Generating Clarifying Questions for Information Retrieval

Hamed Zamani, Susan T. Dumais, Nick Craswell, Paul N. Bennett, and Gord Lueck
Microsoft
{hazamani,sdumais,nickcr,pauben,gordonl}@microsoft.com

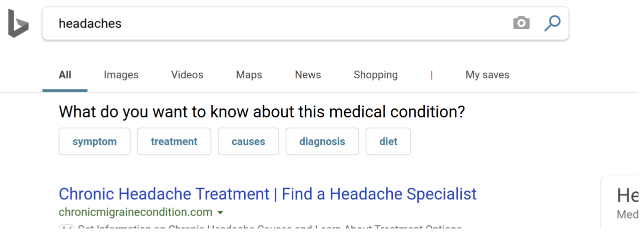
ABSTRACT

Search queries are often short, and the underlying user intent may be ambiguous. This makes it challenging for search engines to predict possible intents, only one of which may pertain to the current user. To address this issue, search engines often diversify the result list and present documents relevant to multiple intents of the query. An alternative approach is to ask the user a question to *clarify* her information need. Asking clarifying questions is particularly important for scenarios with "limited bandwidth" interfaces, such as speech-only and small-screen devices. In addition, our user studies and large-scale online experiments show that asking clarifying questions is also useful in web search. Although some recent studies have pointed out the importance of asking clarifying questions, *generating* them for open-domain search tasks remains unstudied and is the focus of this paper. Lack of training data even within major search engines for this task makes it challenging. To mitigate this issue, we first identify a taxonomy of clarification for open-domain search queries by analyzing large-scale query reformulation data sampled from Bing search logs. This taxonomy leads us to a set of question templates and a simple yet effective slot filling algorithm. We further use this model as a source of weak supervision to automatically generate clarifying questions for training. Furthermore, we propose supervised and reinforcement learning models for generating clarifying questions learned from weak supervision data. We also investigate methods for generating candidate answers for each clarifying question, so users can select from a set of predefined answers. Human evaluation of the clarifying questions and candidate answers for hundreds of search queries demonstrates the effectiveness of the proposed solutions.

ACM Reference Format:
Hamed Zamani, Susan T. Dumais, Nick Craswell, Paul N. Bennett, and Gord Lueck. 2020. Generating Clarifying Questions for Information Retrieval. In *Proceedings of The Web Conference 2020 (WWW '20), April 20–24, 2020, Taipei, Taiwan.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3366423.3380126

1 INTRODUCTION

Search queries are often short and ambiguous, which makes it difficult for retrieval systems to identify the actual user intents. A standard solution to mitigate this issue in modern search engines is result list diversification [43]. This helps the system to satisfy multiple intents of the query in the first result page. An alternative


headaches

All Images Videos Maps News Shopping | My saves

What do you want to know about this medical condition?

symptom treatment causes diagnosis diet

Chronic Headache Treatment | Find a Headache Specialist
chronicmigrainecondition.com

Figure 1: An example of clarifying questions in web search.[1]

solution would be asking questions of the users to *clarify* their information needs. This is of particular importance in interactive information seeking systems with "limited bandwidth" interfaces, such as conversational search systems using speech-only and small-screen devices [14, 15]. The reason is that in such information seeking scenarios, users cannot easily go through a diversified result list and find the document that satisfies their needs. In other words, due to interface limitations, the system can only return a limited number of results, thus being confident about the retrieval performance becomes even more important, which can achieve by asking clarifying questions [2]. In addition, asking natural language questions is the most convenient interaction form in conversational systems and users enjoy such interactions [25]. In addition to conversational systems, we claim that asking clarifying questions is of significance in web search. Figure 1 shows an example of clarifying question and a set of candidate answers in response to the query "headaches". Our user studies showed that users enjoy seeing clarifying questions in web search, not only because of their *functional* benefits, but also due to their *emotional* benefits. In other words, asking clarifying questions about the user query gives a sense of confidence to the user, since the search engine looks more intelligent. To understand the value of asking clarifying questions in web search instead of just showing the candidate answers (similar to query suggestion), we ran a large-scale online experiment and observed over 48% clickthrough rate improvement (see Section 3 for more information). This substantial improvement in a real-world setting clearly describes the value of clarifying questions in web search.

Despite the importance and usefulness of clarifying questions for both web and conversational search, it is relatively less explored in the literature. Recently, Aliannejadi et al. [2] studied the task of asking clarifying questions for open-domain information retrieval. They asked human annotators to generate different clarifying questions for a given query and focused on selecting a good clarifying question from the human-generated question set. However, in a real system, a major challenge is how to *generate* a clarifying question. In addition to its modeling complexity, another challenge is that no data is available for supervised training of such systems (even

Most of the work was done while Hamed Zamani was affiliated with the University of Massachusetts Amherst and was an intern at Microsoft.

WWW '20, April 20–24, 2020, Taipei, Taiwan
© 2020 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-7023-3/20/04.
https://doi.org/10.1145/3366423.3380126

[1]Retrieved on August 15, 2019. We noticed that this feature is not yet available for some international markets.

in major search industry). At the same time, developing a question generation model for an open-domain setting requires large-scale training set, which is quite expensive to collect.

In this paper, we address these challenges by proposing clarifying question generation models that are trained using *weak supervision*. We believe that a good clarifying question can be generated, if the systems is aware of different aspects of the query. Therefore, we propose to use query reformulation data to identify different query aspects. In more detail, we first performed a large-scale query log analysis using the data collected by the Bing search engine to identify a taxonomy of different clarification types required in open-domain information retrieval (Section 4). Our attempts to produce human-generated clarifying questions for different clarification types help us understand that many clarifying questions can be formulated using a small number of question templates. Based on such observation, we propose a simple yet effective rule-based model for selecting and filling out pre-defined question templates (Section 5.2). This rule-based model uses query aspects and query entity type information for generating clarifying questions. We further propose a machine learning model trained based on maximum likelihood using the data generated by the rule-based model. In fact, the rule-based model produces weak supervision data for training our question generation model, which is a sequence-to-sequence model based on recurrent neural networks (Section 5.3). We also propose to further train the model using a reinforcement learning algorithm whose reward function is a non-differentiable function estimating the clarification utility (Section 5.4).

In addition to clarifying questions, we propose a solution to generate candidate answers for a given pair of query and clarifying question. The objective in our model is a monotone submodular function, which allows us to use an efficient greedy approximation algorithm for optimization. Intuitively, our model selects the phrases that are good answers for the given clarifying question in the context of the query (Section 5.5).

To evaluate our models, we use human annotations for a diverse set of real search queries. We ask the trained annotators to evaluate the generated clarifying questions and candidate answers. Our results demonstrate that the weakly supervised models improve the weak labeling model (i.e., the rule-based model). In addition, the reinforcement learning model performs the best with the quality of producing 38% Good, 60.4% Fair, and 1.5% Bad questions with the standards of a real-world production system. 28.4% of the generated questions are specific questions. We have also evaluated the quality of candidate answers and the secondary result page quality obtained by clicking on each of the candidate answers.

2 RELATED WORK

In this section, we review prior work on asking clarifying questions, query reformulation in search, and weak supervision.

Asking Clarifying Question. Research on clarifying questions has attracted considerable attention in natural language processing and information retrieval [2, 41, 45]. The studies on human-generated dialogues on question answering websites have provided analysis on the intent of each utterance [37], including clarifying questions [8]. Generating questions whose answers appeared in a given passage has been extensively studied in the machine reading comprehension literature [19, 23, 60]. There is a more relevant line of research that focuses on asking clarifying questions for pointing out missing information in a passage. For example, Rao and Daumé III [42] proposed a model for generating a clarifying question for identifying missing information in a closed-domain setting. They proposed a reinforcement learning model that maximizes a utility function based on the added value obtained by the potential response to the clarifying question [41]. Trienes and Balog [49] focused on identifying unclear CQA posts that require further clarification. Asking clarifying questions have been also studied in other contexts, such as speech recognition [45] and dialogue systems [7, 16, 32, 38], which are fundamentally different from asking a question to clarify the information need of users.

