Improving Search Clarification with Structured Information Extracted from Search Results

Ziliang Zhao
Zhicheng Dou*
zhaoziliang@ruc.edu.cn
dou@ruc.edu.cn
Gaoling School of Artificial Intelligence
Renmin University of China
Beijing, China

Yu Guo
yu_guo@ruc.edu.cn
Engineering Research Center of Next-Generation Intelligent Search and Recommendation, Ministry of Education
Renmin University of China
Beijing, China

Zhao Cao
Xiaohua Cheng
caozhao1@huawei.com
chengxiaohua1@huawei.com
Huawei Poisson Lab
Beijing, China

ABSTRACT

Search clarification in conversational search systems exhibits a clarification pane composed of several candidate aspect items and a clarifying question. To generate a pane, existing studies usually rely on unstructured document texts. However, important structured information in search results is not effectively considered, making the generated panes inaccurate in some cases. In this paper, we emphasize the importance of structured information in search results for improving search clarification. We propose enhancing unstructured documents with two kinds of structured information: one is "In-List" relation obtained from HTML list structures, which helps extract groups of high-quality items with abundant parallel information. Another is "Is-A" relation extracted from knowledge bases, which is helpful to generate good questions with explicit prompts. To avoid introducing excessive noises, we design a relation selection process to filter out ineffective relations. We further design a BART-based model for generating clarification panes. The experimental results show that the structured information is good supplement for generating high-quality clarification panes.

CCS CONCEPTS

• **Information systems** → **Search interfaces**.

KEYWORDS

Clarifying Question, Search Clarification, Conversational Search

ACM Reference Format:
Ziliang Zhao, Zhicheng Dou, Yu Guo, Zhao Cao, and Xiaohua Cheng. 2023. Improving Search Clarification with Structured Information Extracted from Search Results. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '23), August 6–10, 2023, Long Beach, CA, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3580305.3599389

*Zhicheng Dou is the corresponding author

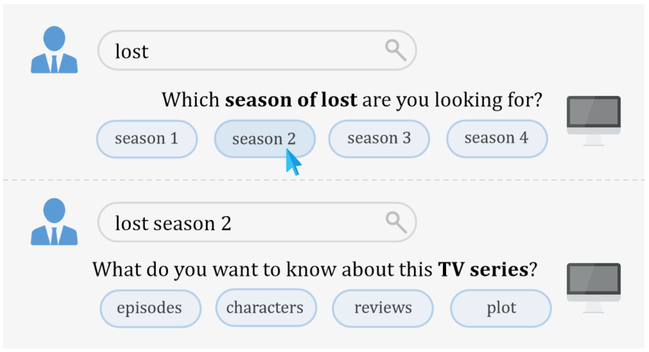
lost
Which **season of lost** are you looking for?
season 1
season 2
season 3
season 4
lost season 2
What do you want to know about this **TV series**?
episodes
characters
reviews
plot

Figure 1: Search clarification in conversational search systems. The quality of clarifying questions and aspect items directly determines the effectiveness of clarification.

1 INTRODUCTION

Search clarification is important for conversational search [29, 38], aiming to understand users' intents by interaction. It is particularly useful for *ambiguous* or *faceted* queries [9]. The basic clarification process is shown in Figure 1. A user first submits a query (i.e., lost, a TV series) to the system. Since the query is faceted, the system will return a search clarification pane which includes several clickable *aspect items* representing sub-intents or sub-topics of the query (i.e., each season of the TV series), together with a proactive *clarifying question* (i.e., Which season of lost are you looking for?) to help the user clarify her intent. When a user clicks one of the provided items, the query will be refined accordingly and the clarification process can be continued until the query is no longer ambiguous or faceted. In such a search clarification system, both the items and the question are important: high-quality items will help users identify their intents quickly, and a good question will improve the intelligence and trustworthiness of the response.

In order to generate aspect items and clarifying questions, researchers have extensively utilized **retrieved document texts** (or snippets) as the main resource. For example, to generate high-quality aspect items, Dou et al. proposed a rule-based model [7] to extract query facets from list structures in top HTML documents. Recently, Hashemi et al. [10, 11] learned multiple representations of a query and generate aspect items based on the query and top retrieved snippets. As for clarifying question generation, Wang and Li [39] studied question template selection and slot filtering in a multi-task manner utilizing snippets as the input. Zhao et al. [49] utilized abundant description information in top retrieved documents to generate more readable and informative questions. Compared with other resources like the query log [45], search results (or retrieved documents) are publicly available and contain more contextual information about the query, which is deemed crucial to generate high-quality aspect items and clarifying questions.

Although existing methods leveraging search snippets (as the retrieved documents) have been able to generate high-quality items and questions in most cases [10, 11, 30, 39], besides **unstructured information** like snippets, search results also contain abundant **structured information**, which is less considered. In fact, both aspect items and clarifying questions have a strong correlation with the structured information. First, for **aspect items**, they are organized in the form of a list [7], which is very similar to some list-structured information in search results. For instance, the HTML source of the query "lost" contain several drop-down lists that exhibit all seasons of the TV series, which may overlap with ground-truth items, while unstructured texts do not contain this kind of explicit information. Second, for **clarifying questions**, they are largely based on the *descriptions* [45] of the query and the items. Therefore, **integrating such "Is-A" description into the text will also help generate better questions**. For instance in Figure 1, episodes, characters, reviews, and plot are four kinds of information of the query "lost season 2". A good question can be "What do you want to know about this TV series?". However, it is challenging to extract "TV series" from unstructured snippets without any other guidance. To make up for this deficiency, we can combine search result contexts and external "Is-A" like knowledge bases to know that "TV series" is a good description of the query, which can then be applied to generate high-quality questions.

Based on the observations above, we propose extracting **two kinds of relations: "In-List" relation and "Is-A" relation** and integrating them into plain texts as supplements for improving search clarification. The two kinds of relations are illustrated in Figure 2. Among them, the "In-List"relation integrates list-structured information extracted from search results into plain texts by explicitly denoting which term in plain texts appears in the extracted lists. The "Is-A" relation extracted from knowledge bases (such as Concept Graph [42, 43]) adds the descriptions of the query and items explicitly into plain texts to promote the model to generate more accurate clarifying questions. As shown in Figure 2, after leveraging these two kinds of relations, the unstructured document text is extended into a structured tree. A branch starting from a term denotes which list the term is contained in, and/or which "Is-A" description can describe the term appropriately.

On the other hand, the extracted relations may contain excessive noises like unrelated lists and descriptions. To ensure the quality of extracted relations, we first rank and select important "In-List" relations using human-designed features including frequency, semantic similarity, and common-occur information, and then filter "Is-A" relations using the search result contexts. To avoid introducing knowledge noises [22], we modify a BART [21] encoder with two-granularity visible matrices: (1) **Intra-document matrix**: following the existing study [22], only tokens on the same branch in a document can see each other; (2) **Cross-document matrix**: for different search result documents, only those containing the same "In-List" or "Is-A" relations can see each other, so as to prevent too much ineffective attention between unrelated documents.

We use the MIMICS dataset [46] to train and evaluate our proposed methods. The experimental results show that, compared with existing models for either generating items or questions, our proposed method performs better in various metrics. In addition, although our proposed method can independently generate high-quality aspect items or clarifying questions, they may indicate different meanings in some cases, which lacks unity and influences the user experience. Therefore, we also try to generate a question and several items at the decoder side of the model simultaneously, which can directly deliver a high-quality integrated clarification pane to users. We further conduct ablation experiments which show that our proposed components, such as "In-List" and "Is-A" relation extraction, relation selection, and multi-granularity visible matrix, all have beneficial impacts on the experimental results, which proves the effectiveness of the motivation of each module.

