Dependency Network Based Real-Time Query Expansion

Jiaqi Zou[1,*] and Xiaojie Wang[2]

School of Computer Science, Beijing University of Posts and Telecommunications
Beijing, 100876, China
jia7.zou@gmail.com, xjwang@bupt.edu.cn

Abstract. This paper presents a novel real-time query expansion method which offers expansion related to user query intention. A dependency relation network is first built by merging dependency trees of large numbers of sentences from a large-scale corpus. User queries are then expanded to verb-noun pairs or verb-attributes-noun multiple grams based on relations in the network, which is helpful to identify user intentions and to reduce search spaces. Experiments show that the proposed expansion method is not only effective in saving user inputs, but also brings significant improvement to retrieval performance.

Keywords: real-time query expansion, dependency relation, query intention.

1 Introduction

Real-time query expansion (RTQE) recommends a list of expanded query words when user types a query, which reduces user's keystrokes to complete information retrieval. Moreover, RTQE can help the user complete query intention, thus getting better retrieval performance. White and Marchionini [1] showed real-time and interactive query expansion is useful, it can reduce the time needed to perform a query and improve the query quality.

The most widely used way to RTQE is offering the most frequent query from the query log that includes the current input as a substring [2], [3]. With users inputting more letters or words, the expanded queries changes all along. So the more words inputted by users, the more precise the expanded words are. For example, in the RTQE method used by Microsoft Bing[1], when the inputted words are "olympic lo", the first expanded query is "olympic logo"; when the inputted words change to "olympic lon", the first expanded query changes to "olympic london 2012".

This kind of RTQE is based on the letters inputted. It expands words according to word frequency or word co-occurrence frequency. In the research of Ji et al. [2], they used enhanced string matching method to support interactive fuzzy keyword expansion.

An alternative method for RTQE uses user context information. Arias et al. [4] proposed a method using user context information on mobile platform. However, it lacks scalability due to the use of manually built rules collection. Bar-Yossef et al. [5]

[*] Corresponding author.

[1] www.bing.com

also used user context to recommend queries of the same search session that have current input as their prefix. Their method is effective when user input is short.

However, there is little work on RTQE concerning user's query intention. Strohmaier et al. [6] introduced an intentional query expansion method to make users' intentions more explicit during searching. By mapping implicit queries to explicit queries in the same search session from query logs, they suggested that explicit queries containing at least one verb word might reflect possible user intentions. Their results show that intentional query expansions can be used to diversify result sets and improve click-through ratio.

This paper proposes a RTQE method using a dependency network to get better query intentions. The dependency network is constructed using dependency relations from the parsing results of large-scale corpus. A dependency relation is either a verb-noun pair or a verb-attributes-noun multiple gram.

Experiments show that the proposed RTQE method improves the retrieval performance and reduces user input effort greatly. It is also suggested that dependency relations can be used to determine user's query intentions.

The rest of this paper is organized as follows. Section 2 describes the building of dependency network and the RTQE method based on this dependency network. The evaluation and experiments are detailed on section 3. Finally, section 4 gives the conclusion and future work.

2 Method

2.1 Representation of Query Intention

Border [1] proposed task-oriented classification of query intention, i.e. navigational, informational and transactional. He et al. [8] used search results to find query intention. They represented query intentions using verb-noun pairs. Duan et al. [9] pointed out that this kind of representation is suitable for both informational and transactional intention, but not good for navigational intention. Meanwhile, they showed that verb-noun pairs can be acquired from dependency relations in sentences.

But we found the verb-noun pairs are still not sufficient to express query intention clearly and completely. They can be used as the backbone of query intention. As an illustrating example, two queries "learn English language" and "learn programming language" are both represented by the verb-noun pair "learn language". But the intention difference between them is quite big. So other components needs to be added in verb-noun pair in order to describe query intention better.

In this research, we add modifiers of noun, namely attributes, to complete the query intention. We mainly consider pre-attributes here. Query intention can be represented by verb-attributes-noun. In English, the normal word order of attributes is determiner, adjective, participle, gerund and noun. Determiner contains article, possessive pronoun, substantive genitive, numeral and classifier which have little function in intention expression. So we only consider attributes of adjective, participle, gerund and noun words.

For the above example, after adding attribute to the verb-noun pair "learn language", the intention difference between them is clearly expressed.

2.2 Construction of Dependency Relation Network

First we collect dependency relations from large-scale sentences parsed according to dependency structures. Then we build dependency relation network by combining and relating these dependency relations.