In the realm of IR, Kiesel et al. [25] studied the impact of voice query clarification on user satisfaction and concluded that users like to be prompted for clarification. Earlier in the TREC HARD Track [3], participants could submit a form containing clarifying questions in addition to their runs. The importance of asking for clarification in conversational search has been also raised by Radlinski and Craswell [39]. Yang et al. [53] proposed a model for retrieving the next question in conversation. Most recently, Aliannejadi et al. [2] studied the task of selecting clarifying questions from a set of human-generated questions for open-domain information seeking. Unlike these studies, our work focuses on *generating* clarifying questions for open-domain information retrieval. Coden et al. [13] studied the task of asking clarifying questions for entity disambiguation mostly in the form of "did you mean A or B?". This approach is only useful for entity disambiguation, and cannot be applied to many queries, including faceted queries.

Much work has been also done on conversational recommender systems by asking questions about different item attributes for providing more accurate recommendation [46, 59]. For instance, Christakopoulou et al. [12] designed a system that can interact with users to collect more information about their preferences for venue recommendation. The unique challenges and techniques used for identifying different aspects of search queries and generating clarifying questions in response to real queries are fundamentally different from those reviewed in this section.

Query Reformulation, Suggestion, and Auto-Completion. Previous work [21, 29] showed that an overlapping query syntax between two consecutive queries in a session is an indication of low satisfaction with the first query, and users describe their intents more clearly in the second query. Based on such observation, in this paper, we focus on additive overlapping query reformulations to identify different aspects and intents behind the original query (see Section 5.1). Query reformulation data has been previously used for various IR tasks, including query suggestion and query auto-completion [9, 34, 52]. The goal of query suggestion is recommending a set of possible queries that are likely to be searched by the user. Boldi et al. [4, 5] used query reformulation data to construct a query flow graph. Later on, Diaz [18] looked at query reformulation as a discrete optimization problem by constructing an unweighted graph of queries. Szpektor et al. [48] employed entity type information together with query reformulation data for improving the query suggestion quality in tail queries. In this work, we also found entity type information useful for generating clarifying questions for tail queries. Although query suggestions are also often generated from query reformulation data, they are fundamentally different from candidate answers to a clarifying question. The reason is that in our case, the candidate answers should clarify the information need of the user behind the current search query. While, in query suggestion, the next search query might be in another topic that are usually searched together. For example, a sensible next query for "parkinson" can be "alzheimer". Query auto-completion is similar to candidate answer selection in the sense that they both focus on additive reformulations. However, any possible reformulation is not necessarily a good candidate answer for clarification. Despite these fundamental differences, we use a few query auto-completion and suggestion baselines in our experiments to evaluate the quality of candidate answers. In addition, our online experiments in Section 3.3 show the substantial impact of clarifying questions on user engagement in terms of clickthrough rate (i.e., over 48% improvement).

Weak Supervision. Limited training data has been a perennial problem in information retrieval, and many machine learning-related domains [58]. This has motivated researchers to train models using *pseudo-labels*. As widely known, deep neural networks often require large-scale data for training. Training neural IR models based on pseudo-labels has been shown to produce successful results in various tasks, including ad-hoc retrieval [17, 57], query performance prediction [56], and query disambiguation [28]. This learning approach is called *weak supervision*. Dehghani et al. [17] proposed training a neural ranking model for ad-hoc retrieval based on the labels generated by an existing retrieval model, such as BM25. Following these studies, the idea of training neural IR models with weak supervision has been further employed in [31, 35, 56]. Recently, Zamani and Croft [55] provided theoretical foundations for explaining the successful empirical results achieved by weakly supervised IR models. In this paper, we also use a weak supervision strategy to train our model based on the output of a rule-based question generation model. The main difference between our work and the existing weak supervision models in IR is that our weak signals is a text that should be generated, rather than a single label.

3 ON THE USEFULNESS OF SEARCH CLARIFICATION

One of the major motivations for asking clarifying question in IR is closing the gap between the traditional "query-response" paradigm and interactive conversational information seeking systems. Kiesel et al. [25] showed that asking clarifying questions in case of ambiguous and faceted queries is a convenient interaction form in conversational system and users enjoy such interactions. In addition, modern search engines often return a diversified result list in response to ambiguous or faceted queries in order to satisfy different possible query intents. However, most conversational systems provide a speech-only or a small screen interface. These interfaces do not allow information systems to present a long diversified result list. Therefore, when it comes to conversational systems, understanding query intent for providing confident and accurate results becomes even more important. This can be achieved by asking questions to clarify the user information need [2].

In addition to conversational systems, we claim that clarification is also useful in typical web search interfaces. Figure 1 demonstrates an example of using clarification in web search. In such case, the system produces a clarifying question and a set of clickable candidate answers. To study the usefulness of clarification for web search, we rely on users. In the rest of this section, we use the method used for generating clarifying questions in Bing, and present a set of user studies and online experiments to highlight the usefulness of clarifying questions in web search. Since this is not the core of the paper, we keep brevity in the analysis reported here.

3.1 User Study I

In the first user study, we interviewed five participants, including three female and two male participants aged 24 to 48 years old, from five different states in the United States and different educational backgrounds. We defined four diverse scenarios, and asked them to use the search engine we provided (the interface is shown in Figure 1). The scenarios include:

- You have acquired a lot of favorites in your web browser, but they are old and you want to get rid of them.
- You are moving to San Clemente, CA, and your teenaged relative is going to start at San Clemente High School.
- Your doctor has recommended you start taking a new supplement called Vitar C Capsule Extended Release.
- You are interested in learning more about Irish castles.

After completion of the task, all participants showed high enthusiasm for the clarification pane which includes both clarifying question and candidate answers. When the clarification pane was relevant, the participants described it as "convenient and easy", and they believed that "it saves time and steps". Interestingly, they mentioned that "it sometimes cues the user to things they may not have considered". They believed that the clarification pane "helped them find more relevant results".

They were also asked to provide their opinion on non-relevant or low quality clarifications. Interestingly, they did not think that the search experience was degraded. One participants mentioned "It's like when I look at iPhones, and eBay says 'since you looked at iPhones you may be interested in these hair curlers!' And I'm like, well that's weird, whatever". We believe that this happens because of the high quality of search result list. On the other hand, the participants believed that the quality of the secondary result page (after clicking on one of the candidate answers in the clarification pane) perceived the usefulness of the clarification pane. Motivated by this observation, we also evaluate the quality of the secondary result page in our experiments (see Section 6.5).

3.2 User Study II

Followed by the positive feedback received from the first user study, we conducted a second round with a bit larger participant pool and a more realistic setting. We provided 24 participants with the same search engine that features the clarification pane. Similar to the previous user study, the participants were diverse in terms of gender, location in the United States, and educational background. We asked the participants to perform their everyday searches through our system for two weeks. The participants were further interviewed in four different sessions to give us their honest opinion. The participants have been asked a set of questions, such as "how do you feel about the search engine asking a question back to you?". The summary of the interviews are as follows:

- They believed that it has **functional benefits**. In particular, they pointed out that "questions help guide users in the right direction and to the right conclusion".
- They believed that clarification pane also has some **emotional benefits**. In more detail, it brings to users a sense of confidence that the search engine understands what the user wants. Moreover, it gives the users a sense of security and coming to the right conclusion. They pointed out that sometimes, especially when it comes to product search, they feel less stress when the search engine asks questions on different features of the product. Because without it, they often feel they forgot to check all necessary properties of the product.

They are finally asked to rate every feature in the search engine result page (including the entity card, query suggestion, similar questions, etc). The clarification pane was amongst the most favorite features with the rating of 9.14 out of 10.

3.3 Online Experiment

As the last step to identify the usefulness of clarification pane for web search, we performed an online experiment using Bing. The main goal of our online experiment was to understand the benefit of clarifying questions, as opposed to just showing some options, such as query suggestion. We consider the following two treatments: (1) clarifying questions and candidate answers, and (2) a static title (i.e., "select one to refine your search") and candidate answers.

In other words, this experiment shows the importance of asking clarifying questions. Note that all other details, such as interface design, position in the SERP, and search result quality were the same in both treatments. We ran the experiment for one week with 2.5 million users from the EN-US market for each treatment. We obtained 48.57% more engagements (relative) using the clarification pane with clarifying question, compared to the one with static title (i.e., similar to query suggestion). The engagement is measured by clickthrough rate, and the improvements were statistically significant based on paired t-test ($p < 10^{-20}$).