Overall, our contributions include:

- To our best knowledge, we are the first trying to incorporate structured information for search clarification generation. This idea can also be extended to some other IR tasks.
- We design a process to mine structured relations from search results together with an end-to-end model to integrate structured relations with unstructured plain text.
- Experimental results demonstrate the effectiveness of our idea and method for generating clarifying questions and aspect items for search clarification.

2 RELATED WORK

Clarifying Question Generation. Open-domain search clarification was originally proposed for conversational search systems [2–4, 32]. When the user's intent is ambiguous, the system proactively asks a clarifying question to help the user re-articulate its intent [9, 20]. Compared with conversational search systems, an information retrieval (IR) system can also clarify user intents by asking questions. Zamani et al. [45] defined the clarification in IR systems that, when a user submits a query, the system generates several items for clicking together with a question displayed to the user, shown in Figure 1. They then analyzed interactions between users and the search engine to improve clarification quality [47]. They also built a dataset MIMICS [46] based on the query log of Bing search engine. Recently, Wang et al. [39] and Zhao et al. [49] emphasized the importance of search results for high-quality question generation. However, all of the work does not consider combining structured information with unstructured information for clarifying question generation. We argue that unused structured information in search results can improve unstructured retrieved document texts to generate high-quality questions.

Aspect Items Generation. The aspect items can be expressed in different forms. For example, query suggestion (or query auto-completion) [1, 14, 16, 24, 25, 34, 41] suggests one or more queries that are likely to be entered. Query subtopics mining [6, 13, 17, 33, 40] is a technology often used in search result diversification. Query facets mining [7, 8, 15, 18, 19] aims to find *multiple groups* of phrases that explain the underlying query facets. In search clarification [45, 46], aspect items are defined as a group of highly-related candidate terms, usually revealing an important aspect of the query. Recently, Hashemi et al. [10, 11] mined aspect items by learning multiple representations of a query to improve their quality. In fact, aspect items can be treated as a list, which can correspond to much list-structured information contained in HTML documents. Therefore, these list structures should be important for items generation, which is less considered in existing studies. Aspect items can also be generated with clarifying questions simultaneously [45] to provide mutual information for generation.

Language Modeling by Leveraging Structured Information. Pre-trained language model (PLM) [5, 27, 28, 37] has become an important research topic [23, 26]. However, most PLMs cannot integrate structured information like knowledge into languages. This makes PLMs insensitive to model structured information. To integrate knowledge into PLMs [12, 35], ERNIE [36, 48] applied entity-level masking to promote the model to learn more accurate entity-related knowledge in the pre-training stage. K-BERT [22] modified the visible matrix when fine-tuning to avoid introducing excessive knowledge noises. In this paper, we aim to integrate structured relations into retrieved snippets as supplements for fine-tuning a PLM, similar to K-BERT [22], to generate better items and questions. Differently, we need to supply two types of structured relations rather than a single type. Besides, since one query can correspond to several retrieved documents (snippets), we need to consider modifying not only the visibility of tokens within a document, but also the visibility of tokens between documents.

3 METHODS

3.1 Structured Information in Search Results

Top retrieved documents have been applied to *generate* both aspect items S [10] and clarifying questions Q [39]. Existing work usually utilizes search snippets as the retrieved documents D, and represents the query q and each document d_i by concatenating them with a special token ([SEP]) like "q [SEP] d_1 [SEP] d_2 [SEP] ... [SEP] $d_{|D|}$", or encode them respectively to obtain their representations. However, no matter what encoding method is selected, structured information contained in search results is ignored, making the generated items and questions inaccurate in some cases.

(1) For aspect items S, they are usually organized as a list structure in HTML documents ("In-List") [7]. We deem it helpful to generate S by explicitly representing these structures as introduced in Section 1. Without their guidance, the model will be difficult to summarize structured lists from unstructured documents, thereby will reduce the quality of generated items.

(2) For clarifying question Q, a large proportion of Q rely on the "Is-A" descriptions of the query q or items S [45, 46, 49]. For example in Figure 1, "season of lost" is the description of S: [season 1, season 2, season 3, season 4], so a good question can be "Which *season of lost* are you looking for?". Without any information guidance, the model can be easily confused by various terms in plain texts. In this situation, the model tends to generate a large number of *generic questions* like "Select one to refine your search", which brings a negative impact on the user experience.

(3) The "Is-A" relation can help generate not only clarifying questions, but also aspect items, especially for ambiguous queries. For example, the query "apple" can have corresponding ambiguous items such as "apple company", "apple fruit", and "apple film". For such queries, understanding their various potential "Is-A" relations helps identify their ambiguous items. On the contrary, we deem that **the "In-List" relation also helps generate better clarifying questions**, because list structures introduce potential description candidates for items into the plain texts.

Given the effectiveness of the "In-List" and "Is-A" relations in search results, it is crucial to integrate them into the generation process of aspect items and clarifying questions. In the following parts of this section, we first introduce the overall framework of our proposed method, then thoroughly explain each component in the framework one by one.

3.2 Framework

To integrate the extracted "In-List" relation and "Is-A" relation for generating aspect items S and clarifying questions Q, we propose a framework shown in Figure 2. First, the user inputs a query q ("google chrome exe") into the search engine to get the retrieved documents $D = \{d_1, d_2, ..., d_{|D|}\}$. In this paper, we use the snippets as D to restrict document lengths, which is consistent with previous work [10, 11, 30, 39]. Then, we implement an algorithm to get list structures $L = \{l_1, l_2, ..., l_{|L|}\}$ in the HTML documents. Next, we inject the "In-List" and "Is-A" relations into D to expand plain snippet texts into tree structures. To avoid integrating unnecessary relations, we conduct a relation selection process, so that only important relations for generating S or Q are retained, and useless relations are removed. Finally, we apply BART [21] to model the tree structure for end-to-end training and generation. As the original attention matrix of BART does not consider the tree structure, following K-BERT [22], we modify the visible matrix in two granularity to avoid introducing knowledge noises.

3.3 Obtaining Search Result Documents

We obtain the search results returned by the Bing's Web Search API for all queries in the MIMICS datasets.[1] For each query, we get top-k snippets as $D = \{d_1, d_2, ..., d_k\}$, where $k \leqslant 10$. In addition, we also crawl the HTML source file corresponding to each search result URL for further structured information extraction.

