Dialogue Strategy to Clarify User's Queries for Document Retrieval System with Speech Interface

Teruhisa Misu, Tatsuya Kawahara

School of Informatics, Kyoto University,
Sakyo-ku, Kyoto 606-8501, Japan

misu@ar.media.kyoto-u.ac.jp

Abstract

This paper addresses a dialogue strategy to clarify and constrain the queries for document retrieval systems. In spoken dialogue interfaces, users often make utterances before the query is completely generated in their mind; thus input queries are often vague or fragmental. As a result, usually many items are matched. We propose an efficient dialogue framework, where the system dynamically selects an optimal question based on information gain (IG), which represents reduction of matched items. A set of possible questions is prepared using various knowledge sources. As a bottom-up knowledge source, we extract a list of words that can take a number of objects and potentially causes ambiguity, using a dependency structure analysis of the document texts. This is complemented by top-down knowledge sources of metadata and hand-crafted questions. An experimental evaluation showed that the method significantly improved the success rate of retrieval, and all categories of the prepared questions contributed to the improvement.

1. Introduction

The target of spoken dialogue systems is being extended from simple databases such as flight information[1] to general documents[2] including manuals[3] and newspaper articles[4]. In such systems, the automatic speech recognition (ASR) result of the user utterance is matched against a set of target documents using the vector space model, and documents with high matching scores are presented to the user. We have developed a "Speech Dialogue Navigator", which can retrieve information from a large-scale software support knowledge base (KB) with a spoken dialogue interface[5].

In this kind of document retrieval systems, user queries must include sufficient information to identify the desired documents. In conventional document query tasks with typed-text input, such as TREC QA Track[6], queries are (supposed to be) definite and specific. However, this is not the case when speech input is adopted. The speech interface makes input easier. However, this also means that users can start utterances before queries are thoroughly formed in their mind. Therefore, input queries are often vague or fragmental, and sentences may be ill-formed or ungrammatical. Moreover, important information may be lost due to ASR errors. In such cases, an enormous list of possible relevant documents is usually obtained because there is very limited information that can be used as clues for retrieval. Therefore, it is necessary to narrow down the documents by clarifying the user's intention through a dialogue.

There have been several studies on the follow-up dialogue, and most of these studies assume that the target knowledge base has a well-defined structure. For example, Denecke[7] proposed a method to generate guiding questions based on a tree structure constructed by unifying pre-defined keywords and semantic slots. We also proposed a method to generate optimal questions by using the contents structure extracted from the manual of electric appliances[3]. However, these approaches are not applicable to general document sets without such structures.

In this paper, we propose a dialogue strategy to clarify the user's query and constrain the retrieval for a large-scale knowledge base, which is a collection of plain texts and does not have a structure nor any semantic slots. In the proposed scheme, the system dynamically selects an optimal question, which can reduce the number of matched items most efficiently. As a criterion of efficiency of the questions, information gain (IG) is defined. A set of possible questions is prepared using bottom-up and top-down knowledge sources. As a bottom-up knowledge source, we conduct dependency structure analysis of the document texts, and extract a list of words that can take a number of objects, thus potentially causing ambiguity. This is combined with top-down knowledge sources of metadata and hand-crafted questions. The system then updates the query sentence using the user's reply to the question, so as to generate a confirmation to the user.

2. Document retrieval systems with spoken dialogue interface

Our task involves text retrieval from a large-scale knowledge base. As the target domain, we adopt a software support knowledge base (KB) provided by Microsoft Corporation. The specification is listed in Table 1, and there are about 40K documents in total.

We have developed a document retrieval system

Table 1: Document set (Knowledge Base: KB)

Text collection	# documents	text size (byte)
glossary	4,707	1.4M
FAQ	11,306	12M
DB of support articles	23,323	44M

called "Speech Dialogue Navigator" for this KB with a spoken dialogue interface[5]. The system makes confirmations prior to and posterior to the retrieval based on two statistical measures. A *relevance score* represents matching degree with the document set, and is used for detecting erroneous keyword hypotheses. A *significance score* detects portions that consequently affect the retrieval results. With these measures, the system copes with ASR errors and redundancy in spoken language expressions. However, the previous version basically assumes 'one question and one answer'; thus, a lot of documents are matched if the user's input is vague or fragmental. In this paper, we present an enhanced "Speech Dialogue Navigator" that incorporates a dialogue strategy to clarify the user's queries.