Stanford Parser[2] is used here to do dependency parsing on the sentences. From the parsing result of each sentence, we extract dependency relations between verb and noun and relations between this noun word and its attributes.

For example, for the sentence "How to change a car tire", we get results as shown in Fig. 1 after dependency parsing using Stanford Parser.

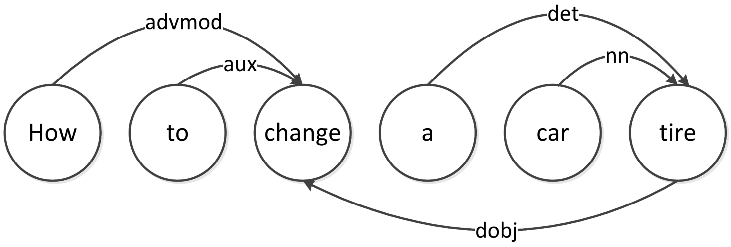

Fig. 1. Parsing result of sentence "How to change a car tire"

Each arrow in this figure means a binary dependency relation. We use vn[verb, noun] to represent dependency relation between verb and noun, and an[attribute, noun] to represent dependency relation between noun and its attributes. In this example, we get vn[change, tire] and an[car, tire].

We combine these two relations in a network to represent the query intention as shown in Fig. 2. Because user can input both verb and noun first when inputting queries, for the convenience of indexing during the RTQE process, we construct two networks for each query intention. One starts from verb as shown in the left of Fig. 2 and the other one in the right starts from noun. We attach the arrow that represents 'an' to that of 'vn' since 'vn' is the kernel relation in concern and 'an' is second level

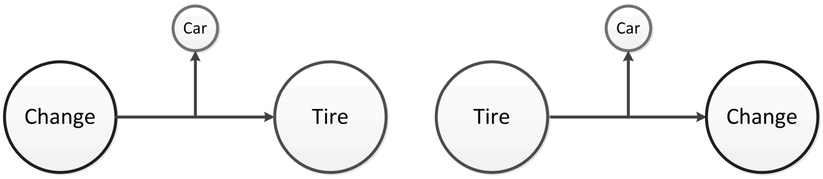

Fig. 2. The left network starts from verb, the right one starts from noun

[2] http://nlp.stanford.edu/software/lex-parser.shtml

relation. Although the network is a little different from dependency network, we still call it dependency network in the following parts of the paper for simplification.

It should be noted here that in some sentences there are more than one attributes which have dependency relations with the head noun like “How to buy a used wedding dress”. In this sentence, we extract vn[buy, dress] and an[used wedding, dress]. For this kind of sentence, we represent each attribute with a node. As concluded in linguistic analysis, the closer an attribute is to the noun, the closer relationship it has with the noun. So we first connect the attribute which is closet to noun, and then connect the second closet one, and so on.

In the above example, there is a verb node “buy”, a noun node “dress”, and two attribute nodes “used” and “wedding”. In the network, verb node and noun node are connected first, that is vn[buy, dress]. Then the attribute node “wedding” is connected to the path between verb node and noun node. Finally the attribute node “used” is connected to attribute node “wedding”. The network using verb node as starting node is shown in Fig. 3.

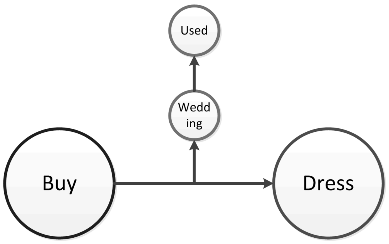

Fig. 3. Network with plural attributes

Moreover, for intransitive verbs, they are connected with noun word by a preposition word. So in the network between an intransitive verb node and a noun node, a preposition node is needed. We use vpn[prep, vn] to represent relation between intransitive verb and noun. For sentences such as “How to look up phone number”, we extract vpn[up, vn], vn[look, number] and an[phone, number]. In the dependency network, the preposition node is attached to the path between verb node and noun node. The network of this example using noun node as starting node is shown in Fig. 4.

