principles of annotation are summarized as follows:

- The text span should be as concise as possible.
- There is a multi-level hierarchical relationship, only the last level is marked
- The sample can be abandoned in the following situations:
  - If the context cannot answer the question.
  - All answers are adjacent and can be regarded as one.
  - Most of the words in the context are part of the answer. It is better to give the entire context as a long answer in this case. (Because the number of the answer spans of each sample is variable, we did not give a specific context length as the threshold to filter out samples. Experts give examples that should be abandoned, and the annotators make their judgments based on these examples.)
  - The question is a combination of multiple sub-questions.

Moreover, when the experts conduct pilot annotation, we find the mismatch in *condition* occurs mostly due to inclusion/exclusion some boundary words. Since these boundaries are difficult to determine sometimes, we decide to expand the boundary of the *condition* to its nearest stop words based on the annotation. In the final release, the average length of *condition* increased from 10.4 to 13.4 after expansion.

We has obtained the liecense of using the data from the community, which will be published with the final release.

Annotation Principles with examples are shown in Appendix, and the full text of the annotation principles will be publish with the dataset.

Dataset	Language	#QA	Conditional	Hierarchical	Question source	Answer amount	Answer length	Answer granularity
QUOREF (Dasigi et al., 2019)	English	2K	-	-	crowdsourcing	2.5	1.6	word/phrase
DROP (Dua et al., 2019)	English	5K	-	-	crowdsourcing	2.5	2.0	word/phrase
MASH-QA (Zhu et al., 2020)	English	29K	-	-	real-world users	4.2	19.3	sentence
CMQA	Chinese	8K	✓	✓	real-world users	7.5	4.5	word/phrase

Table 2: Comparison of CMQA with other QA datasets containing multi-span samples. Questions in QUOREF and DROP are collected by Mechanical Turk while questions in MASH-QA and CMQA are collected from real-world users. *Answer amount* refers to the average number of answer span per sample, and *answer length* refers to the average length per answer span. Note that only multi-span samples are compared.

Label type	IAA score
condition	95.1
coarse	86.1
fine	90.0
condition-answer	98.3
coarse-fine	97.6
all	92.7

Table 3: Inter-annotator agreement scores. ***all*** refers to calculating the micro F1-score of all label types.

Span type			Proportion
condition	*coarse*	*fine*	
✓	✓	✓	22.66%
✓	✓		0.47%
✓		✓	27.30%
	✓	✓	33.93%
	✓		1.40%
		✓	14.24%
50.43%	58.46%	98.13%	

Table 4: The division of CMQA according to text span types.

IAA Study We task annotators with annotating the same samples to take an inter-annotator agreement (IAA) study. We use the F1-score to compute an agreement score between two annotators. Following Gurulingappa et al. (2012); Legrand et al. (2020), we treat one annotator's annotation as the reference and the other's as the prediction. The final agreement score of three annotators is the average of the pair-wise agreement scores, formally as: $S_{abc} = \frac{1}{3}(S_{ab} + S_{ac} + S_{bc})$. We first evaluate the IAA score of different label types separately and then use micro F1-score as the metric to evaluate the IAA score of all labels. The inter-annotator agreement study result of 150 samples is shown in Table 3.

After training and inter-annotator agreement study, we assign different parts of the samples to the annotators for annotation. At last, we get 7,861 samples, in which each sample contains an average of 14.4 labels (8.8 text spans and 5.6 relations). To maximize the reusability of the dataset, we provide a pre-defined split of the dataset into training, development, and test sets in the final release, which consist of 5,861, 1000, and 1000 samples, respectively.

4 Data Analysis

In this section, we analyze the label and question properties of the new dataset. Table 2 shows the comparison of CMQA with other QA datasets containing multi-span samples.

4.1 Label Properties

There are five label types in CMQA: *condition*, *coarse*, *fine*, *condition-answer* and *coarse-fine*. Since the only principle of picking out data to construct the dataset is that the question cannot be answered with a single text span, a sample in our dataset does not necessarily contain all five label types. We investigate the properties of different label types.

Text Span Properties Table 4 shows the division of CMQA according to the included text span types. As shown in Table 4, 50.4% of the samples in CMQA are conditional, and 56.6% are

We choose F1-score instead of other conventional metrics such as the kappa coefficient (Cohen, 1960) because there is no one-to-one correspondence between annotations from different annotators in multiple-span scenarios. Some Named Entity Recognition (NER) datasets (Balasuriya et al., 2009; Srirangam et al., 2019) use kappa coefficient by examining tags of all tokens. However, since most tokens in the context are not parts of the answers in the Question Answering task, such a strategy will lead to a very high agreement score that fails to reflect the annotation quality.