3.4 Summary of Findings

This section provides multiple evidence to show the importance of clarification in information seeking systems, summarized as below:

- In conversational information seeking, asking question is the most convenient interaction to clarify the user information need. Interfaces with small or no screen (speech-only) signify the importance of asking for clarification.
- In our user studies, clarification pane was found useful by the users. They pointed out its both functional and emotional benefits. Emotional benefits include the sense of confidence and security.
- Over 48% relative clickthrough rate improvement compared to query suggestion in our large-scale online experiment demonstrates the usefulness of clarifying questions in web search.

4 TAXONOMY OF CLARIFICATION TYPES

This section summarizes a number of different type of clarification required in an open-domain information seeking system, such as web search. This taxonomy was derived by analyzing a large-scale query log. We looked at thousands of query reformulations sampled from the Bing query logs. Existing query reformulation taxonomies [6, 24, 27, 29] cover broad types of reformulations (e.g., generalization, specialization, change in topic or search domain). In this paper we focus on refining the types of specializations that are important in generating clarifying questions. The general idea in creating this taxonomy is that the specializations seen in query reformulations indicate the types of clarifications that will be required for open-domain information seeking. For example, when the query "trec" is followed by the query "trec conference", it gives us information about one intent of the original query. The clarifications we identified are summarized below.

Disambiguation: Some queries are ambiguous and could refer to different concepts or entities. For example, the query "trec" can refer to either "Text Retrieval Conference" or "Texas Real Estate Commission". A clarifying questions could ask about whether the user's intent is to find the conference or the real estate commission.

Preference: Other queries are not ambiguous, but a clarifying question can help identify a more precise information need. Four major subcategory of preference clarifications are:

- Personal information ("for whom"): personal information, such as gender, age, language, and expertise, can limit the search space. For example, a query about "sneakers" might be followed by "for women" or by "for kids", which suggest useful clarifying questions.
- Spatial information ("where"): spatial information is also reflected in reformulations in many cases. For example, a user looking for an apartment to rent may have different neighborhood preferences, which can be clarified by asking a question.
- Temporal information ("when"): some queries have a temporal aspect which can be clarified by the system. For example, the query "wsdm" can refer to either "wsdm 2019" or "wsdm 2020".
- Purpose ("for what purpose"): if the answer to a query depends on the purpose of user, a clarifying question can seek to identify the purpose. For example, a user searching for "apartment" may be interested in renting or buying.

Topic: If the topic of the user's query is too broad, the system can ask for more information about the exact need of the user. Topic clarifications include:

- Sub-topic information: The user might be interested in a specific sub-topic of the query.
- Event or news: based on an event or breaking news, many users often search for a topic related to the news, while the query out of the context of that event or news may have different meaning.

Comparison: Comparing a topic or entity with another one may help the user find the information they need. For example, for a user who wants to purchase a gaming console, the system may ask whether the user wants to compare xbox with play station.

Note that different reformulations of the same query may lie in multiple categories in this taxonomy.

5 GENERATING CLARIFYING QUESTIONS

Given different clarification types required in an open-domain information seeking system (see Section 4), in this section, we introduce three approaches for generating clarifying questions. Figure 1 and Table 6 show some examples of clarifying questions. Our first approach (called RTC) is a simple yet effective model based on a rule-based slot filling algorithm (Section 5.2). We then introduce our second model (called QLM) which is a weakly supervised text generation model based on recurrent networks trained using maximum likelihood (Section 5.3). We finally present our third model (called QCM) which maximizes a clarification probability using reinforcement learning (Section 5.4). Since creating large-scale human-generated clarifying questions for open-domain information retrieval is costly and time consuming, none of our methods rely on supervised clarifying questions for training. In fact, QLM and QCM are trained based on weakly supervised data generated by the rule-based model.

In addition to clarifying questions, we propose a model for generating candidate answers in Section 5.5. All of these models rely on a query aspects generation method presented below.

5.1 Query Aspects Generation

In order to generate an appropriate clarifying question, we must identify different aspects of the query. One can employ different information sources for identifying query aspects, including query reformulation data, clickthrough data, and retrieved documents.

In this paper, we focus on **query reformulation data** with the goal of finding query reformulations that reveal different aspects of the query. Regarding our initial query log analysis, the users mostly clarify their information needs by adding one or more terms to their original query. This is often called query specialization [27]. Therefore, we are interested in the query reformulations in which the reformulated query specifies what aspect of the query is in the need of user. In more detail, we follow the steps below:

(1) We first collect large-scale web search query logs (see Section 6.1).
(2) We then extract a set of query reformulation triples (q, qq', c) (or $(q, q'q, c)$), which denotes that the query q is followed by the query qq' (or $q'q$) in the same search session (i.e., immediate successive queries) with a frequency of c, when it is aggregated over the whole query log data for all users. Note that qq' is the concatenation of the query text q and a term or phrase q', where $|q'| > 0$. Note that if q' starts with (or ends with) a stopword, we drop that stopword. Throughout this paper, we use the following notation for such reformulation: $q \xrightarrow{c} q'$. For example, if the query "running shoes" is followed by "running shoes for women" with the frequency of 10, then we have: "running shoes" $\xrightarrow{10}$ "women". Similarly, if the query "shoes" is followed by the query "running shoes" with a frequency of 12, we have: "shoes" $\xrightarrow{12}$ "running".
(3) Query reformulation data can be extremely sparse, especially for tail queries. To mitigate this issue, we aim at predicting unseen reformulations. This is similar to a standard collaborative filtering task. Therefore, we used neural collaborative filtering (NCF) [22], a state-of-the-art collaborative filtering model, for predicting possible missing reformulations. We used mean squared error (MSE) as the loss function and trained the model for 20 epochs. We took 5 random negative samples per positive instance (each positive instance is a unique query reformulation in the data). We used 50 latent factors per query. After training, we can compute a *weight* for each possible query reformulation.
(4) We lastly compute a probability distribution for query aspects as follows: $p(q \rightarrow q') = \frac{\text{weight}(q, q')}{\sum_{q''} \text{weight}(q, q'')}$, which is the normalized values obtained by the NCF model.

5.2 RTC: A Template-based Approach

A major challenge in generating clarifying questions for open-domain information retrieval is the lack of training data. Writing a number of clarifying questions for a number of random queries made it clear to us that most clarifying questions can be formulated using a few question templates. Therefore, we went through the clarification types taxonomy discussed in Section 4 and produced the following question templates that cover most clarification types:

(1) What do you want to know about QUERY?
(2) What do you want to know about this QUERY_ENTITY_TYPE?
(3) What ASPECT_ENTITY_TYPE are you looking for?
(4) Whom are you looking for?
(5) Who are you shopping for?

We found a simple yet effective rule-based template completion (RTC) model. For each query, we first compute three variables: (1) QUERY: query string, (2) QUERY_ENTITY_TYPE: entity type of the query; `null`, if unknown, and (3) ASPECT_ENTITY_TYPE: the entity type for the majority aspects of the query. If the percentage of query aspects with the entity type `et` exceeds a threshold τ, this variable would be equal to `et`, otherwise `null`. We empirically set τ to 70%.

We then select a question template using the following rule-based algorithm:

- If ASPECT_ENTITY_TYPE is related to personal information, such as gender or age, and if QUERY_ENTITY_TYPE is a product (or related), we choose template (5). If QUERY_ENTITY_TYPE is not a product, we choose template (4).
- If still no template is selected and if ASPECT_ENTITY_TYPE is not `null`, we choose template (3).
- If still no template is selected and if QUERY_ENTITY_TYPE is not `null`, we choose template (2).
- If still no template is selected, we choose template (1).

In fact, the template numbers show their priority, and we select the template with the highest priority that can match the query given the above rules. Despite its simplicity, this rule-based approach often generates appropriate clarifying questions.