3.4 Obtaining List Structures

As mentioned in Section 3.1, list structures in HTML usually contain important structured information. For example, the navigation bar of shopping websites facilitates users to search for goods from different perspectives, and some important attributes in Wikipedia are usually listed in the form of list-structured tables. Unlike plain texts, extracting list structures contained in search results is challenging, because the lists can be organized in different forms of HTML codes, while plain texts in search results are more simple and more convenient to be extracted. To obtain list structures from HTMLs, we implement an effective algorithm [7]. This algorithm extracts list structures from different resources like HTML list tags, tables, and repeat regions using human-designed rules. For each user query q, the algorithm parses each downloaded HTML file and gets corresponding lists. The lists are then gathered as one total

[1]https://github.com/microsoft/MIMICS

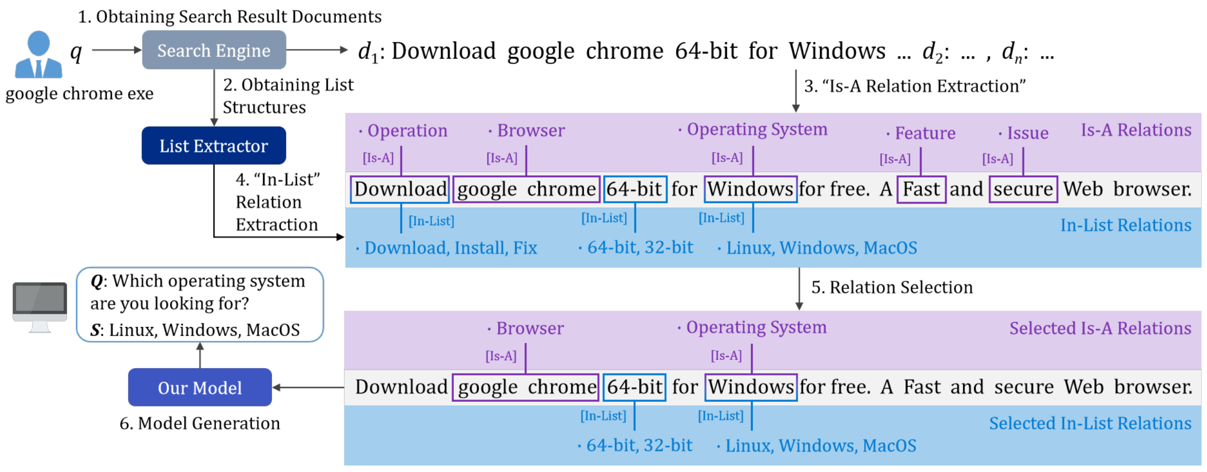
1. Obtaining Search Result Documents
q
google chrome exe
Search Engine
d_1: Download google chrome 64-bit for Windows ... d_2: ... , d_n: ...
2. Obtaining List Structures
3. "Is-A Relation Extraction"
List Extractor
4. "In-List" Relation Extraction
· Operation · Browser · Operating System · Feature · Issue
Is-A Relations
[Is-A]
Download google chrome 64-bit for Windows for free. A Fast and secure Web browser.
[In-List]
In-List Relations
· Download, Install, Fix · 64-bit, 32-bit · Linux, Windows, MacOS
Q: Which operating system are you looking for?
S: Linux, Windows, MacOS
5. Relation Selection
Selected Is-A Relations
Our Model
6. Model Generation
Selected In-List Relations

Figure 2: An overview of our proposed methods for generating aspect items S and clarifying questions Q.

list set $L = \{l_1, l_2, ..., l_{|L|}\}$, where each list is composed of several terms $l_i = \{t_1, t_2, ..., t_{|l_i|}\}$. Since the extracted original lists have excessive noise, we further design a relation selection process to cluster similar lists, select important lists, and delete useless lists and list terms. See Section 3.7 for details.

3.5 "In-List" Relation Extraction

We define the "In-List" relation as a triple $(t, \textit{In-List}, l)$, where t is a term (n-gram) in retrieved document texts (snippets) and l is a list extracted from search results using the method mentioned in Section 3.4. The extraction process can be effectively finished by matching each snippet term with each list term. The "In-List" relation indicates that **a term in plain texts occurs in a list structure extracted from HTML source**. This kind of relation is beneficial for finding aspect items of a query, because items are usually organized as a list structure, and the extracted lists are likely to contain potential high-quality items. For example, for the query "watches", the term "Omega" in snippets is in an extracted list [Omega, Casio, Citizen, Rolex, Cartier]. By incorporating these list-structured relations into search snippets, information about other watch brands can be integrated to enhance the representation of the original term "Omega", so these brands are more likely to be treated as a group of high-quality items.

3.6 "Is-A" Relation Extraction

The "Is-A" relation is usually used to find the hypernyms of a term, so as to explicitly indicate the category or description of an entity. Like the "In-List" relation, the "Is-A" relation can be formalized as a triplet $(t, \textit{Is-A}, h)$, where t represents a term, and h represents its hypernym. For example, when t is "apple", then h can be "company" or "fruit". "Is-A" relation is helpful for generating high-quality informative clarifying questions [39, 45, 49] because it denotes the description of q or S explicitly. In order to label the search snippets with "Is-A" tags, we first need to obtain the knowledge base with "Is-A" relations. We choose Concept Graph [42, 43] and WebIsA [31], two commonly used knowledge bases containing large-scale "Is-A" relations. Then, together with "In-List" relation, we also integrate "Is-A" relation into plain texts simultaneously. By integrating these two kinds of relations, the plain retrieved document texts can be extended as tree-shaped data shown in Figure 2.

3.7 Relation Selection

When all the extracted "In-List" relations and "Is-A" relations are simply taken into account, it will produce excessive noises, that is, the introduction of many irrelevant relations will have no effect or even have a negative effect on the generation of items and questions. For example, in Figure 2, the "Is-A" relation ("download" is an "operation") does not help generate clarifying questions, and the list [download, install, fix] does not help generate high-quality items for this case. Therefore, we believe that these unimportant relations should be removed explicitly to avoid negative effects. To this end, we design methods for the "In-List" relations and "Is-A" relations respectively. The relation selection process can also limit the length of the enhanced snippets, which is convenient to use a mainstream self-attention-based encoder for representation.

3.7.1 "In-List" Relation Selection. For the "In-List" relation, the extracted list set L can be analyzed from two perspectives: (1) **A large number of items between some pairs of extracted lists may be duplicated**. For example, l_1: [Windows 7, Windows 10, Windows XP] and l_2: [Windows 10, Windows XP, Android] both contain "Windows 10" and "Windows XP", so they both represent a similar meaning potentially. For this circumstance, it is necessary to cluster these multiple lists into a single list for removing duplication. (2) **In the clustering process, some items will appear frequently, while some other items only appear occasionally, so their importance will be different**. In the above example, "Windows 10" and "Windows XP" are more important because they appear more frequently. Low-frequency terms like "Android" are deemed to be noisy, which should be removed.

Therefore, for the first perspective, we apply a list clustering algorithm [7] to cluster similar lists in L into a single list. For the second perspective, we design an **importance score** $c(l)$ for each list l in the clustered list set. This score measures the importance of each list and each term in a list, which is used for ranking and selecting lists and terms for the follow-up generation. To consider the importance of a list from various perspectives, we define the importance score as the product of four kinds of features: list frequency feature $f_l(l)$, document frequency feature $f_d(l)$, semantic feature $f_s(l)$, and common feature $f_c(l)$: $c(l) = f_l(l) \times f_d(l) \times f_s(l) \times f_c(l)$. Finally, we select top-x list structures in L as the "In-List" relation pool. Besides, for each list, we only keep top-y terms with the highest frequency to filter out irrelevant and unnecessary terms. Above, x and y are two hyper-parameters to be determined. The definitions of the above four features are illustrated as follows:

- List Frequency Feature f_l. Terms with higher *frequency in lists* indicate that they occupy a higher proportion in the structured information of search results. We define the list frequency feature of a list as the summation of the frequency of all terms in the list:

$$f_l(l) = p_l \cdot \tanh(k_l \cdot \sum_i N_L(t_i)) \tag{1}$$

where $N_L(t_i)$ is the frequency of term t_i. p and k are two parameters controlling the importance and scale. The tanh() function is to scale the range of the feature, which is the same as below.