Label type	Amount	Length
condition	1.3	13.4
coarse	1.6	6.1
fine	5.9	4.1
condition-answer	2.8	/
coarse-fine	2.8	/

Table 5: The average amount of labels and the average text span length per sample.

multi-granularity, which shows the necessity and effectiveness of our labeling strategy. Contexts of CMQA have an average of 182.3 words, with 18.8% annotated as conditions and 11.3% annotated as answers. The average amount and length of text spans are shown in Table 5.

Relation Properties Samples containing *condition* also contain related conditional answer spans, which means *condition-answer* is also included. However, containing *condition* doesn't mean all answers in the sample are conditional. There are samples where part of answers are conditional while the others are not, which accounts for 24.8% of *condition*-containing samples. There might be hierarchical relations between answer spans of different granularity. There are 12,458 *coarse* answers in CMQA, and 63.1% of them are connected to *fine* answers. A *coarse* answer can connect to several *fine* answers. The average amount of relations per sample is shown in Table 5.

4.2 Question Properties

Questions in CMQA are collected from users in the QA community. We heuristically identified question types for these questions. Specifically, we first manually observe the questions and summarize some frequently occurring question words. Then we assign question types based on whether these words are included in the question. We visualize the distribution of question types in Figure 3. As shown in Figure 3, the majority of questions in CMQA are about detailed information about specific facts, such as expense, duration, and food.

5 Experiments

This section establishes models using methods drawn from related literature as the baseline of CMQA and analyzes the experiment results.

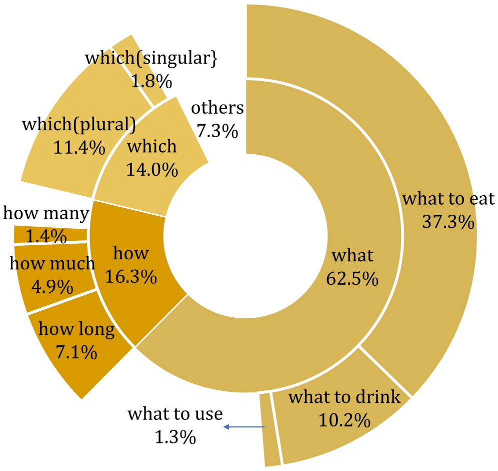


Figure 3: Question types in CMQA. The ring chart on the outer side shows the breakdown of the question types on the inner side, and the blank part indicates the question that does not belong to the summarized question types.

5.1 Models

We decompose the problem into two sub-tasks:

- Extracting text spans from context C as conditions and answers according to question Q.
- Predicting if a relation (condition-answer or coarse-fine) exists between two text spans.

Thus, our approach consists of a span model and a relation model. The span model first takes the context and questions as input and predicts conditions and answers. Then the relation model processes every pair of predicted text spans and judges if there is a relation between them. We find that sharing the contextual representations between the span and relation models will cause a performance decline. We hypothesize that using the same contextual representation to capture both span boundary information and span dependency information is suboptimal.

Span Model Traditional single-span extraction models, which search for the most likely token as the start/end of the answer, are unsuitable for the multiple-span extraction task. Thus, similar to Segal et al. (2020), we cast the task as a sequence tagging problem, which is demonstrated effective on multi-span samples in DROP and QUOREF. We experiment with the well-known BIO tagging (Huang et al., 2015). Concretely, we first concat the context and the question into one sequence and use a pretrained language model to obtain contextualized representations hi for each input token t_i. The representation hi is then fed into a feedforward network to predict the probability distribution of the tag type. Two different modes are used to get the prediction: 1) **Separate Mode** uses different models to predict text spans of different types. These models share the same model structures but are trained independently. 2) **Merged Mode** uses one model to predict all text spans and using multiple begin tags ($B_{condition}$, B_{coarse} and B_{fine}) to distinguish span types.