5.3 QLM: Question Likelihood Maximization

In this subsection, we describe QLM, a weakly supervised neural question generation model based on maximum likelihood training. QLM first encodes the query and its different aspects, then generates a natural language clarifying question using a decoding component. The architecture of the model is presented in Figure 2. As shown in the figure, QLM consists of a hierarchical encoder with the following components:

- Query Encoder: The query encoder takes the query text in addition to its entity type (which can be "unknown") and returns a d-dimensional representation for the query.
- Single Aspect Encoder: This component learns a representation for each $q \longrightarrow q'$ extracted by the Query Aspect Generation model (see Section 5.1). Note that the parameters for all the Single Aspect Encoders are shared.
- Query Aspects Encoder: This component takes the representations for the top k query aspects (sorted based on their probability computed by the Query Aspects Generation model) and computes a high-dimensional representation.

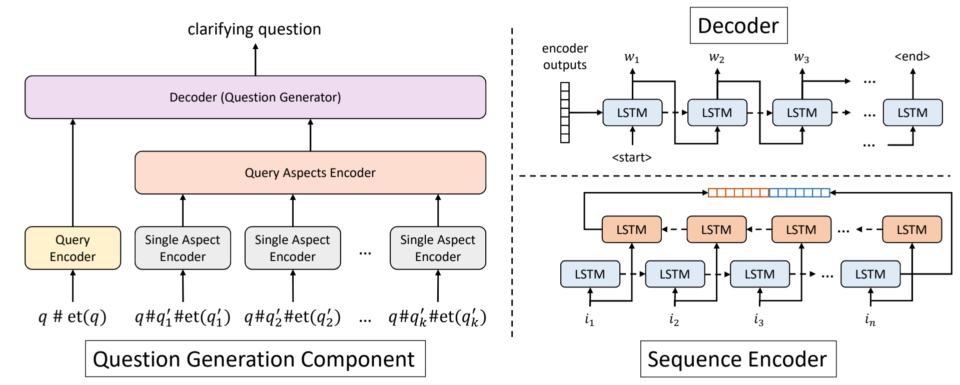


Figure 2: An overview of the question generation component (left) and the network architectures for the decoder (top right) and all the encoders (bottom right).

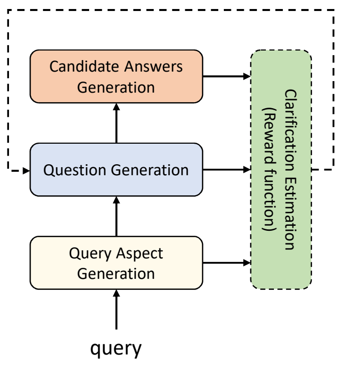

Candidate Answers Generation
Question Generation
Query Aspect Generation
query
Clarification Estimation (Reward function)

Figure 3: Overview of the QCM framework.

All of these encoding components are modeled using a bidirectional long short-term memory network (BiLSTM) [44], shown in the bottom right of Figure 2. In fact, an input sequence (either term representation or single aspect representation) is fed to the BiLSTM network. Due to the short length of inputs in our case, the BiLSTM network works sufficiently efficient and effective. In Query Aspect Encoder, each single aspect representation is a sequence input, ordered by its weight. The BiLSTM network can be simply replaced by any other sequence encoder, such as Transformers [50].

The decoding component takes the concatenated representations learned by Query Encoder and Query Aspects Encoder in order to generate a natural language clarifying question. The architecture of this decoding component is shown in the upper right of Figure 2, which is similar to the one used in the seq2seq model [47]. As depicted, the encoder outputs are used as the initial memory for the first LSTM cell and a start token is fed to the cell. This LSTM cell generates a representation with d dimensions. By feeding this representation to a fully connected network with a single layer and the layer size of vocabulary size, and applying a softmax function to the output of this network, we can obtain a probability distribution over all vocabulary terms (i.e., unigram language model). We train our models based on seq2seq training [47]. In more detail, we use the cross entropy loss function to minimize the divergence between the generated unigram distribution and the true distribution. In true distribution, the target term has a probability of 1 and the other terms have zero probability.

Our model is trained based on the clarifying questions generated by RTC as a weak supervision data. We expect QLM to generalize the observed training set and perform better than the RTC model. The reason is that entity type information plays a key role in RTC, and it can be incomplete. Therefore, QLM that has access to different query reformulations can automatically generate an appropriate clarifying question, even if the query entity type is missing.

At inference time, we take the term with highest probability from the generated unigram distribution, and feed that term to the next LSTM cell as the input (see Figure 2). Using this procedure, the model keeps generating terms until observing an `END` token.

5.4 QCM: Query Clarification Maximization

As explained earlier, QLM maximizes the likelihood of generating the questions observed in the training set. Therefore, it tends to generate common questions in the training set. This has also been observed by Rao and Daumé III [42]. This even gets worse when the training data is noisy, which is the case in weak supervision.

To address this issue, we propose QCM, a machine learning model that generates clarifying questions by maximizing a clarification utility function. As we will see later, this utility function is not differentiable. Therefore, we optimize our model using reinforcement learning. The architecture of the QCM framework is shown in Figure 3. In fact, this model generates clarifying questions and candidate answers simultaneously, and the reward function measures their clarification utility. The network architecture of the question generation model in this framework is the same as the QLM model. The query aspects generation component is also the same as the one described in Section 5.1. We will describe the candidate answer generation model later in Section 5.5. Here we explain how the optimization in QCM works.

We use the mixed incremental cross-entropy REINFORCE algorithm (MIXER) [40] for training our model. The reason is that since our reward function only measures the clarification utility, which makes it difficult to force the model to generate natural language clarifying questions. In such cases, MIXER takes advantage of maximum likelihood training to make sure that the generated questions are similar to those in the training set and then it uses the exploration-exploitation property of REINFORCE to maximize the ultimate reward function. In more detail, we first train our question generation model using maximum likelihood (same as QLM). We further train this pre-trained model using REINFORCE [51] by minimizing the following loss function:

$$L = -(r(q^*) - r(q^*_{QLM})) \sum_{t=1}^{T} \log p(q^*_t | q^*_1, \cdots, q^*_{t-1})$$

where $r(\cdot)$ denotes the reward function, q^*_{QLM} is the question generated by QLM, q^* is the question generated by QCM, and T is the sequence length. Minimizing this loss function using gradient-based optimization methods pushes the sequence likelihood distribution towards the one with the highest reward value. To let the model explore the question generation space, instead of taking the term with highest probability (as done in QLM) we randomly sample a term from the unigram distribution generated by the model.

In the following subsection, we explain our clarification utility that is used as the reward function.

5.4.1 *Clarification Utility Estimation.*

To compute the reward function, we estimate the clarification probability for a pair of query and clarifying question. In other words, this component computes the probability that the clarifying question q^* clarifies the information need of user when submitting the query q. Formally, the goal is to estimate $p(c = 1|q, q^*)$ where c is a binary variable for clarification. Given the law of total probability, this probability is computed as follows:

$$p(c = 1|q, q^*) = \sum_{a \in A} p(c = 1|a, q, q^*) p(a|q, q^*)$$

where A is the set of all possible candidate answers. Intuitively, the clarification probability of a clarifying question is computed based on the clarification probability of each candidate answer (i.e., $p(c = 1|a, q, q^*)$) and the quality of each candidate answer (i.e., $p(a|q, q^*)$).

Estimating $p(c = 1|a, q, q^*)$ depends on the intent of the user. In other words, a candidate answer may clarify an intent behind the query, while may not clarify another intent. Therefore, we estimate this probability as follows:

$$p(c = 1|a, q, q^*) = \sum_{i \in I_q} p(c = 1|i, a, q, q^*) p(i|a, q, q^*)$$

where I_q is the set of all possible intents behind the query q. In this equation, $p(c = 1|i, a, q, q^*)$ computes the clarification probability of each candidate answer for each query intent, and $p(i|a, q, q^*)$ computes the probability of each intent.

The intent set I_q is generally unknown. However, it can be estimated based on the top n query aspects computed using the approach presented in Section 5.1 ($n = 20$).

Estimating $\mathbf{p(i|a, q, q^*)}$: The intent behind a query only depends on the user and the query she submitted, and thus is independent of the clarifying question and its candidate answers. Therefore, $p(i|a, q, q^*)$ can be estimated using maximum likelihood estimation, which will be equal to $p(q \rightarrow i)$ (see Section 5.1).