- Document Frequency Feature f_d. In addition to lists, terms appearing in the *texts* are also related to the user query. Therefore, we also calculate the document frequency of a list as the summation of the frequency of all terms occurring in the HTML texts:

$$f_d(l) = p_d \cdot \tanh(k_d \cdot \sum_i N_D(T_i)) \tag{2}$$

where $N_D(t_i)$ denotes the frequency that t_i occurs in HTML texts.

- Semantic Feature f_s. High-quality items usually have a strong semantic correlation [46]. Therefore, we apply a pre-trained BERT-base model to obtain the vector representation of each term, then calculate their average cosine similarity:

$$f_s(l) = p_s \cdot \tanh(k_s \cdot \frac{1}{|l|^2} \sum_i \sum_j \mathrm{sim}(\mathrm{BERT}(t_i), \mathrm{BERT}(t_j))) \tag{3}$$

where BERT() is a BERT model, sim() is cosine similarity function.

- Common Feature f_c. Some uncommon terms could have a negative influence on the quality of extracted lists, so we need to keep terms that appear frequently on the whole. To achieve this, we go through MIMICS-Manual [46] dataset to get a term-frequency dictionary $F[\cdot]$, then calculate the common feature as:

$$f_s(l) = p_c \cdot \tanh(k_c \cdot \frac{1}{|l|} \sum_i F[t_i]) \tag{4}$$

3.7.2 "Is-A" Relation Selection. For the "Is-A" relation, only descriptions of the query q and items S can help generate clarifying questions [45]. On the other hand, if a large number of irrelevant terms in documents D are labeled with "Is-A" relations, many irrelevant noises will be introduced. Therefore, we make the following two assumptions to ensure the effectiveness of the extracted "Is-A" relations: (1) We do not label all terms with "Is-A" relations, but only terms that *overlap* with the **query** q or terms that are **marked with "In-List" relations**. The former premise is to integrate the description of query q, and the latter is to integrate the description of potential items S, so as to generate better questions Q. (2) Besides, the extracted "Is-A" descriptions can be sometimes ambiguous. For example, when a user search for Apple's related electronic products, labeling "Apple is a fruit" will not provide any useful information. Therefore, we ensure that the "Is-A" descriptions should **occur in retrieved document texts** to ensure the correlation. In this situation, "Apple is a company" is more likely to be extracted, which provides additional information for question generation.

3.7.3 Relation Re-sampling. In addition, since the selected relations may still be numerous, and a large number of repeated relations can cause redundancy, we further sample the extracted relations so that the total length of these relation texts does not exceed 20% of documents D, which maintains a balance of the length between plain texts and relation-based texts, so that the semantic information in the text will not be dominated by these relations.

3.8 BART-based Generation Model

After relation extraction and selection, original snippet texts have been converted into relation-enhanced tree structures shown in Figure 2. It is important to model the tree structures using a proper network structure for the follow-up generation. To encode these structured data, K-BERT [22] injects knowledge triples directly after the entity. However, introducing too much knowledge into the texts may influence the meaning of the original sentence, which is called knowledge noise problem. Therefore, K-BERT adjusts the visible matrix, so that the embedding of a word only comes from the context of the same branch in a tree-structured text, and the words of different branches do not attend to each other. Different from K-BERT, we need to consider not only the visibility within a document, but also the visibility among different documents.

To model the relation-enhanced tree-structured data and then generate aspect items and/or clarifying questions, we propose a BART-based language model composed of an encoder and a decoder for sequence-to-sequence learning. The decoder is consistent with that of the original BART for generation. The encoder mainly models the visibility of each token in the input data with two levels of granularity visible matrix: **intra-document matrix** and **cross-document matrix**. The two kinds of matrices are then concatenated together as the whole visible matrix. The illustration of our proposed model is shown in Figure 3, which takes "Download google chrome 64-bit for Windows. Fast and secure browser." as an example of retrieved document text (snippet).

3.8.1 Intra-document Matrix. Similar to K-BERT [22], in a single retrieved document (snippet), a token can only attend to other tokens that are on the same branch. In addition, if a token is simultaneously marked with "In-List" and "Is-A" relation, we let its corresponding "In-List" relation and "Is-A" relation attend to each other, so as to integrate the co-occurrence information of the two relations, as the token "Windows" in Figure 3. We define the visible matrix within a single snippet as the intra-document matrix I:

$$I_{ij} = \begin{cases} 0 & w_i \ominus w_j \text{ or } w_i \oplus w_j \\ -\infty & w_i \oslash w_j \end{cases} \tag{5}$$

where i and j are hard indexes of each token. $w_i \ominus w_j$ means that the two tokens are in the same branch, and $w_i \oslash w_j$ means they are not. $w_i \oplus w_j$ indicates that the two tokens are in "In-List" and "Is-A" relations generated by the same token respectively.

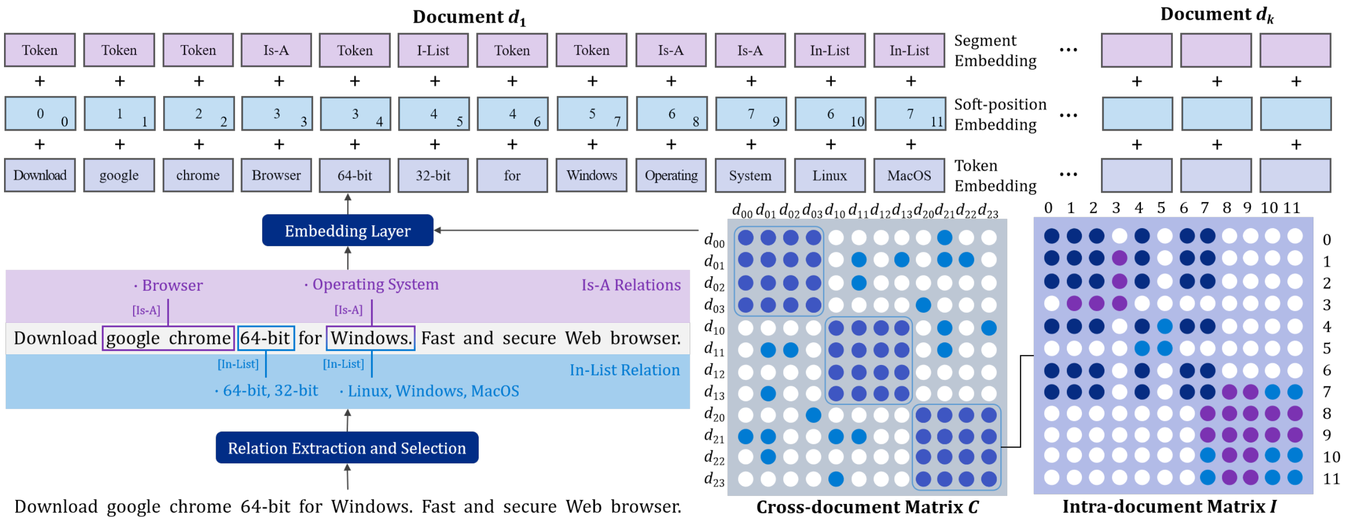

Figure 3: The modified BART encoder. The retrieved document texts are first enhanced with relations extracted from search result HTML files. They are then encoded with a BART encoder with the improved visible matrix.