3. Dialogue strategy to clarify user's vague queries

In the proposed framework, the system asks optimal questions to constrain the given retrieval results and help users find the intended ones. Questions are dynamically generated by selecting from a pool of possible candidates that satisfy the precondition. The information gain (IG) is defined as a criterion for the selection. The IG represents a reduction of entropy, or how many retrieved documents can be eliminated by incorporating additional information (a reply to a question in this case). Its computation is straightforward if the question classifies the document set in a completely disjointed manner. However, the retrieved documents may belong to two or more categories for some questions, or may not belong to any category. For example, some documents in our KB are related with multiple versions of MS-Office, but others may be irrelevant to any of them. Moreover, the matching score of the retrieved documents should be taken into account in this computation. Therefore, we define IG $H(S)$ for a candidate question S by the following equations.

$$H(S) = -\sum_{i=0}^{n} P(i) \cdot \log P(i)$$

$$P(i) = \frac{|C_i|}{\sum_{i=0}^{n} |C_i|}$$

$$|C_i| = \sum_{D_k \in i} CM(D_k)$$

Here, D_k denotes the k-th retrieved document by matching the query to the KB, and $CM(D)$ denotes the matching score of document D. Thus, C_i represents the

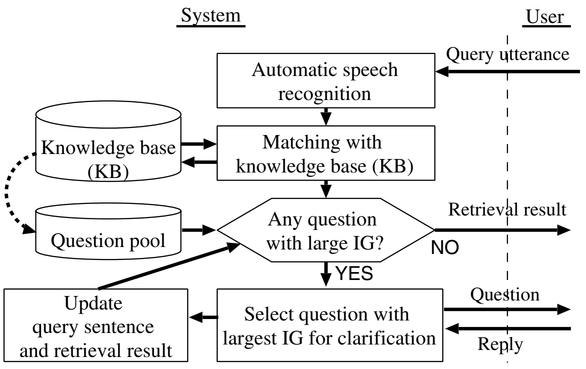
System
User
Query utterance
Automatic speech recognition
Knowledge base (KB)
Matching with knowledge base (KB)
Question pool
Any question with large IG?
Retrieval result
NO
YES
Update query sentence and retrieval result
Select question with largest IG for clarification
Question
Reply

Figure 1: System overview

number of documents classified into category i by candidate question S, which is weighted with the matching score. The documents that are not related to any category are classified as category 0.

The system flow incorporating this strategy is summarized below and also shown in Figure 1.

1. For a given ASR result of query utterance (query sentence), retrieve from the KB.
2. Calculate IG for all possible candidate questions.
3. Select the question with the largest IG (larger than a threshold), and ask the question to the user. Otherwise, output the current retrieval result.
4. Update the query sentence using the user's reply to the question and retrieve again.
5. Return to 2.

This procedure is explained in detail in the following sections.

4. Question generation based on bottom-up and top-down knowledge sources

We prepare a pool of questions using three methods based on bottom-up knowledge together with top-down knowledge of the KB. For a bottom-up knowledge source, we conducted a dependency structure analysis on the KB. As for top-down knowledge, we make use of metadata included in the KB and human knowledge.

4.1. Questions based on dependency structure analysis (method 1)

This type of question is intended to clarify the modifier or object of some words, based on dependency structure analysis, when they are uncertain. For instance, the verb "delete" can have various objects such as "application program" or "address book". Therefore, the query can be clarified by identifying such objects if they are missing. However, not all words need to be confirmed because the modifier can be identified almost uniquely for some words. For instance, the object of the word "shutdown" is "computer" in most cases in this task domain. It is tedious to identify the modifier of such words. We therefore determine the words to be confirmed by calculating entropy for modifier-head pairs from the text corpus. The procedure is as follows.