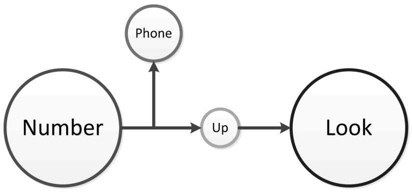

Fig. 4. Network with a prep node

After constructing network for each sentence, the next step is to combine all the networks. The combination is done separately for those starting from verb node and noun node. For networks start from verb node, if two of them share the same verb node, they will be combined into one network rooted with this verb. In this way, we can get all the dependency relations starting from the same verb in one network. Similar operation is done for networks starting from noun node. Finally, we have two dependency relation based networks. (Fig. 5)

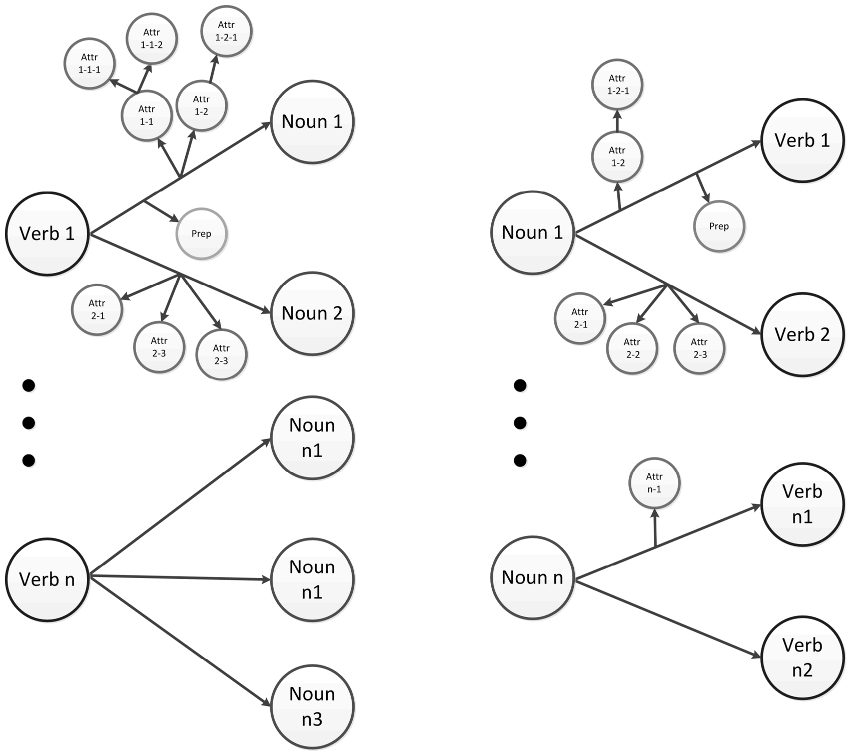

Fig. 5. A fragment of the combined network, attribute node is labeled by "Attr", preposition node is labeled by "Prep", the left one starts from verb, the right one starts from noun

2.3 RTQE Method

Based on this dependency relation network, we can expand noun for inputted verb and expand verb for inputted noun. Then we can expand attribute words for verb-noun pair to express query intention more complete and precise. Furthermore, by combining this kind of expansion with the string matching expansion technique, we construct a RTQE system.

Specifically this RTQE method works in the following process:

First, the user inputs some letters (prefix of a word, e.g., "ti") to the search box. According to the prefix, our system searches the starting nodes from the two dependency networks constructed in section 2.2 for words beginning with this prefix, and list matched words sorted by their frequency in the corpus (Fig. 6-a).

Second, when the user selects a word from the list, our system can locate the node in the network that stands for the selected word ("tire" in this example). Then our system can expand verbs or nouns connected to this node. (Fig. 6-b) The user can modify expanded words by inputting prefix (Fig. 6-c). After this step, the intention backbone (verb-noun) is built. If the verb is intransitive, we can get (verb-preposition-noun).

Third, when the user selects a verb or noun word shown in the expanded list, our system can locate this verb-noun relation in the network. If there are some attributes attached to this relation, our system will pop-up the attribute words as expanded words. The attributes that are closest to the nouns are expanded first, then the less closest ones (Fig. 6-d). Still user can reject one recommendation by typing letters he prefers (Fig. 6-e).

The procedure ends when there are no more words can be expanded or the user feels satisfied with the current query.