3.8.2 Cross-document Matrix. The input data of our model is composed of top-k relation-enhanced snippets. When encoding the data, different documents can fully pay attention to each other, which introduces unnecessary noise [22]. One way is that each document can only focus on itself, but this ignores the interactive information between documents. For example, d_1 mentions the symptoms and treatment of headache, while d_2 mentions its treatment and causes. The two documents need to attend to each other to make the model understand "headache" more comprehensively. Therefore, we redesign the attention across different documents. For two documents d_i and d_j, if they contain the same "In-List" or "Is-A" relation, the visibility between their texts should be built. This lead to $|D|^2 - |D|$ cross-document matrices $C^{i,j}$, where $1 \leqslant i, j \leqslant |D|$. We finally combine all intra-document matrix I and cross-document matrix C to construct the final visible matrix $\mathcal{M}$ of the BART encoder:

$$\mathcal{M} = \begin{bmatrix} I^1 & C^{1,2} & \cdots & C^{1,k} \\ C^{2,1} & I^2 & \cdots & C^{2,k} \\ \vdots & \vdots & \ddots & \vdots \\ C^{k,1} & C^{k,2} & \cdots & C^{k,k} \end{bmatrix} \tag{6}$$

The self-attention mechanism can then be redefined with:

$$\boldsymbol{h} = \mathrm{softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^T + \mathcal{M}}{\sqrt{d_k}}\right)\boldsymbol{V} \tag{7}$$

Besides the visibility, we further modify the embedding layer of BART to make it easier for the model to distinguish different parts of the input. The original BART [21] embedding layer composed of **Position Embedding** and **Token Embedding**. Both are learnable parameters that encode the position information and the semantics of tokens in the input sequence respectively. However, these two embeddings cannot distinguish whether a term comes from the snippet text or the relations. Therefore, we add the third **Segment Embedding** shown in Figure 3, using different tokens to represent different parts of the input data, which is similar to [5].

4 EXPERIMENTS

In this section, we hope to answer the following three research questions: (1) **RQ1**: Does the extracted structured information help generate better items and questions? (2) **RQ2**: what is the impact of co-generation and generation order of Q and S on the results? (3) **RQ3**: How can each part of our proposed framework affect the experimental results? Among them, RQ1 mainly studies whether our proposed methods can perform better than existing baselines in generating items and questions. RQ2 mainly explores whether the model can generate items and questions at the same time so that it can be applied to generate more integrated clarification panes. RQ3 further explores whether our proposed important components are helpful to the experimental results.

4.1 Dataset

MIMICS [46] is a dataset with over 400k pieces of data, each piece of data is composed of a query, several aspect items, and one clarifying question. We process MIMICS dataset to obtain a set of (q, S, Q), where q indicates the query, S is a set of items, and Q is a clarifying question. We further download the top-ten retrieved snippets provided by MIMICS as the retrieved document set D for each query q, together with their corresponding URLs and HTML files. To stay consistent with previous work [30], for aspect items generation, we use MIMICS-Click data for training and use MIMICS-Manual data with Fair or Good labels for evaluating our proposed model. For clarifying question generation, we use a subset of MIMICS [39] (about 40k) which maintains a balance in the number of different question templates for training and evaluation.

4.2 Evaluation Metrics

For aspect items generation, we apply four sets of evaluation metrics in previous work [10, 11, 30]. (1) **Term overlap** precision, recall, and F1: these three metrics calculate the overlap between the set of generated terms with the ground truth terms, which have been

Table 1: Items generation evaluation results, ablation studies, and co-generation results. Each best evaluation metric is bold. "†" denotes significant improvement compared with the best baseline with p-value < 0.05.

	Term Overlap			Exact Match			Set BLEU				Set BERT-Score		
Model	**Prec**	**Recall**	**F1**	**Prec**	**Recall**	**F1**	**1-gram**	**2-gram**	**3-gram**	**4-gram**	**Prec**	**Recall**	**F1**
QDist	0.1275	0.1108	0.1121	0.0084	0.0103	0.0087	0.2042	0.1752	0.1578	0.1439	0.5185	0.5114	0.5108
QFI	0.1525	0.1840	0.1606	0.0137	0.0162	0.0155	0.2141	0.1845	0.1597	0.1561	0.5113	0.5174	0.5156
QFJ	0.1504	0.1853	0.1584	0.0143	0.0151	0.0144	0.2144	0.1879	0.1591	0.1540	0.5174	0.5208	0.5185
QDMiner	0.1629	0.1879	0.1690	0.0207	0.0278	0.0223	0.2225	0.1970	0.1680	0.1605	0.5118	0.5170	0.5124
NMIR	0.1856	0.1965	0.1905	0.0354	0.0388	0.0370	0.2524	0.2139	0.1906	0.1741	0.5311	0.5368	0.5344
BART-qD	0.2014	0.3084	0.2361	0.0417	0.0655	0.0496	0.2972	0.2339	0.2030	0.1855	0.5361	0.5412	0.5382
BART-q	0.1816	0.2861	0.2161	0.0304	0.0458	0.0355	0.2993	0.2355	0.2032	0.1852	0.5319	0.5401	0.5356
Labeling	0.2075	0.2424	0.2131	0.0588	0.0878	0.0678	0.2236	0.1637	0.1380	0.1260	0.5389	0.5252	0.5314
Classification	0.0608	0.0626	0.0594	0.0254	0.0373	0.0294	0.1227	0.0638	0.0386	0.0305	**0.5455**	0.5182	0.5130
Extraction	0.1748	0.2465	0.1971	0.0149	0.0287	0.0192	0.2915	0.2203	0.1851	0.1669	0.5260	0.5362	0.5307
Our	0.2316†	0.3124	0.2621†	**0.0859**†	**0.0914**†	**0.0820**†	0.3375†	**0.2817**†	0.2590†	0.2322†	0.5416	0.5425	**0.5422**
Our-SQ	**0.2407**†	**0.3144**†	**0.2721**†	0.0798†	0.0815†	0.0804†	**0.3408**†	0.2803†	**0.2592**†	**0.2410**†	0.5413	0.5399	0.5405
Our-QS	0.2328†	0.3029	0.2639†	0.0820†	0.0874†	0.0839†	0.3288†	0.2753†	0.2520†	0.2228†	0.5387	**0.5433**	0.5406
w/o. "In-List"	0.2091	0.2928	0.2408	0.0688	0.0723	0.0705	0.3033	0.2457	0.2121	0.1896	0.5397	0.5402	0.5399
w/o. "Is-A"	0.2312	0.3068	0.2630	0.0848	0.0903	0.0870	0.3343	0.2772	0.2511	0.2259	0.5408	0.5427	0.5414
w/o. Selection	0.1823	0.2799	0.2193	0.0382	0.0462	0.0412	0.2779	0.2286	0.1951	0.1664	0.5266	0.5302	0.5285
w/o. $\mathcal{M}$	0.2155	0.2374	0.2245	0.0519	0.0669	0.0581	0.3083	0.2525	0.2192	0.1868	0.5328	0.5337	0.5331

previously used for evaluating query facets mining [18]. (2) **Exact match** precision, recall, and F1: these three metrics are more strict than term overlap metrics. They only measure the condition when the generated items are the same as ground truth items. (3) **Set BLEU**: BLEU is usually implemented to measure the n-gram similarity between a piece of candidate text and a set of reference texts. It has been widely used for evaluating all kinds of text generation tasks. We implement the Set BLEU score from 1-gram to 4-gram so that these metrics can measure the n-gram similarity between two sets. (4) **Set BERT score** precision, recall, and F1: BERT score is widely used to compute semantic similarity between sentences. Similar to the Set BLEU, we also implement the Set BERT-score to evaluate the semantic similarity between two sets of texts.