Relation Model Because the type of relationship depends on the span type, we only need to predict if there is a dependency between a pair of text spans: s_a and s_b. We build the relation model following previous work (Wu and He, 2019; Zhong and Chen, 2021), which gets competitive results in several relation extraction benchmarks, such as ACE04 and SciERC (Luan et al., 2018). Concretely, we first obtain contextualized representations h_i for each input token x_i. Then we concatenate the token representations of the start positions of two text spans and obtain the span-pair representation: $h_{(s_a,s_b)} = [h_{start(a)}; h_{start(b)}]$, where $start(a)$ and $start(b)$ are the indices of start tokens of s_a and s_b. Finally, the span-pair representation $h_{(s_a,s_b)}$ will be fed into a feed forward network to make the binary classification. Moreover, we further investigate using span boundaries to enhance the relation model (**span assist**), where additional markers are used to highlight all conditions and answer spans in a sample. Concretely, we insert special markers (such as '<*fine*>' and '</*fine*>') at the span boundaries in the input layer, and the embeddings of these markers are trainable vectors.

5.2 Evaluation Metrics and Experimental Details

We adopt two evaluation metrics: Exactly Match (EM) and F1-score. **EM:** Prediction of a sample is considered as correct the prediction is equal to the annotation, formally as $anno_{pre} = anno_{ref}$. **F1 score:** The exact steps of calculating F1 scores are the same as that in the IAA study, shown in Algorithm 1 in the Appendix.

In all experiments, we use bert-base-chinese as the encoder to get contextual representations. The

Label type/ Model	Separate EM	 $F1$	Merged EM	 $F1$
condition	42.3	71.8	47.3	72.6
coarse	32.8	65.0	33.1	65.1
fine	44.5	84.3	42.7	83.6
all	19.3	79.4	20.5	79.0

Table 6: Model performance on span extraction. ***all*** refers to evaluate all span types together. We use the micro F1-score to calculate F1-score of ***all***. Note that we only evaluate samples that contain the related span type in EM metric (all samples are evaluated in ***all***).

Model/ Label type	*condition* *acc*	*coarse* *acc*	*fine* *acc*
Separate	89.2	93.6	98.9
Merged	88.3	92.8	99.1

Table 7: Model performance on judging if a certain span type is included in the sample.

implementation is based on Hugging-Face's *Transformers* library (Wolf et al., 2019). We report the averaged test set results of 3 runs for all the experiments. The relation models are trained with ground-truth span labels.

5.3 Experimental Results and Analysis

Span Extraction As shown in the experimental results summarized in Table 6, the separate and merged models get similar performance on the span extracting task. We can see from the table that model performance in EM is really poor, while the performance in F1 is much higher. An important reason is that there exist many simple spans besides those difficult to extract in one sample. For example, there might be answers connected in a one-word interval (e.g., '*ginger, pepper, garlic ...*').

We analyze samples on the development set and find that 47.1% of fine-grained answers and 8.6% of coarse-grained answers are connected to others. In this case, the model will easily extract the others if one is recognized as the answer. However, to maintain the authenticity of the dataset, we did not eliminate or adapt these simple labels. The poor EM score in Table 6 shows that extracting all text span in one sample correctly is rather difficult. We further analyze model performance on judging if a certain span type is included in a sample. The results in Table 7 show the model can achieve very high performance (around 90% accuracy) on this

https://catalog.ldc.upenn.edu/LDC2005T09
https://github.com/google-research/bert

Question: *How long does it take to recover after a needle stick?*

Context: *... After topical application of chlortetracycline eye drops, erythromycin eye ointment and hot compress, you will usually recover after 3-4 days from mild local red nodules. If the pus is formed locally, it usually breaks in 2-3 days, and it will recover in 5-6 days ...*

Answer: *3-4 days, 5 to 6 days*

Prediction: *3-4 days, 2-3 days, 5 to 6 days*

Table 8: An example the merged model's wrong predictions on the development set.

Model/Label type	*corase*	*fine*
Separate	81.8%	77.2%
Merged	76.0%	80.6%

Table 9: Percentage of samples with *distinction error* in wrong-predicted samples from the development set.

test.

The multi-span extraction QA task is similar to NER in the output form. However, in NER, the model extracts all spans of the target entity type. But in the QA task, spans of the correct entity type might be either an answer or not, which should be judged based on the question and context. Thus the model cannot simply predict the answer type and extract all spans of this type as answers, which is one of the main characteristics of QA tasks and challenging for models to solve. Take the case in Table 8 for example, the model's prediction is *'3-4 days, 2-3 days, 5 to 6 days'*. However, *'2-3 days'* is not an answer to this question. We hypothesize that the model has learned that the answer type is 'period' but fails to understand the context and pick out the correct spans.