Estimating $\mathbf{p(c = 1|i, a, q, q^*)}$: For estimating the clarification probability, we hypothesize that the more similar the candidate answer a to the intent i, the higher the clarification probability. On the other hand, the user only clicks on one candidate answer, therefore, we would like to have a diverse set of candidate answers that cover different intents behind the query. Taking all these into account, we estimate this probability as follows:

$$p(c = 1|i, a, q, q^*) = \begin{cases} \text{sim}(a, i) & \text{if } a = \arg\max_{a' \in A} \text{sim}(a', i) \\ 0 & \text{otherwise} \end{cases}$$

where the similarity function is computed using the normalized cosine similarity of average word embeddings [54]. In more detail, the most similar candidate answer to an intent can only get a similarity score for clarifying the intent. This equation makes sure that the candidate answers are diverse.

Estimating $\mathbf{p(a|q, q^*)}$: We should estimate the probability of a being an answer to q^* in the context of q. This probability depends on the query, its entity type, and the answer type required by the clarifying question. Therefore, we estimate this probability using the following information: (1) query text, (2) query entity type, and (3) answer entity type (if the clarifying questions specifies the entity type of answers, then it should be taken into account for computing the answer probability).

Therefore, we train a feed-forward network with a single hidden layer (similar to the word2vec model [33]). In more detail, we concatenate the average word embedding of the query terms, the query entity type embedding, and the answer entity type embedding as the input of the network, and all possible candidate answers as the output. We train the model using the noise contrastive estimation (NCE) loss. We train this component using the query reformulation data, described in Section 5.1. In other words, based on each unique reformulation $q \longrightarrow q'$, we create a training input example (q, entity type(q), entity type(q')) and training label q'. Similar to the word2vec idea, this model can generalize over different entity types and similar texts. The word embedding matrix is initialized by pre-trained vectors (see Section 6.2 for more information).

5.5 Candidate Answer Selection

For each clarifying question, we are interested in generating up to m candidate answers. To do so, we generate a candidate answer set A that maximizes the clarification probability $p(c = 1|q, q^*)$ (see Section 5.4.1). Given the discrete nature of the candidate answer set, solving this problem is NP-hard. However, the way that we estimate this probability in Seciont 5.4.1 makes it a monotone submodular function. For the sake of space, we omit the straightforward submodularity proof. Thus, a greedy algorithm can lead to a $1 - \frac{1}{e}$ approximation. Therefore, we use a greedy algorithm that generates candidate answers one by one, called CAS. This speeds up the candidate answer generation process. Its computation complexity is $m|V|$, where V denotes the vocabulary set. At each step i where $i < m$, if the utility gain compared to the previous step is less than a threshold τ' we stop selecting options and also drop the last one. Since CAS is used as a component in the reinforcement learning of QCM, its efficiency is crucial for efficient training. To speed up the CAS, for each query q, we select the candidate answers from the top 30 words in the $p(q \longrightarrow \cdot)$ distribution (see Section 5.1).

6 EXPERIMENTS

Evaluating the quality of clarifying questions and candidate answers is challenging. We use trained human annotators to evaluate the quality of clarifying questions (Section 6.4), quality of candidate answer set (Section 6.5), and quality of secondary result page by clicking on candidate answers (Section 6.6).

6.1 Data

The proposed methods require query reformulation data (for query aspects generation) and entity type information.

Query Reformulation Data: The query reformulation data was obtained from the Bing search query logs, randomly subsampled from the data collected in a 1.5 year period of the EN-US market. This data is a set of triples (q, q', c), where c is the frequency of the $q \longrightarrow q'$ query reformulation in the same session. We only kept the query reformulations with a minimum frequency of 2. The data contains over 1.6 billion unique query reformulation pairs and over 500 million unique queries.

Entity Type Data: Since we would like to have entity types with high coverage, existing knowledge bases do not meet our expectations. Therefore, we decided to use an open information extraction toolkit to extract "is a" relations from a large-scale corpus. Since web search queries and web documents cover a wide range of topics, we decided to create a corpus based on a diverse sample of Bing search snippets. This resulted in an over 35 petabyte text corpus. We then ran Reverb [20], an open-source IE toolkit, on the collected data and kept the "is a" relations with the confidence of at

Table 1: Question generation evaluation results. †/‡ denote significant improvements compared to RTC/QLM, respectively.

Model	Quality					Specificity					Quality-Specificity Corr.	
	Good	Fair	Bad	score (linear)	score (exp)	Specific	Fair	General	score (linear)	score (exp)	Pearson's ρ	Kendall's τ
RTC	27.6%	71.2%	1.2%	1.264	1.540	7.6%	52.8%	39.6%	0.680	0.756	-0.35	-0.34
QLM	32.8%	66.4%	0.8%	1.320	1.648	22.4%	44.8%	32.8%	0.896†	1.120†	-0.40	-0.38
QCM	38.0%	60.4%	1.6%	**1.364**†‡	**1.744**†‡	28.4%	46.0%	25.6%	**1.028**†‡	**1.312**†‡	-0.18	-0.16

least 96%. This results in over 27 millions relations for over 20 millions unique phrases. The data contains over 6 millions entity types, out of which over 17,000 entity types have a minimum frequency of 10. This data covers the entity types for over 40% of search queries.

In the following, we describe our query sampling strategy:

Training Query Set for Weak Supervision: As mentioned above, the proposed machine learning models are trained using weak supervision. To construct the training set, we first generated clarifying questions for all the queries in the sampled query logs (i.e., over 500 million unique queries) using the RTC model. Out of which, we sampled $100k$ queries for training the QLM model. In our query sampling, we made sure to include the queries that cover a wide range of clarifying questions. This is necessary to allow the model learn to generate different clarifying questions, and avoid overfitting on general questions. We sampled an additional $5k$ queries with the same strategy to create a validation set. For training the QCM model, we used $80k$ of the training query set for pre-training the question generation model and the remaining $20k$ for fine tuning the model using reinforcement learning.

Evaluation Query Set: To evaluate our models, we randomly sampled 250 queries from the Bing query logs. This query set has a uniform distribution across the following four buckets (i.e., 62 or 63 queries per bucket): (1) Head & Many Reformulations, (2) Head & Few Reformulations, (3) Tail & Many Reformulations, and (4) Tail & Few Reformulations. If a query has over 20 unique reformulations, it is considered as a query with many reformulations, while few reformulations means less than or equal to 5 unique reformulations. Being "head" or "tail" is determined based on the frequency of the query in the search logs. We made sure that the intersection between the training, validation, and evaluation sets are empty.

6.2 Implementation Details

We implemented our model using TensorFlow [1]. In all experiments, the network parameters were optimized using the Adam optimizer [26] with its default hyper-parameters. The batch size in all the experiments was set to 128. We selected the learning rate from $\{1 \times 10^{-5}, 5 \times 10^{-4}, 1 \times 10^{-4}, 5 \times 10^{-4}, 1 \times 10^{-3}\}$ based on the loss value on the validation set. The size of hidden parameters in LSTM cells was set to 64. In all the experiments, we used the standard early stopping strategy to avoid overfitting. The parameters k (the number of aspects encoded in the question generation model) and m (maximum number of candidate answers per clarifying question) were set to 10 and 5, respectively. The word embedding dimensionality was set to 100 and the embedding matrix was initialized by the pre-trained GloVe [36] vectors learned from Wikipedia 2014 and Gigawords 5. For text pre-processing, we removed non-alphanumerical characters and lower-cased all text. We further filtered out queries with detrimental contents using a proprietary tool. We use NLTK [30] for text tokenization.

6.3 Human Annotation

We evaluate our models using human annotation. We assigned each human intelligent task (HIT) to three annotators and obtained the labels based on majority voting. The hired annotators were well trained and informed about the task and its applications. Each labeling task asks the annotators to choose between Good, Fair, Bad, and Detrimental labels. Detrimental refers to the sensitive contents that are harmful for real commercial systems. As mentioned in Section 6.2, we removed detrimental contents from the data and thus we did not observe any detrimental label in our human annotations. Therefore, hereafter, we do not mention this label. In some rare cases (less than 4%), there is no agreement between the annotators (i.e., no label with more than 1 voter). In such cases, we considered Fair (i.e., the middle label) as the label. The overall Fleiss' kappa inter-annotator agreement is 64.58%, which is considered as good.