We also use the four kinds of evaluation metrics defined in previous work [39] to evaluate the quality of clarifying questions. For clarifying question selection (CQS) models and clarifying template selection (CQT) models, we use **accuracy** to measure how many question templates of the generated questions can be matched with that of ground truth questions. We also use **MRR@3** to evaluate the quality of the top-3 ranked question templates by applying a 3-head beam search for generation. For clarifying question generation (CQG) models, we evaluate the results using **BLEU** and **entity-F1**. BLEU calculates the 4-gram overlaps between generated questions and ground truth questions, which evaluates the results from a lexical perspective. Since the most replaced part in a clarifying question is the entity description filled in the question template, we further implement Entity F1 by micro-averaging precision and recall over these entity descriptions to evaluate the results.

4.3 Implementation Details

In the "In-List" relation selection module, all importance parameters p_x and scale parameters k_x are determined by grid search with the step of 0.1 within the range of (0, 1]. Specifically, we first sample 1k held-out queries and their corresponding items. To measure the quality of extracted "In-List" relations, for each parameter combination, we calculate the frequency that the extracted "In-List" terms that occur in the ground-truth items and finally choose the best parameter. This process ensures that the "In-List" relation selection module is effective to mine information about the aspect items. We also ensure that the four features for selecting "In-List" relations used in Section 3.7.1 all have positive effects on the quality of extracted "In-List" relations. For the hyper-parameter x and y mentioned in Section 3.7, they are both set to be 5 intuitively, because a ground-truth items set contains at most five items. All learnable parameters in the BART-based model are initialized with the pre-trained BART-base model. In all the experiments, the batch size is set to 8, the input length is set to 768, and the maximum output length is set to 64. To discriminate different part of the input, following previous work [30], we use a special token "[SEP]" to concatenate user query q and documents D as q [SEP] d_1 [SEP] d_2 [SEP] ... [SEP] d_k. In the decoder part, the model generates a clarifying question Q or generates aspect items one by one as S_1 [SSEP] S_2 [SSEP] ... [SSEP] S_N. We use AdamW optimizer with a learning rate of 2×10^{-5} to optimize the loss function.

4.4 Experimental Results (RQ1)

To answer **RQ1**, we report the experimental results of aspect items generation and clarifying question generation in the upper part of Table 1 and Table 2 respectively and analyze the results.

4.4.1 Aspect Items Generation. For aspect items generation, we compare three groups of baseline methods: (1) QDist [44], QFI, QFJ [18], and QDMiner [7] are four rule-based or machine learning-based methods for query facets mining. They make use of the list-structured information in search results, but do not combine them with unstructured text. (2) NMIR [10], a neural model that learns multiple representations of a query to generate items, but does not utilize structured relations. (3) Four types of PLM-based models [30] to obtain aspect items, including generation, labeling, classification, and extraction. Among these models, we mainly aim to compare the BART generation model (BART-*qD* in Table 1) which uses only the concatenated query and snippet texts as the input for items generation, to prove the effectiveness of our extracted structured relations. Table 1 reports the experimental results.

The first and most important conclusion is that **our proposed model outperforms existing baselines in nearly all evaluation metrics**, including term and exact match metrics, n-gram metrics, and semantic metrics. Furthermore, most of the metrics pass the t-test with p-value < 0.05, denoting the significance of the improvements. The results illustrate that our proposed model is effective to generate more satisfying aspect items compared with other PLM-based models or traditional extraction models. Especially, our model outperforms the original BART model (BART-*qD* in Table 1) significantly, illustrating that **the snippets enhanced by our extracted structured information are more effective to mine and generate aspect items compared with pure snippet texts**. On the other hand, our model has little improvement on the Set-BERT score metric. Compared with the optimal baseline BART-*qD*, the F1-score of this metric has only increased by 0.004, while the precision even decreases slightly. This illustrates that, compared with the original snippet texts, **the structured information hardly brings significant improvement on semantic similarity**, which is different from the former three metrics. However, we argue that the improvement on the Set BERT-core does not completely represent the improvement of the quality of the items. For example, the query "google chrome exe" may correspond to two groups of items: S_1: [32 bit, 64 bit], S_2: [Windows, Linux, MacOS]. These two groups of items have roughly equal Set BERT-score, yet S_1 is the ground truth and S_2 is deemed sub-optimal. To sum up, we deem that the results generated by structured information are significantly better than those generated by plain texts, thus demonstrating the effectiveness of our proposed idea and model.

We also notice that, compared with traditional extraction methods utilizing structured information (including QDist, QFI, QFJ, and QDMiner), neural models utilizing snippets (NMIR, BART, etc.) can gain overwhelming advantages in almost all evaluation metrics. The results demonstrate that, compared with traditional machine learning or human-designed features, **unstructured data containing abundant contextual information like the snippet is also important to mine query intents**. Our proposed model combines the advantages of unstructured snippets and structured data, so it further achieves an effective positive effect on the results.

4.4.2 Clarifying Question Generation. For clarifying question generation, we compare several basic clarifying question selection (CQS), clarifying template selection (CTS), and clarifying question generation (CQG) baselines. We then compare our proposed model with TG-ClariQ [39], which is a multi-task architecture for question template selection and slot filling. We use training and evaluation data consistent with these baselines. As can be seen from Table 2, first, our model shows a slight improvement in the Accuracy and MRR@3 compared with the best baseline, illustrating that our proposed generation model can match the selection model (TG-ClariQ-BERT) in template selection, or even exceed the selection model. However, it is more important and challenging to generate appropriate **descriptions** filled in question templates, which denotes that BLEU and Entity-F1 can better evaluate the

Table 2: Experimental results of question generation. Each best metric is bold. "†" denotes significant improvement compared with the best baseline with p-value < 0.05.

Model	Accuracy	MRR@3	BLEU	Entity-F1
BM25 (CQS)	0.355	0.399	-	0.414
RankNet (CQS)	0.308	0.384	-	0.203
LambdaMART (CQS)	0.490	0.564	-	0.214
BERT (CQS)	0.394	0.440	-	0.356
BM25 (CTS)	0.095	0.191	-	-
RankNet (CTS)	0.323	0.455	-	-
LambdaMART (CTS)	0.564	0.621	-	-
BERT (CTS)	0.676	0.794	-	-
LSTM (CQG)	-	-	45.30	0.166
LSTM+Copy (CQG)	-	-	52.64	0.495
Trm+Copy (CQG)	-	-	55.37	0.546
TG-ClariQ-LSTM	0.659	0.791	55.05	0.682
TG-ClariQ-BERT	0.722	0.827	60.49	0.788
Our	0.734	**0.837**	71.56†	0.835†
Our-SQ	**0.736**	0.831	71.50†	0.837†
Our-QS	0.735	0.834	**71.59**†	**0.839**†
w/o. "In-List"	0.729	0.828	67.83	0.823
w/o. "Is-A"	0.722	0.823	67.94	0.825
w/o. Selection	0.656	0.714	62.75	0.755
w/o. $\mathcal{M}$	0.694	0.781	64.13	0.780

overall quality of the generated questions. As shown in the BLEU and Entity-F1 results, our proposed model significantly outperforms existing best baselines with BLEU rising from 60.49 to 71.56, and Entity-F1 rising from 0.788 to 0.835. These results effectively prove that **the extracted structured information contributes to the generation of high-quality clarifying questions**.