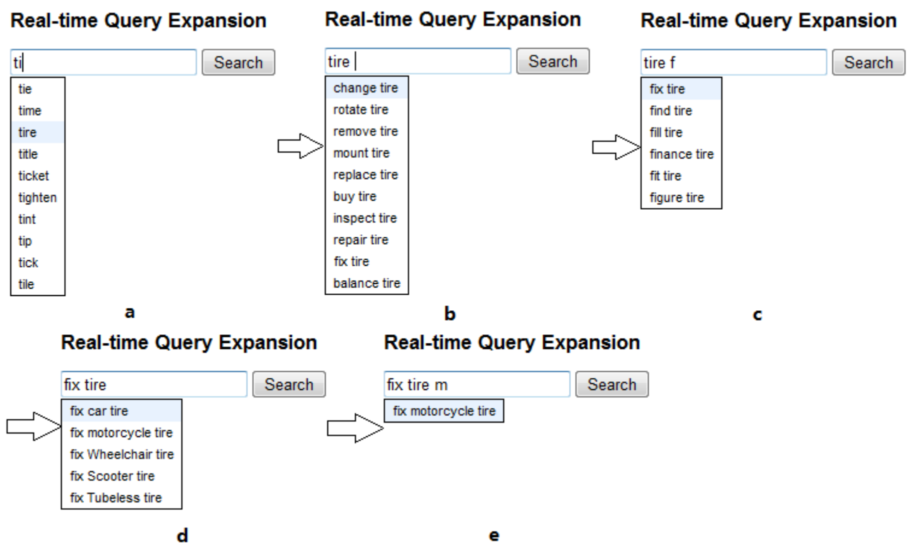

Fig. 6. Example of RTQE process

3 Experiments

We designed four experiments to test the performance of the RTQE method described in Section 2.

3.1 Experimental Corpus

We used corpus from www.ehow.com. This website is an online guide offering step-to-step instructions for how-to questions. According to Broder's query intention classification [7], how-to questions belong to informational intention. From Duan's research [9] we know that informational query intention can be represented by verb-noun pair. So we can represent the query intention of how-to questions by the improved method using verb-attributes-noun.

The corpus we used has articles of 20 categories from ehow, including Health, Cars, Computers, Food & Drink and so on. The total size of the corpus is 915,600 articles. Each article is a sequence of instructions about one how-to question, and each title represents this question. When building dependency relation network, we did dependency parsing for each title, like "How to fix motorcycle tire".

3.2 Experimental Evaluation Measurement

We invited 10 volunteers to participate in the experiments. All the volunteers have advanced engineering education background.

First we test how many keystrokes our RTQE method will save. Reducing users' efforts in generating queries is very important for RTQE method, especially for users of mobile search systems.

For each query, the keystrokes and mouse clicks needed to generate a query is recorded. Each keystroke or mouse click is recorded as an operation.

Suppose that for query x, the operations needed when the user does not use RTQE is $OPFull_x$, and the operations needed when using RTQE is $OPExpanded_x$.

The percentage of saved operations is:

$$Saved = \frac{\sum_{x=1}^{n}(OpFull_x - OpExpanded_x)}{\sum_{x=1}^{n} OpFull_x} \tag{1}$$

Second, we test the expansion success percentage of this RTQE method. For a given query intention, if the user can find a query exactly related to this intention from the expanded list, we call it a successful expansion. The expansion success percentage is the percentage of queries with a successful expansion in all the queries.

For example, the user wants to search a query "install windows system", if this query can be found in expanded list during RTQE process, then the expansion is successful, otherwise it is not.

Suppose that the number of all the queries is AllNum, the number of query with a successful expansion is SuccessNum. The expansion success percentage is:

$$SuccessPercentage = \frac{SuccessNum}{AllNum} \tag{2}$$

Moreover, in the queries with successful expansion, we compare the retrieval performance of the original query user typed in, the query after verb-noun expansion and the query after verb-attributes-noun expansion. The aim of this test is to know how much effect this RTQE method has for query intention completion.

We use precision and nDCG score for evaluation.

Precision means the percentage of retrieved correct answers of all the correct answers.

nDCG is a measure of effectiveness in information retrieval. We grade the relevance of a result from 0 to 3. 0 means this result is totally not related to the query while 3 means totally relevant. The rank of each result is done by the user who does this search.

Suppose that rel_x means the graded relevance of the result at position x in the search results. For a search result list, its DCG score is:

$$DCG_p = \sum_{x=1}^{p} \frac{2^{rel_x} - 1}{\log_2(1+x)} \tag{3}$$

And nDCG score is the DCG score divide the maximum possible DCG score from position 1 till position p.

$$nDCG_p = \frac{DCG_p}{IDCG_p} \tag{4}$$

Finally, we compare the RTQE result of our method with that of Microsoft Bing search engine.

According to the experiments described above, we assigned the following tasks for each test participant:

In accordance with the limit of the 20 categories, do search for a how-to question each time. Make sure that the tester is clear about the query intention before inputting. Use the RTQE method to expand the query.

Our system can get the amount of operations of the user for each query. The tester should record whether an expansion is successful or not.

After generating the query and doing search, the system will return the Top 10 search result of the three kind of queries. Rank the results by judging the relevance of each result with the query intention.