Each HIT is designed for a single query. It starts with a SERP Review step that asks annotators to read and review three pages of the search results returned by Bing. The goal is to make sure that the annotators are familiar with different aspects of the query. We further present some labeling tasks to the annotators. In the next section, we describe each labeling task and discuss the results.

6.4 Evaluating Question Quality

In this subsection, we evaluate the quality of the generated clarifying questions, independent of candidate answers. In our HIT, after reviewing the search engine result page, we present the generated clarifying question to the annotator and ask for a question quality label. At this stage, annotators are not aware of the candidate answers. We gave detailed definitions, guidelines, and examples for each of the Good, Fair, and Bad labels. In summary, the guideline indicates that a Good question should correctly address and clarify different intents of the query, and should be fluent and grammatically correct. If a question fails in terms of any of these factors, but still an acceptable question to be asked in a real system, the label Fair is desired. Otherwise, we asked the annotators to give a Bad label to the question. We also identify the specificity of the generated questions. The results are reported in Table 1. In addition to the percentage of each label obtained from the annotators, we compute two scores for question quality. (1) The **linear score** is obtained by assigning the weight 0, 1, and 2 to Bad, Fair, and Good labels, respectively. (2) The **exponential score** is computed by respectively using 0, 1, and 3 as the weights for the Bad, Fair, and Good labels. This is similar to the gain function in the NDCG formula (i.e., $2^w - 1$) which favors the Good label. Therefore, the linear and exponential scores are bounded by $[0, 2]$ and $[0, 3]$, respectively. We compute these scores for specificity as well by assigning higher weights to more specific questions. Statistically significant differences of these scores were computed using the two-tailed paired t-test with Bonferroni correction at a 95% confidence level.

Table 2: Clarifying questions quality by query type.

Model	Head & Many Reform.			Head & Few Reform.		
	Good	Fair	Bad	Good	Fair	Bad
RTC	30.6%	67.8%	1.6%	20.6%	79.4%	0.0%
QLM	37.1%	62.9%	0.0%	19.0%	79.4%	1.6%
QCM	48.4%	48.4%	3.2%	39.7%	60.3%	0.0%

Model	Tail & Many Reform.			Tail & Few Reform.		
	Good	Fair	Bad	Good	Fair	Bad
RTC	28.6%	68.2%	3.2%	30.6%	69.4%	0.0%
QLM	33.3%	66.7%	0.0%	41.9%	56.5%	1.6%
QCM	25.4%	74.6%	0.0%	38.7%	58.1%	3.2%

According to Table 1, QCM outperforms QLM which performs better than the RTC model in terms of both quality scores. The results also show that all the models barely generate Bad questions, and QCM (the best model) generates Good questions in 38% of the times. It is interesting that QCM also generates more specific questions compared to the first two models. Note that generating specific good questions is important. From these results, one may think that annotators tend to give better scores to specific questions. To validate this hypothesis, we compute the correlation between the question quality and specificity. The results in terms of Pearson's ρ and Kendall's τ are reported in the last two columns of Table 1. According to the results, the quality of all models have negative correlations with the specificity score, meaning that specific questions tend to have lower quality. The reason is that generating "good" specific questions is difficult, while general questions often get Fair or Good labels. The results show that QCM has the least negative quality-specificity correlation. This means that QCM generates higher quality specific questions compared to the other methods. The results suggest that both specificity and quality should be computed for evaluating clarifying questions.

For a deeper analysis, we provide the results for each query type in Table 2. Note that the evaluation queries are uniformly distributed across these four query types (see Section 6.1 for more information). According to the table, QCM substantially outperforms the other models for the head queries. In case of tail queries with many reformulations, QLM performs much better than the other models. The reason is that in case of many reformulation, it is more difficult to compute a proper reward function. The highest improvement is obtained for head queries with few reformulations.

6.5 Evaluating Candidate Answer Set

In the next experiment, we ask human annotators to evaluate the quality of candidate answer set. To do so, we present the query, the clarifying question, and the candidate answer set (up to five candidate answers) to the user. Note that the annotator has already reviewed multiple pages of the result list to understand different possible intents behind the query. We ask the annotators to evaluate the candidate answer set in terms of usefulness for clarification, comprehensiveness, coverage, understandability, grammar, diversity, and importance order. We give them clear definition of each label. To summarize, they are asked to assign the Good label, if the candidate answer set satisfies all the mentioned aspects. While, the Fair label should be given to an acceptable (with production standards) candidate answer set that does not satisfy at least one of the above criteria. Otherwise, the Bad label should be chosen.

Table 3: Candidate answer set evaluation. †/‡ denote significant improvements compared to MLE/MMR, respectively.

Model	Good	Fair	Bad	score (linear)	score (exp)
MLE	0.4%	82.4%	17.2%	0.832	0.836
MMR	0.4%	94.4%	5.2%	0.952†	0.956†
CAS	4.8%	87.2%	8.0%	**0.968**†	**1.016**†‡

Table 4: Secondary result page evaluation. †/‡ denote significant improvements compared to MLE/MMR, respectively.

Model	Good	Fair	Bad	score (linear)	score (exp)
MLE	69.9%	18.0%	12.1%	1.578	2.278
MMR	66.8%	24.1%	9.1%	1.578	2.246
CAS	87.3%	6.0%	6.7%	**1.806**†‡	**2.678**†‡

Although there is no existing model for generating candidate answers for clarifying questions, we considered the following natural baselines to evaluate our candidate answer selection model:
MLE: This model is based on maximum likelihood estimation over the output of neural collaborative filtering (see Section 5.1). In fact, this model is similar to query auto-completion models that try to generate the next phrases for a query. To do so, we order all possible candidate answers based on the reformulation probability $p(q \longrightarrow q')$ and choose the top m phrases with highest probabilities. This model is similar to the BPMF model in [10].
MMR: This is a model based on maximal marginal relevance [11] that finds the candidate answers that are relevant to the query (based on the reformulation probabilities $p(q \longrightarrow q')$) and far from each other (based on word embedding similarity). The intuition is to produce a diverse set of candidate answers, similar to [10].

The results are presented in Table 3.[2] The linear and exponential scores and the significance tests are computed in the same way as question evaluation. According to the table, most candidate answer sets are given a Fair label. The reason is that covering all possible intents for each query with maximum five options is difficult or even impossible in many cases. Note that the Fair label in our guideline still meets the production standards, thus the obtained results are considered as a successful candidate answer set for production, but not optimal. The results show that CAS and MMR significantly outperform MLE. The reason is that both CAS and MMR have a notion of diversity which helps them increase intent coverage. Among the tested models CAS performs the best.

6.6 Secondary Search Result Evaluation

When the user selects a candidate answer, we revise the search results by concatenating the initial query and the candidate answer and submitting the obtained query to the search engine. This is called secondary search result page. Our user study suggests that disappointing secondary search results degrade the satisfaction of users from the clarification pane (see Section 3.3 for more detail). Motivated by this observation, in the next experiment we ask the annotators to click on each individual candidate answer and evaluate the SERP quality (including result list, answers, etc). A

[2]For the sake of space, the results by query type is not reported. They are mainly uniformly distributed across different query types.

Table 5: Secondary result page evaluation by query type.

Model	Head & Many Reform.			Head & Few Reform.		
	Good	Fair	Bad	Good	Fair	Bad
MLE	80.3%	11.9%	7.8%	53.7%	24.1%	22.2%
MMR	75.7%	17.8%	6.5%	55.3%	29.4%	15.3%
CAS	93.7%	2.0%	4.3%	80.5%	10.6%	8.9%

Model	Tail & Many Reform.			Tail & Few Reform.		
	Good	Fair	Bad	Good	Fair	Bad
MLE	78.7%	13.7%	7.6%	67.1%	22.3%	10.6%
MMR	70.4 %	22.6%	7.0%	66.0%	26.5%	7.4%
CAS	89.4%	4.7%	5.9%	85.4%	6.8%	7.8%

secondary result page is considered as Good, if the answer to all possible information needs behind 'query + selection option' can be *easily* found in a prominent location in the page (e.g., an answer box on top of the page or the top three retrieved documents) and the retrieved information *correctly* satisfies the possible information needs. If the result page is still useful, but finding the answer is not easy or is not on top of the page, the Fair label should be chosen. Otherwise, the secondary page is Bad.