1. Extract all modifier-head pairs from the KB and the query sentences to a retrieval system[1] provided by Microsoft Japan.
2. Calculate entropy $H(m)$ for every word based on probability $P(i)$. This $P(i)$ is calculated with the occurrence count $N(m)$ of word m that appears in the text corpus and the count $N(i, m)$ of word m whose modifier is i.

$$H(m) = -\sum_{i} P(i) * \log P(i)$$

$$P(i) = \frac{N(i, m)}{N(m)}$$

As a result, we selected 40 words that have a large value of entropy, for example, "delete", "install" and "save".

4.2. Questions based on metadata included in KB (method 2)

We also prepare candidate questions using the metadata attached to the KB. In general large-scale KBs, metadata is usually attached to manage them efficiently. For example, category information is attached to newspaper articles and books in libraries. In our target KB, some documents include metadata of product names to which the document applies. The system can generate a question to which the user's query corresponds using this metadata. Fourteen candidate questions are prepared using this method.

4.3. Questions based on human knowledge (method 3)

Software support is conventionally provided by operators at call centers. We therefore prepare candidate questions based on the human knowledge that has been accumulated there. This time, three kinds of questions are handcrafted. For instance, the question "When did the symptom occur?" tries to capture key-phrases to identify relevant documents.

Table 2 lists examples of candidate questions prepared using the above three methods. An example dialogue where the system asks questions based on IG is in Figure 2.

5. Update of retrieval query sentence

Through the dialogue to clarify the user's query, the system updates the query sentence using the user's reply to the question, and then updates the retrieval result. Our backend information retrieval system does not adopt simple "bag-of-words" model, but conducts a more precise dependency structure analysis for matching; therefore forming an appropriate query sentence is desirable rather than simply adding keywords. Moreover, when making confirmation to the user, it is more comprehensible to present the updated query sentence than to show the list of ASR results. Here, the update rules of the query sentence are prepared beforehand. In the example in Figure 2, the system confirms "Retrieving with 'Characters on window button are too garbled to read' " at the end of the dialogue before making an actual retrieval.

Table 2: Example of candidate questions

Question	Ratio of applicable doc.	IG
· **Dependency structure analysis (Method 1)**		
What did you delete?	2.15 (%)	7.44
What did you install?	3.17 (%)	6.00
· **Metadata (Method 2)**		
What is the version of your Windows?	30.03 (%)	2.63
What is your application?	30.28 (%)	2.31
· **Human knowledge (Method 3)**		
When did the symptom occur?	15.40 (%)	8.08
Tell me the error message.	2.63 (%)	8.61

S1:	What is your problem?	
U1:	Too garbled to read.	
(Retrieval results):	1. Close button and maximize button are garbled.	
	2. Characters are garbled in Outlook Today.	
	3. Characters are garbled while inserting Japanese text.	
	4. VB application is garbled to read.	
	. . .	
(Calculate IG):	· Candidate question 1:	
	What is garbled to read?	– *IG* 5.27
	· Candidate question 2:	
	What is the version of your Windows?	– *IG* 1.43
	· Candidate question 3:	
	When did the symptom occur?	– *IG* 2.47
	. . .	
S2:	**(Select question with largest IG)** What is garbled to read?	
U2:	Characters on window button.	
S3:	**(Update query sentence)** Retrieving with "Characters on window button are too garbled to read".	

Figure 2: Example dialogue

6. Experimental evaluation

We implemented and evaluated the proposed method. The ASR system consists of Julius[8] for SAPI[2] and a trigram language model trained with the query corpus as well as texts of the KB. We collected test data from 14 subjects who had not used our system. Each subject was requested to retrieve support articles for 14 tasks, which consisted of prepared scenarios (query sentences were not given). The subjects were allowed to utter a query again up to twice per task if (they thought) an adequate retrieval result was not obtained. We collected 238 utter-

[1]http://www.microsoft.com/japan/enable/nlsearch/

[2]http://julius.sourceforge.jp/sapi/

ances for 196 (=14 × 14) tasks in total.

First, we evaluated the success rate of retrieval. We regarded a retrieval as successful when the retrieval result contained a correct document entry for the scenario. We compared the following cases.

1. Transcript: A correct transcript of user utterances, prepared manually, was used as an input.
2. ASR result (baseline): The ASR result was used as an input.
3. Proposed method (log data): The system generated questions based on the proposed method, and the users replied to them as they thought appropriate.