We denote such error as *distinction error*, which means all the predicted spans are in the correct type, but not all of them can be seen as an answer, or some correct spans in the context are ignored. We manually counted the *distinction error* amount from 200 randomly sampled wrong-predicted samples. Results in Tabel 9 show that *distinction error* occurs in a large proportion of wrong-predicted samples, which demonstrates the characteristic and main challenge of the span extraction task.

Label type/	*condition-answer*		*coarse-fine*	
Model	EM	$F1$	EM	$F1$
Normal♠	3.9	47.9	16.2	54.6
Span-assist♠	6.5	51.1	22.1	57.2
Normal◇	0.4	21.6	7.3	34.9
Span-assist◇	1.8	24.0	13.0	36.5

Table 10: Results of relation models. ♠: results of using the ground-truth span labels. ◇: results of using the prediction of the merged span model.

Relation Extraction We report experimental results of two settings for the relation extraction task: using the ground-truth label as the input and using the prediction of the span model as the input. The results are shown in Table 10.

Due to the limited relation type in CMQA, the relation extraction task seems to be simpler than traditional relation extraction tasks containing multiple relation types (Augenstein et al., 2017; Luan et al., 2018; Gábor et al., 2018). Surprisingly, results in Table 10 show that the competitive method on these tasks performs poorly on CMQA. We hypothesis that one reason is that the text span and relation amount per sample in CMQA are higher than many traditional relation extraction datasets, which often focus on intra-sentence relations. And another reason is that the span type plays a very marginal role in relation extraction in CMQA. In contrast, the entity type is an important feature for judging relationships in most traditional relation extraction tasks (Zhong and Chen, 2021). These differences indicate new approaches needed to be developed to solve relation extraction in CMQA. Results in Table 10 show that introducing the boundary information of other spans can improve model performance. However, the improved model performance is still far from satisfying, which makes providing structured answers as Figure 1 very difficult.

6 Conclusion

In this paper, we propose a new challenge: conditional question answering with hierarchical multi-span answers, which might be widespread in multi-span QA in real-world scenarios. Moreover, we introduce CMQA, which contains conditional and hierarchical samples to study the new proposed task. Data analysis and experimental results show the main characteristics and challenges of CMQA, and the poor model performance demonstrates that the proposed task is challenging for the community to solve. We believe CMQA can serve as a benchmark to study the new proposed task and help build more reliable and sophisticated QA systems.

Acknowledgements

This work is supported by the National Key Research and Development Program of China (No.2020AAA0106400), the National Natural Science Foundation of China (No.61922085, No.61976211, No.61906196) and the Key Research Program of the Chinese Academy of Sciences (Grant NO. ZDBS-SSW-JSC006). This work is also supported by TencentKG, the Open Project of Beijing Key Laboratory of Mental Disorders (2019JSJB06), Yunnan Provincial Major Science and Technology Special Plan Projects (No.202103AA080015) and in part of the Youth Innovation Promotion Association CAS.

References

Isabelle Augenstein, Mrinal Das, Sebastian Riedel, Lakshmi Vikraman, and Andrew McCallum. 2017. SemEval 2017 task 10: ScienceIE - extracting keyphrases and relations from scientific publications. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 546–555, Vancouver, Canada. Association for Computational Linguistics.

Dominic Balasuriya, Nicky Ringland, Joel Nothman, Tara Murphy, and James R. Curran. 2009. Named entity recognition in Wikipedia. In *Proceedings of the 2009 Workshop on The People's Web Meets NLP: Collaboratively Constructed Semantic Resources (People's Web)*, pages 10–18, Suntec, Singapore. Association for Computational Linguistics.

Wenhu Chen, Hanwen Zha, Zhiyu Chen, Wenhan Xiong, Hong Wang, and William Yang Wang. 2020. HybridQA: A dataset of multi-hop question answering over tabular and textual data. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1026–1036, Online. Association for Computational Linguistics.

Eunsol Choi, He He, Mohit Iyyer, Mark Yatskar, Wentau Yih, Yejin Choi, Percy Liang, and Luke Zettlemoyer. 2018. QuAC: Question answering in context. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2174–2184, Brussels, Belgium. Association for Computational Linguistics.

Jacob Cohen. 1960. A coefficient of agreement for nominal scales. *Educational and psychological measurement*, 20(1):37–46.