In this experiment, we use the same baselines as those introduced in Section 6.5 (i.e., MLE and MMR). The results are reported in Table 4. According to the results, most secondary result pages are labeled as good for all models. CAS substantially outperforms the other two models; over 87% of secondary result pages obtained by CAS are labeled as Good. It is interesting that while both MMR and CAS diversify the candidate answer list, the quality of candidate answers generated by these two models are substantially different.

Table 5 reports the results per query types. According to the table, queries with many reformulations generally have higher quality candidate answers. The reason is that the candidate answer selection algorithms have access to several good follow up options.

6.7 Case Study

We report some example outputs of the proposed QCM model in Table 6, which uses CAS for candidate answer selection. For the example "rytary" the model generates a question about this medication and the candidate answer set consists of its different aspects. The query "acts 17:16" is an example of tail query in our data, and the generated candidate answers are different bible translations. By observing a couple of these translations in the query reformulation data, the model identifies that this is related to bible and should use the information generalized from other bible-related queries. The last example, "alan turing" is a head query with many reformulations. The selected candidate answers clearly demonstrate the power of CAS in selecting diverse options. Although it is not a complete answer set (e.g., the Turing award is missing), the selected options are sensible.

7 LIMITATIONS AND FUTURE DIRECTIONS

This work has several limitations that can be improved in the future:

- **Beyond query logs**: To identify different aspects of each query, we rely on query reformulation data, which is not always available. Going beyond query logs by analyzing the retrieved documents or other information sources (e.g., clicked documents and anchor text) is a desired future direction.

Table 6: Example outputs of the QCM model.

Query	rytary
Question	what do you want to know about this medication?
Options	dosage, coupon, side effects, cost, information
Query	acts 17:16
Question	what bible translation are you looking for?
Options	american standard version, kjv, esv, niv, nlt
Query	that's how i got to memphis
Question	what song information are you looking for?
Options	lyrics, stream, download, artist
Query	alan turing
Question	what do you want to know about this british mathematician?
Options	movie, suicide note, quotes, biography

- **Answer coherency**: Modifying the clarification utility function by considering answer coherency is interesting for future work.
- **Multi-turn conversation**: In this work, we do not study multi-turn interactions, in which the users answer to multiple clarifications. The user behavior and their preference may change in a multi-turn setting which is worth exploring.
- **Personalized and session-aware clarification**: This work does not consider short- and long-term user histories. Generating personalized and session-aware clarification would be important.
- **Utilizing user feedback**: It is important to think ahead on how to collect feedback from users for improve question generation models. It is an important and interesting research question.

8 CONCLUSIONS

In this paper, we introduced and addressed the task of generating clarifying questions for open-domain information seeking systems. We presented a clarification type taxonomy based a large-scale search log analysis. We presented three models for asking clarifying questions: (1) a simple yet effective rule-based template selection and slot filling model, (2) a neural question generation model based on maximizing the likelihood of generating clarifying questions, and (3) a reinforcement learning model that maximizes a clarification utility function. Our machine learning models were trained using weak supervision, where the training clarifying questions were generated using the proposed rule-based model. We also introduced a greedy algorithm for selecting candidate answers in response to a given pair of query and clarifying question. We evaluated our models using human annotation, and showed that the presented reinforcement learning model produces the best results by generating 38% good, 60.4% fair, and 1.6% bad questions independent of the clarifying question. Our proposed solution also achieved 87.3% good secondary result pages by selecting candidate answers.

9 ACKNOWLEDGEMENTS

The authors acknowledge the contributions of Flint Luu and Samantha Neufeld in conducting the user studies presented in Sections 3.1 and 3.2. We also would like to thank Behnam Shahbazi and Bodo von Billerbeck for providing the data used in our experiments. This work would have not been possible without their help.