We also evaluated the proposed method by simulation to confirm its theoretical effect. Various factors of the entire system might influence the performance in a real dialogue which is evaluated by the log data. Specifically, the users might not have answered the questions appropriately or the replies might not have been correctly recognized. Therefore, we also evaluated with the following condition.

4. Proposed method (simulation): The system generated questions based on the proposed method, and appropriate answers were given manually.

Table 3 lists these results. The proposed method achieved a better success rate than when the ASR result was used. An improvement of 12.6% was achieved in the simulation case, and 7.7% by the log data. These figures demonstrate the effectiveness of the proposed approach. The success rate of the retrieval was about 5% higher in the simulation case than the log data. This was because there were ASR errors in user's uttered replies and also because users sometimes misunderstood the system's questions.

We also evaluated the efficiency of the individual methods. In this experiment, each of the three methods was used to generate questions. The results are in Table 4. The improvement rate by the three methods did not differ very much, and most significant improvement was obtained by using the three methods together. While the questions based on human knowledge are rather general and were used more often, the questions based on the dependency structure analysis are specific, and thus more effective when applicable. Hence, the questions based on the dependency structure analysis (method 1) obtained a relatively high improvement rate per question.

7. Conclusion

We proposed a dialogue strategy to clarify user' queries for document retrieval tasks. Candidate questions are prepared based on the dependency structure analysis of the KB together with KB metadata and human knowledge. The system selects an optimal question based on information gain (IG). Then, the query sentence is updated using the user's reply. An experimental evaluation showed that the proposed method significantly improved the success rate of retrieval, and all categories of the prepared questions contributed to the improvement.

Table 3: Success rate and average rank of correct document in retrieval

	Success rate	Rank of correct doc.
Transcript	76.1%	7.20
ASR result (baseline)	70.7%	7.45
Proposed method (log data)	78.4%	4.40
Proposed method (simulation)	83.3%	3.85

Table 4: Comparison of question methods

	Success rate	# generated questions (per dialogue)
ASR result (baseline)	70.7%	—
Dependency structure analysis (method 1)	74.5%	0.38
Metadata (method 2)	75.7%	0.89
Human knowledge (method 3)	74.5%	0.97
All methods (method 1-3)	83.3%	2.24

8. Acknowledgements

The authors are grateful to Prof. Kurohashi and Dr. Kiyota at University of Tokyo and Dr. Komatani at Kyoto University for their helpful advice.

9. References

[1] E. Levin, S. Narayanan, R. Pieraccini, K. Biatov, E. Bocchieri, G. Di Fabbrizio, W. Eckert, S. Lee, A. Pokrovsky, M. Rahim, P. Ruscitti, and M. Walker. The AT&T-DARPA communicator mixed-initiative spoken dialogue system. In *Proc. ICSLP*, 2000.

[2] A. Fujii and K. Itou. Building a test collection for speech-driven Web retrieval. In *Proc. EUROSPEECH*, 2003.

[3] K. Komatani, T. Kawahara, R. Ito, and H. G. Okuno. Efficient dialogue strategy to find users' intended items from information query results. In *Proc. COLING*, pages 481–487, 2002.

[4] C. Hori, T. Hori, H. Isozaki, E. Maeda, S. Katagiri, and S. Furui. Deriving disambiguous queries in a spoken interactive ODQA system. In *Proc. IEEE-ICASSP*, 2003.

[5] T. Misu, K. Komatani, and T. Kawahara. Confirmation Strategy for Document Retrieval Systems with Spoken Dialog Interface. In *Proc. ICSLP*.

[6] NIST, DARPA, and ARDA. The twelfth Text REtrieval Conference (TREC 2003). In *NIST Special Publication SP 500–255*, 2003.

[7] M. Denecke and A. Waibel. Dialogue strategies guiding users to their communicative goals. In *Proc. EUROSPEECH*, 1997.

[8] A. Lee, T. Kawahara, and K. Shikano. Julius – an open source real-time large vocabulary recognition engine. In *Proc. EUROSPEECH*, 2001.