4.5 Co-generation of Items and Questions (RQ2)

In Section 4.4, we focus on evaluating the quality of generated items and questions independently. However, in real-world systems, the **integration** of items S and question Q is also important, which means that the question should accurately clarify these items. However, generating items and questions separately will lose their co-occurrence information, which would generate inconsistent clarification panes. For example, for the query "watches", the generated items are [Omega, Casio, Citizen, Rolex, Cartier], while the generated question is "Who are you shopping for?". This is because the supervised data does not contain the interactive information of Q and S at the same time. To solve this problem, we can easily change the supervised data on the decoder side so that the model can decode clarifying questions and aspect items sequentially. There are two ways to achieve this: generating questions followed by several items like "Q [QSEP] S_1 [SSEP] ... [SSEP] S_k" (denoted as Our-QS), or first generating several items followed by a question like "S_1 [SSEP] ... [SSEP] S_k [QSEP] Q" (denoted as Our-SQ).

To compare with the previously generated items and questions, we still use the evaluation data and metrics mentioned in Section 4.2 to evaluate the generated items and questions. The results are also recorded in Table 1 and Table 2 respectively. The experimental results show that, first, compared with generating items separately, the co-generation results have a little fluctuation in these automatic

Table 3: Human annotation of integrated clarification panes.

Our	Our-QS	Our-SQ	Tie
13.5%	31.0%	32.5%	23.0%

evaluation metrics, yet the impact is not great. For questions, the results show a slight increase in accuracy and entity F1 and a slight decrease in MRR@3. In addition, the order of generating items and questions has little effect on the results. In general, the co-generation of Q and S and the generation order of the two have little influence on the experimental results.

On the other hand, evaluating items and questions separately ignores their consistency. For example, for the query "apple", [company, fruit, film] and [ipad, iPhone, macbook] are both good items set. However, a model may generate the former set of items followed by asking the question "which product of apple are you looking for?", which is more suitable for the latter items set. In this situation, although the generated items and questions are both of high quality, they are inconsistent and are not satisfying to be delivered in real-world conversational search systems. Therefore, we hope to further evaluate the **overall quality** of several items and a question, that is, the quality of a **whole clarification pane**. To achieve this, we randomly sampled 200 queries from the evaluation set. For each query, we generate three panes: (1) Q and S generated by our proposed model separately (Our), (2) Q and S generated by **Our-QS**, (3) Q and S generated by **Our-SQ**. We hire three annotators to select one best pane manually, and the final winner is determined by major voting. In 23% conditions, the three methods generate the same panes, so these results are labeled "Tie". The winning and losing situation is shown in Table 3. In most cases (31.0% for Our-QS and 32.5% for Our-SQ), co-generation models can generate more integrated clarification panes which defeat separate generation results which only win in 13.5% of the data. The result shows that the co-generation paradigm is more suitable to be applied in online search engines for clarification.

4.6 Ablation Studies (RQ3)

Our proposed model is composed of several important components like "In-List" relation, "Is-A" relation, relation selection module, and two-granularity visible matrix. These modules are directly constructed on the BART-base model together. Therefore, it is very important to clarify their separate effect on the results (**RQ2**). To achieve this, we briefly remove the above four modules separately and then recalculate all evaluation metrics for ablation. The results of items and questions are shown in the bottom part of Table 1 and Table 2 starting with "w/o." followed by the module name.

The ablation results show that, after removing the "In-List" relation, items generation showed a serious decline in all evaluation metrics, and clarifying question generation results are also slightly affected. The results are intuitive: we have assumed and demonstrated that the "In-List" relation is mainly helpful for aspect items generation, and the experimental results further prove that it is not only helpful for items generation, but also has an impact on clarifying question generation. Similar to the "In-List" relation, by removing the "Is-A" relation, the evaluation metrics of items generation do not change significantly, while the clarifying question generation showed a certain decline in the four metrics. It shows that the "Is-A" relation mainly has a certain effect on questions, yet has little effect on items. To remove the relation selection module, we directly sample relations from the original extracted lists and "Is-A" descriptions without weighting and selecting them, which could lead to a large number of noises. The results of aspect items and clarifying questions both show a significant decline. Specifically, for items generation, after removing the relation selection process, our proposed model performs even worse than the original BART model for generating items, and most metrics evaluating questions also drop significantly to under-baseline levels. This is because the unfiltered "In-List" and "Is-A" relations have a lot of information that is not related to items and questions, which introduces excessive noise. Therefore, the model input does not contain much useful information, which cannot help generate, but will bring negative effects. We finally change the visible matrix $\mathcal{M}$ into a full-of-zero matrix, which means that each token in the whole input can see each other. This leads to a slight decline for all metrics of aspect items and clarifying questions. This illustrates that the adjustment of the visibility of input data is effective and can reduce the noise caused by the integration of structured relations.

5 CONCLUSION

In this paper, we study generating clarification panes with structured information extracted from search results. We design two kinds of relations: "In-List" relation and "Is-A" relation, as supplements for enhancing unstructured retrieved document texts. We then apply a BART encoder to model the enhanced texts and generate items and/or questions. To filter out noisy relations, we design a rule-based relation selection module to select relations according to their importance score. We further design a two-granularity visibility matrix to integrate the two kinds of relations into the BART encoder. The experimental results on the MIMICS dataset show that our proposed methods can generate clarification panes with higher quality compared with strong baselines on several evaluation metrics. We further try to generate items and questions simultaneously to deliver integrated clarification panes, which is important for online systems. Finally, we conduct ablation studies to prove the usefulness of each of our proposed component.

ACKNOWLEDGMENTS

This work was supported by National Key R&D Program of China No. 2022ZD0120103, National Natural Science Foundation of China No. 62272467, Beijing Outstanding Young Scientist Program No. BJJWZYJH012019100020098, Public Computing Cloud, Renmin University of China, and Intelligent Social Governance Platform, Major Innovation & Planning Interdisciplinary Platform for the "Double-First Class" Initiative, Renmin University of China. The work was partially done at Beijing Key Laboratory of Big Data Management and Analysis Methods, and Key Laboratory of Data Engineering and Knowledge Engineering, MOE.

REFERENCES

[1] Wasi Uddin Ahmad, Kai-Wei Chang, and Hongning Wang. 2018. Multi-task learning for document ranking and query suggestion. In *International Conference on Learning Representations*.

[2] Mohammad Aliannejadi, Julia Kiseleva, Aleksandr Chuklin, et al. 2020. ConvAI3: Generating Clarifying Questions for Open-Domain Dialogue Systems (ClariQ). *arXiv preprint arXiv:2009.11352* (2020).

[3] Mohammad Aliannejadi, Julia Kiseleva, Aleksandr Chuklin, et al. 2021. Building and Evaluating Open-Domain Dialogue Corpora with Clarifying Questions. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. 4473–4484.

[4] Mohammad Aliannejadi, Hamed Zamani, Fabio Crestani, et al. 2019. Asking clarifying questions in open-domain information-seeking conversations. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 475–484.

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, et al. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. 4171–4186.

[6] Zhicheng Dou, Sha Hu, Kun Chen, et al. 2011. Multi-dimensional search result diversification. In *Proceedings of the fourth ACM international conference on Web search and data mining*. 475–484.

[7] Zhicheng Dou, Sha Hu, Yulong Luo, et al. 2011. Finding dimensions for queries. In *Proceedings of the 20th ACM international conference on Information and knowledge management*. 1311–1320.

[8] Zhicheng Dou, Zhengbao Jiang, Sha Hu, et al. 2015. Automatically mining facets for queries from their search results. *IEEE Transactions on knowledge and data engineering* 28, 2 (2015), 385–397.