REFERENCES

[1] Martín Abadi et al. 2015. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. Software available from tensorflow.org.
[2] Mohammad Aliannejadi, Hamed Zamani, Fabio Crestani, and W. Bruce Croft. 2019. Asking Clarifying Questions in Open-Domain Information-Seeking Conversations. In *SIGIR '19* (Paris, France). 475–484.
[3] James Allan. 2004. HARD Track Overview in TREC 2004: High Accuracy Retrieval from Documents. In *TREC '04* (Gaithersburg, Maryland).
[4] Paolo Boldi, Francesco Bonchi, Carlos Castillo, Debora Donato, Aristides Gionis, and Sebastiano Vigna. 2008. The Query-flow Graph: Model and Applications. In *CIKM '08* (Napa Valley, CA, USA). 609–618.
[5] Paolo Boldi, Francesco Bonchi, Carlos Castillo, Debora Donato, and Sebastiano Vigna. 2009. Query Suggestions Using Query-flow Graphs. In *WSCD '09* (Barcelona, Spain). 56–63.
[6] Paolo Boldi, Francesco Bonchi, Carlos Castillo, and Sebastiano Vigna. 2011. Query reformulation mining: models, patterns, and applications. *Inf. Retr.* 14, 3 (2011), 257–289.
[7] Marco De Boni and Suresh Manandhar. 2003. An Analysis of Clarification Dialogue for Question Answering. In *NAACL '03* (Edmonton, Canada). 48–55.
[8] Pavel Braslavski, Denis Savenkov, Eugene Agichtein, and Alina Dubatovka. 2017. What Do You Mean Exactly?: Analyzing Clarification Questions in CQA. In *CHIIR '17* (Oslo, Norway). 345–348.
[9] Fei Cai and Maarten de Rijke. 2016. *A Survey of Query Auto Completion in Information Retrieval.* Now Publishers Inc.
[10] Fei Cai, Ridho Reinanda, and Maarten De Rijke. 2016. Diversifying Query Auto-Completion. *ACM Trans. Inf. Syst.* 34, 4 (2016), 25:1–25:33.
[11] Jaime Carbonell and Jade Goldstein. 1998. The Use of MMR, Diversity-based Reranking for Reordering Documents and Producing Summaries. In *SIGIR '98* (Melbourne, Australia). 335–336.
[12] Konstantina Christakopoulou, Filip Radlinski, and Katja Hofmann. 2016. Towards Conversational Recommender Systems. In *KDD '16* (San Francisco, CA, USA). 815–824.
[13] Anni Coden, Daniel Gruhl, Neal Lewis, and Pablo N. Mendes. 2015. Did you mean A or B? Supporting Clarification Dialog for Entity Disambiguation. In *SumPre '15* (Portoroz, Slovenia).
[14] W. Bruce Croft. 2019. The Importance of Interaction for Information Retrieval. In *SIGIR '19* (Paris, France). 1–2.
[15] J. Shane Culpepper, Fernando Diaz, and Mark D. Smucker. 2018. Research Frontiers in Information Retrieval: Report from the Third Strategic Workshop on Information Retrieval in Lorne (SWIRL 2018). *SIGIR Forum* 52, 1 (2018), 34–90.
[16] Marco De Boni and Suresh Manandhar. 2005. Implementing Clarification Dialogues in Open Domain Question Answering. *Nat. Lang. Eng.* 11, 4 (2005), 343–361.
[17] Mostafa Dehghani, Hamed Zamani, Aliaksei Severyn, Jaap Kamps, and W. Bruce Croft. 2017. Neural Ranking Models with Weak Supervision. In *SIGIR '17* (Shinjuku, Tokyo, Japan). 65–74.
[18] Fernando Diaz. 2016. Pseudo-Query Reformulation. In *ECIR '16* (Padua, Italy). 521–532.
[19] Nan Duan, Duyu Tang, Peng Chen, and Ming Zhou. 2017. Question Generation for Question Answering. In *EMNLP '17* (Copenhagen, Denmark). 866–874.
[20] Anthony Fader, Stephen Soderland, and Oren Etzioni. 2011. Identifying Relations for Open Information Extraction. In *EMNLP '11* (Edinburgh, United Kingdom). 1535–1545.
[21] Ahmed Hassan, Xiaolin Shi, Nick Craswell, and Bill Ramsey. 2013. Beyond clicks: query reformulation as a predictor of search satisfaction. In *CIKM '13* (San Francisco, CA, USA). 2019–2028.
[22] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural Collaborative Filtering. In *WWW '17* (Perth, Australia). 173–182.
[23] Michael Heilman and Noah A. Smith. 2010. Good Question! Statistical Ranking for Question Generation. In *NAACL '10* (Los Angeles, CA, USA). 609–617.
[24] Bernard J. Jansen, Danielle L. Booth, and Amanda Spink. 2009. Patterns of Query Reformulation During Web Searching. *J. Am. Soc. Inf. Sci. Technol.* 60, 7 (2009), 1358–1371.
[25] Johannes Kiesel, Arefeh Bahrami, Benno Stein, Avishek Anand, and Matthias Hagen. 2018. Toward Voice Query Clarification. In *SIGIR '18* (Ann Arbor, MI, USA). 1257–1260.
[26] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *ICLR '15* (San Diego, CA, USA).
[27] Tessa Lau and Eric Horvitz. 1999. Patterns of Search: Analyzing and Modeling Web Query Refinement. In *UM '99*, Judy Kay (Ed.). 119–128.
[28] Zhen Liao, Xinying Song, Yelong Shen, Saekoo Lee, Jianfeng Gao, and Ciya Liao. 2017. Deep Context Modeling for Web Query Entity Disambiguation. In *CIKM '17* (Singapore, Singapore). 1757–1765.
[29] Chang Liu, Jacek Gwizdka, Jingjing Liu, Tao Xu, and Nicholas J. Belkin. 2010. Analysis and Evaluation of Query Reformulations in Different Task Types. In *ASIS&T '10* (Pittsburgh, PA, USA). 17:1–17:10.
[30] Edward Loper and Steven Bird. 2002. NLTK: The Natural Language Toolkit. In *ETMTNLP '02* (Philadelphia, PA, USA). 63–70.
[31] Cheng Luo, Yukun Zheng, Jiaxin Mao, Yiqun Liu, Min Zhang, and Shaoping Ma. 2017. Training Deep Ranking Model with Weak Relevance Labels. In *ADC '17* (Brisbane, Australia). 205–216.
[32] Pont Lurcock, Peter Vlugter, and Alistair Knott. 2004. A framework for utterance disambiguation in dialogue. In *ALTA '04* (Sydney, Australia).
[33] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. In *NeurIPS '13* (Lake Tahoe, CA, USA). 3111–3119.
[34] Bhaskar Mitra. 2015. Exploring Session Context Using Distributed Representations of Queries and Reformulations. In *SIGIR '15* (Santiago, Chile). 3–12.
[35] Yifan Nie, Alessandro Sordoni, and Jian-Yun Nie. 2018. Multi-level Abstraction Convolutional Model with Weak Supervision for Information Retrieval. In *SIGIR '18* (Ann Arbor, MI, USA). 985–988.
[36] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global Vectors for Word Representation. In *EMNLP '14* (Doha, Qatar). 1532–1543.
[37] Chen Qu, Liu Yang, W. Bruce Croft, Johanne R. Trippas, Yongfeng Zhang, and Minghui Qiu. 2018. Analyzing and Characterizing User Intent in Information-seeking Conversations. In *SIGIR '18* (Ann Arbor, MI, USA). 989–992.
[38] Luis Quintano and Irene Pimenta Rodrigues. 2008. Question/Answering Clarification Dialogues. In *MICAI '08* (Atizapán de Zaragoza, Mexico). 155–164.
[39] Filip Radlinski and Nick Craswell. 2017. A Theoretical Framework for Conversational Search. In *CHIIR '17* (Oslo, Norway). 117–126.
[40] Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. 2016. Sequence Level Training with Recurrent Neural Networks. In *ICLR '16* (San Juan, Puerto Rico).
[41] Sudha Rao and Hal Daumé III. 2018. Learning to Ask Good Questions: Ranking Clarification Questions using Neural Expected Value of Perfect Information. In *ACL '18* (Melbourne, Australia). 2737–2746.
[42] Sudha Rao and Hal Daumé III. 2019. Answer-based Adversarial Training for Generating Clarification Questions. In *NAACL '19* (Minneapolis, MN, USA).
[43] Rodrygo L. T. Santos, Craig Macdonald, and Iadh Ounis. 2015. Search Result Diversification. *Found. Trends Inf. Retr.* 9, 1 (2015), 1–90.
[44] M. Schuster and K.K. Paliwal. 1997. Bidirectional Recurrent Neural Networks. *Trans. Sig. Proc.* 45, 11 (1997), 2673–2681.
[45] Svetlana Stoyanchev, Alex Liu, and Julia Hirschberg. 2014. Towards Natural Clarification Questions in Dialogue Systems. In *AISB '14* (London, UK), Vol. 20.
[46] Yueming Sun and Yi Zhang. 2018. Conversational Recommender System. In *SIGIR '18* (Ann Arbor, MI, USA). 235–244.
[47] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to Sequence Learning with Neural Networks. In *NeurIPS '14* (Montreal, Canada). 3104–3112.
[48] Idan Szpektor, Aristides Gionis, and Yoelle Maarek. 2011. Improving Recommendation for Long-tail Queries via Templates. In *WWW '11* (Hyderabad, India). 47–56.
[49] Jan Trienes and Krisztian Balog. 2019. Identifying Unclear Questions in Community Question Answering Websites. In *ECIR '19* (Cologne, Germany). 276–289.
[50] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *NeurIPS '17* (Long Beach, CA, USA). 5998–6008.
[51] Ronald J. Williams. 1992. Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning. *Mach. Learn.* 8, 3-4 (1992), 229–256.
[52] Hui Yang, Dongyi Guan, and Sicong Zhang. 2015. The Query Change Model: Modeling Session Search As a Markov Decision Process. *ACM Trans. Inf. Syst.* 33, 4 (2015), 20:1–20:33.
[53] Liu Yang, Hamed Zamani, Yongfeng Zhang, Jiafeng Guo, and W. Bruce Croft. 2017. Neural Matching Models for Question Retrieval and Next Question Prediction in Conversation. In *NeuIR '17* (Shinjuku, Tokyo, Japan).
[54] Hamed Zamani and W. Bruce Croft. 2016. Estimating Embedding Vectors for Queries. In *ICTIR âĂŹ16* (Newark, DE, USA). 123âĂŞ132.
[55] Hamed Zamani and W. Bruce Croft. 2018. On the Theory of Weak Supervision for Information Retrieval. In *ICTIR '18* (Tianjin, China).
[56] Hamed Zamani, W. Bruce Croft, and J. Shane Culpepper. 2018. Neural Query Performance Prediction using Weak Supervision from Multiple Signals. In *SIGIR '18* (Ann Arbor, MI, USA). 105–114.
[57] Hamed Zamani, Mostafa Dehghani, W. Bruce Croft, Erik Learned-Miller, and Jaap Kamps. 2018. From Neural Re-Ranking to Neural Ranking: Learning a Sparse Representation for Inverted Indexing. In *CIKM '18* (Torino, Italy). 497–506.
[58] Hamed Zamani, Mostafa Dehghani, Fernando Diaz, Hang Li, and Nick Craswell. 2018. SIGIR 2018 Workshop on Learning from Limited or Noisy Data for Information Retrieval. In *SIGIR'18* (Ann Arbor, MI, USA). 1439–1440.
[59] Yongfeng Zhang, Xu Chen, Qingyao Ai, Liu Yang, and W. Bruce Croft. 2018. Towards Conversational Search and Recommendation: System Ask, User Respond. In *CIKM '18* (Torino, Italy). 177–186.
[60] Qingyu Zhou, Nan Yang, Furu Wei, Chuanqi Tan, Hangbo Bao, and Ming Zhou. 2018. Neural Question Generation from Text: A Preliminary Study. In *NLPCC '18* (Hohhot, China). 662–671.