Pradeep Dasigi, Nelson F. Liu, Ana Marasović, Noah A. Smith, and Matt Gardner. 2019. Quoref: A reading comprehension dataset with questions requiring coreferential reasoning. 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 5925–5932, Hong Kong, China. Association for Computational Linguistics.

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

Kata Gábor, Davide Buscaldi, Anne-Kathrin Schumann, Behrang QasemiZadeh, Haïfa Zargayouna, and Thierry Charnois. 2018. SemEval-2018 task 7: Semantic relation extraction and classification in scientific papers. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 679–688, New Orleans, Louisiana. Association for Computational Linguistics.

Harsha Gurulingappa, Abdul Mateen Rajput, Angus Roberts, Juliane Fluck, Martin Hofmann-Apitius, and Luca Toldo. 2012. Development of a benchmark corpus to support the automatic extraction of drug-related adverse effects from medical case reports. *Journal of biomedical informatics*, 45(5):885–892.

Wei He, Kai Liu, Jing Liu, Yajuan Lyu, Shiqi Zhao, Xinyan Xiao, Yuan Liu, Yizhong Wang, Hua Wu, Qiaoqiao She, Xuan Liu, Tian Wu, and Haifeng Wang. 2018. DuReader: a Chinese machine reading comprehension dataset from real-world applications. In *Proceedings of the Workshop on Machine Reading for Question Answering*, pages 37–46, Melbourne, Australia. Association for Computational Linguistics.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional lstm-crf models for sequence tagging. *arXiv preprint arXiv:1508.01991*.

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

Joël Legrand, Romain Gogdemir, Cédric Bousquet, Kevin Dalleau, Marie-Dominique Devignes, William Digan, Chia-Ju Lee, Ndeye-Coumba Ndiaye, Nadine Petitpain, Patrice Ringot, et al. 2020. Pgxcorpus, a manually annotated corpus for pharmacogenomics. *Scientific data*, 7(1):1–13.

Yi Luan, Luheng He, Mari Ostendorf, and Hannaneh Hajishirzi. 2018. Multi-task identification of entities, relations, and coreference for scientific knowledge graph construction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3219–3232, Brussels, Belgium. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. 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.

Elad Segal, Avia Efrat, Mor Shoham, Amir Globerson, and Jonathan Berant. 2020. A simple and effective model for answering multi-span questions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3074–3080, Online. Association for Computational Linguistics.

Vamshi Krishna Srirangam, Appidi Abhinav Reddy, Vinay Singh, and Manish Shrivastava. 2019. Corpus creation and analysis for named entity recognition in Telugu-English code-mixed social media data. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pages 183–189, Florence, Italy. Association for Computational Linguistics.

Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. 2017. NewsQA: A machine comprehension dataset. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pages 191–200, Vancouver, Canada. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Shanchan Wu and Yifan He. 2019. Enriching pre-trained language model with entity information for relation classification. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM 2019, Beijing, China, November 3-7, 2019*, pages 2361–2364. ACM.

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, Lisbon, Portugal. 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.

Zexuan Zhong and Danqi Chen. 2021. A frustratingly easy approach for entity and relation extraction. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 50–61, Online. Association for Computational Linguistics.

Ming Zhu, Aman Ahuja, Da-Cheng Juan, Wei Wei, and Chandan K. Reddy. 2020. Question answering with long multiple-span answers. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3840–3849, Online. Association for Computational Linguistics.

A Appendix

A.1 Training Annotators

The annotation process is realized by three full-time annotators. All annotators have at least a high school degree and more than one year of full-time annotation working experience. We let the annotators annotate different parts of data according to the guidelines. All annotators and experts are in an online communication group. We encourage annotators to ask experts questions about samples they are not very sure of. We guarantee that experts will answer annotators' questions immediately in this phase. It takes about three days until all annotators think they can work independently. Each annotator has annotated over 100 samples in this phase. Note that this part of data is only used for training and will not be included in the final release.

A.2 Label Amount

We visualize the amount of each label type in Figure 4, where the first subgraph shows the total label amount of the dataset, and the rest of the subgraphs show that of each sample.

A.3 The exact steps of calculating F1 scores for IAA study and Experiment Evaluation

Algorithm 1 describes the exact steps in the evaluation procedure of IAA study and experiments.