[9] Helia Hashemi, Hamed Zamani, and W Bruce Croft. 2020. Guided Transformer: Leveraging Multiple External Sources for Representation Learning in Conversational Search. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1131–1140.

[10] Helia Hashemi, Hamed Zamani, and W Bruce Croft. 2021. Learning Multiple Intent Representations for Search Queries. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. 669–679.

[11] Helia Hashemi, Hamed Zamani, and W Bruce Croft. 2022. Stochastic Optimization of Text Set Generation for Learning Multiple Query Intent Representations. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. 4003–4008.

[12] Yun He, Ziwei Zhu, Yin Zhang, Qin Chen, and James Caverlee. 2020. Infusing Disease Knowledge into BERT for Health Question Answering, Medical Inference and Disease Name Recognition. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 4604–4614.

[13] Yunhua Hu, Yanan Qian, Hang Li, et al. 2012. Mining query subtopics from search log data. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*. 305–314.

[14] Ayyoob Imani, Amir Vakili, Ali Montazer, et al. 2019. Deep neural networks for query expansion using word embeddings. In *European Conference on Information Retrieval*. Springer, 203–210.

[15] Zhengbao Jiang, Zhicheng Dou, and Ji-Rong Wen. 2016. Generating query facets using knowledge bases. *IEEE transactions on knowledge and data engineering* 29, 2 (2016), 315–329.

[16] Young Mo Kang, Wenhao Liu, and Yingbo Zhou. 2021. QueryBlazer: Efficient Query Autocompletion Framework. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*. 1020–1028.

[17] Se-Jong Kim and Jong-Hyeok Lee. 2015. Subtopic mining using simple patterns and hierarchical structure of subtopic candidates from web documents. *Information Processing & Management* 51, 6 (2015), 773–785.

[18] Weize Kong and James Allan. 2013. Extracting query facets from search results. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*. 93–102.

[19] Weize Kong and James Allan. 2014. Extending faceted search to the general web. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*. 839–848.

[20] Antonios Minas Krasakis, Mohammad Aliannejadi, Nikos Voskarides, et al. 2020. Analysing the Effect of Clarifying Questions on Document Ranking in Conversational Search. In *Proceedings of the 2020 ACM SIGIR on International Conference on Theory of Information Retrieval*. 129–132.

[21] Mike Lewis, Yinhan Liu, Naman Goyal, et al. 2020. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. 7871–7880.

[22] Weijie Liu, Peng Zhou, Zhe Zhao, et al. 2020. K-bert: Enabling language representation with knowledge graph. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 2901–2908.

[23] Bryan McCann, James Bradbury, Caiming Xiong, et al. 2017. Learned in translation: contextualized word vectors. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*. 6297–6308.

[24] Agnès Mustar, Sylvain Lamprier, and Benjamin Piwowarski. 2020. Using BERT and BART for Query Suggestion.. In *CIRCLE*.

[25] Dae Hoon Park and Rikio Chiba. 2017. A neural language model for query auto-completion. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1189–1192.

[26] Matthew E Peters, Mark Neumann, Mohit Iyyer, et al. 2018. Deep contextualized word representations. In *Proceedings of NAACL-HLT*. 2227–2237.

[27] Alec Radford, Karthik Narasimhan, Tim Salimans, et al. 2018. Improving language understanding by generative pre-training. (2018).

[28] Alec Radford, Jeffrey Wu, Rewon Child, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog* 1, 8 (2019), 9.

[29] Filip Radlinski and Nick Craswell. 2017. A theoretical framework for conversational search. In *Proceedings of the 2017 conference on conference human information interaction and retrieval*. 117–126.

[30] Chris Samarinas, Arkin Dharawat, and Hamed Zamani. 2022. Revisiting Open Domain Query Facet Extraction and Generation. In *Proceedings of the 2022 ACM SIGIR International Conference on Theory of Information Retrieval*. 43–50.

[31] Julian Seitner, Christian Bizer, Kai Eckert, et al. 2016. A Large DataBase of Hypernymy Relations Extracted from the Web.. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*. 360–367.

[32] Ivan Sekulić, Mohammad Aliannejadi, and Fabio Crestani. 2021. Towards Facet-Driven Generation of Clarifying Questions for Conversational Search. In *Proceedings of the 2021 ACM SIGIR International Conference on Theory of Information Retrieval*. 167–175.

[33] Wei Song, Ying Liu, Li-zhen Liu, et al. 2018. Semantic composition of distributed representations for query subtopic mining. *Frontiers of Information Technology & Electronic Engineering* 19, 11 (2018), 1409–1419.

[34] Alessandro Sordoni, Yoshua Bengio, Hossein Vahabi, et al. 2015. A hierarchical recurrent encoder-decoder for generative context-aware query suggestion. In *proceedings of the 24th ACM international on conference on information and knowledge management*. 553–562.

[35] Tianxiang Sun, Yunfan Shao, Xipeng Qiu, Qipeng Guo, Yaru Hu, Xuan-Jing Huang, and Zheng Zhang. 2020. CoLAKE: Contextualized Language and Knowledge Embedding. In *Proceedings of the 28th International Conference on Computational Linguistics*. 3660–3670.

[36] Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Hao Tian, Hua Wu, and Haifeng Wang. 2020. Ernie 2.0: A continual pre-training framework for language understanding. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 34. 8968–8975.

[37] Ashish Vaswani, Noam Shazeer, Niki Parmar, et al. 2017. Attention is all you need. In *Advances in neural information processing systems*. 5998–6008.

[38] Alexandra Vtyurina, Denis Savenkov, Eugene Agichtein, et al. 2017. Exploring conversational search with humans, assistants, and wizards. In *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems*. 2187–2193.

[39] Jian Wang and Wenjie Li. 2021. Template-guided Clarifying Question Generation for Web Search Clarification. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. 3468–3472.

[40] Qinglei Wang, Yanan Qian, Ruihua Song, et al. 2013. Mining subtopics from text fragments for a web query. *Information retrieval* 16, 4 (2013), 484–503.

[41] Sida Wang, Weiwei Guo, Huiji Gao, et al. 2020. Efficient Neural Query Auto Completion. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 2797–2804.

[42] Zhongyuan Wang, Haixun Wang, Ji-Rong Wen, et al. 2015. An inference approach to basic level of categorization. In *Proceedings of the 24th acm international on conference on information and knowledge management*. 653–662.

[43] Wentao Wu, Hongsong Li, Haixun Wang, et al. 2012. Probase: A probabilistic taxonomy for text understanding. In *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*. 481–492.

[44] Xiaobing Xue and W Bruce Croft. 2013. Modeling reformulation using query distributions. *ACM Transactions on Information Systems (TOIS)* 31, 2 (2013), 1–34.

[45] Hamed Zamani, Susan Dumais, Nick Craswell, et al. 2020. Generating clarifying questions for information retrieval. In *Proceedings of The Web Conference 2020*. 418–428.

[46] Hamed Zamani, Gord Lueck, Everest Chen, et al. 2020. Mimics: A large-scale data collection for search clarification. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 3189–3196.

[47] Hamed Zamani, Bhaskar Mitra, Everest Chen, et al. 2020. Analyzing and Learning from User Interactions for Search Clarification. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1181–1190.

[48] Zhengyan Zhang, Xu Han, Zhiyuan Liu, et al. 2019. ERNIE: Enhanced Language Representation with Informative Entities. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. 1441–1451.

[49] Ziliang Zhao, Zhicheng Dou, Jiaxin Mao, et al. 2022. Generating Clarifying Questions with Web Search Results. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 385–394.