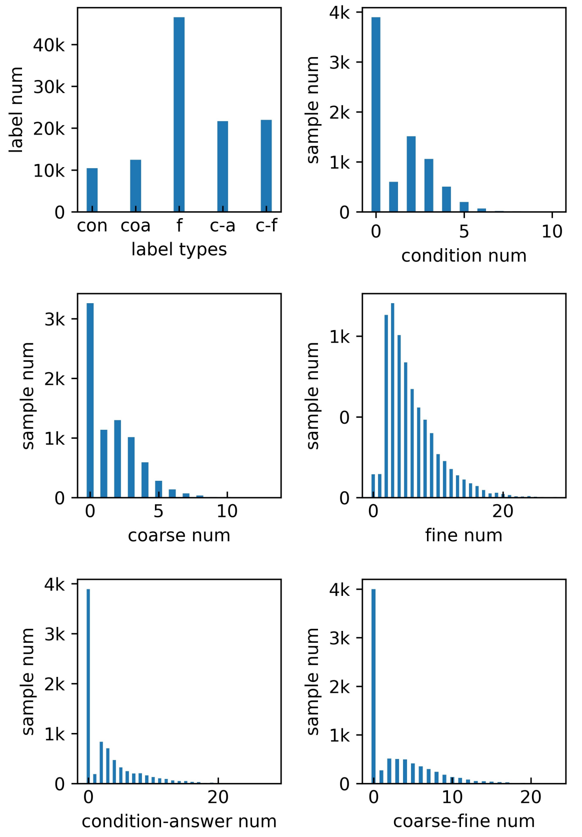


Figure 4: The amount of labels in CMQA. **c-a** refers to condition-answer and **c-f** refers to coarse-fine.

Algorithm 1 Evaluation Algorithm

```
Input: $Anno_{ref}$: the reference annotation; $Anno_{pre}$: the
    perdiction annotation;
Output: $F1 - score$;
function CAL_F1($Anno_{ref}, Anno_{pre}$)
    $recall_{mol} \leftarrow 0$
    $recall_{den} \leftarrow 0$
    $precision_{mol} \leftarrow 0$
    $precision_{den} \leftarrow 0$
    for $anno_{ref}, anno_{pre} \in\ zip(Anno_{ref}, Anno_{pre})$
do
        for $a \in anno_{ref}$ do
            if $a \in anno_{pre}$ then
                $recall_{mol} \leftarrow recall_{mol} + 1$
            end if
        end for
        for $a \in anno_{pre}$ do
            if $a \in anno_{ref}$ then
                $precision_{mol} \leftarrow precision_{mol} + 1$
            end if
        end for
        $recall_{den} \leftarrow recall_{den} + len(anno_1)$.
        $precision_{den} \leftarrow precision_{den} + len(anno_2)$
    end for
    $recall \leftarrow recall_{mol}/recall_{den}$
    $precision \leftarrow precision_{mol}/precision_{den}$
    $F1 \leftarrow 2*recall*precision/(recall+precision)$
    return $F1$
```

$Anno_= \{anno_1, anno_2, ..., anno_n\}$ refers to the annotation for n samples.

A.4 Annotation Principles with Examples

The main principles of annotation with examples are as follows:

- The text span should be as concise as possible. The annotator should exclude non-essential phrases. (e.g., question: *’What to eat for muscle growth?’* context: *’... You can eat some beef ...’*; The answer to this question should be *’beef’* instead of *’some beef’*.)
- If there is a multi-level hierarchical relationship, only the last level is annotated. (e.g., context: *’You can eat some vitamin-rich foods, such as fruits ... Apples are rich in vitamin C ...’*; The *coarse* answer to this question should be *’fruits’* instead of *’vitamin-rich foods.’*, and the *fine* answer should be *’Apples’*.)
- The sample can be abandoned in the following situations:
  - If the context cannot answer the question.
  - Although there are multiple answer spans in the context, these answers are adjacent and can be regarded as one. (e.g., *’ginger, pepper, garlic ...’*)
  - Most of the words in the context are part of the answer. It is better to give the entire context as a long answer in this case. (Because the number of the answer spans of each sample is variable, we did not give a specific context length as the threshold to filter out samples. Experts give examples of abandoned samples and let the annotators make their judgments based on these examples.)
  - The question is a combination of multiple sub-questions. (e.g., *What are the most suitable height and weight for long-distance running?’*) We split such questions into multiple sub-questions in our system.