3.3 Experimental Results and Analysis

By the test of the volunteers, we get 200 queries and their test results as follows:

Table 1. Result of the operation number test

	Without RTQE	With RTQE
Average number of operations	15.0	5.437

Table 1 shows that the percentage of average saved operations is 63.75% after RTQE. It proves our RTQE method is very useful for saving user effort.

This is because the structure of the dependency relation network we built directly connects verb, noun and attributes which combine to be query intentions. So all the expanded words our system recommends are components of possible query intentions. And the expanded words are correct word collocations because we built the network using dependency parsing. Moreover, combining the dependency relation network with prefix string matching method makes our RTQE method more effective in reducing the operations needed.

We use the query in Fig. 6-e for example here. We use a letter "m" to find the word "motorcycle". Only in this step, eight operations are saved. We can see that this RTQE method really reduces the operations.

Table 2. Result of expansion success percentage test

Times	query expansion success	query expansion failed
	168	32

Table 2 shows the expansion success percentage is 84%. It means most of the queries in given categories can use our RTQE method to generate and the dependency relation network represents user query intention well.

Because we built the dependency relation network using a large-scale corpus, so nearly all the possible query intentions of selected categories are included in the network. Therefore, this RTQE method can get a high success percentage in this test.

Table 3. Result of retrieval performance

Query type	Precision	nDCG score
Original query word	0.73%	13.11%
Query after verb-noun expansion	9.47%	37.37%
Query after verb-attributes-noun expansion	79.2%	88.95%

The data of Table 3 is computed by the 168 queries in Table 2 which are successfully expanded. We found that the retrieval performance after verb-attributes-noun expansion is better than the performance after verb-noun expansion. And the performance of the latter is better than the performance of not expanded queries. Although the retrieval performance improves naturally with the increase of query words, but obviously importing words that do not relate to the query intention will make the retrieval performance drop. The significant difference in the performance of the three kinds of queries can prove our RTQE method is effective in representing query intention so that resulting in good retrieval performance.

We still use the query in Fig. 6-e as an example here. The search result of word "tire" are all things related with tire, but contains articles that do not deal with "fix tire", like "How to store a flat tire".

The result after verb-noun expansion is much better. All the results is related to "fix tire", like "How to fix a flat tire" and "DIY flat tire fix", which are not found by the first query method.

The result after verb-attributes-noun expansion is the best of the three. It clearly represents the query intention that we want to fix motorcycle tire. The articles found are like “the best way to fix a flat motorcycle tire”. In contrast, the results of the second method cannot represent the feature “motorcycle”.

However, sometimes the query result after verb-attributes-noun expansion is not the best. For example, the search result of “buy satellite phone” is not good as that of “buy phone”. The reason is that the attribute “satellite” imports some articles that do not deal with “buy phone”, like “How to call a satellite phone”. But this kind of result is quite few so the overall result is influenced very little.

After this test, we compare our method with the widely used search engine Microsoft Bing. Bing has two query expansion functions. One is the RTQE function “Search suggestion”, the other is “Related search” which offers possible useful queries after the users do searches. To get English expanded queries, the location option in Bing is set to United Kingdom. We try the 200 queries contributed by the volunteers in Bing.

After a preliminary investigation, we found that the RTQE result of Bing differs a lot if the word order of a query changes. For example, if we type words in the order of “learn street dance”, Bing can offer the correct recommendations. However, if we change the order, Bing cannot recommend the words we want.

So we categories the RTQE result of Bing into three groups: cannot get correct recommendations (NOT); get correct recommendations only in normal word order(NORMAL); can get correct recommendations both in normal order and other word orders(ALL). Table 4 shows the test result of Bing RTQE method (do not contain “Related search” result.)

Table 4. Result of Bing RTQE test

Group	NOT	NORMAL	ALL
Percentage	49%	33%	18%

As shows in Table 4, half of the queries cannot get correct recommendations while 84% of the queries are successfully expanded in our method. Moreover, only 18% of the queries can get correct recommendations when the query words are not inputted in accordance with the normal word order in Bing, while both verb and noun words are allowed to be inputted first in our method.

For those queries that are not in the ALL group, we do search when the RTQE fails and judge the recommendation results of the ‘Related Search’ function. We find that only 13.3% of these queries can get useful query recommendations.

This test shows that our RTQE method has advantages when compared with industrial search engine.

After testing, we did survey of the testers on their experience of the RTQE method. We got the following message after summarizing their feedback.

This RTQE method is quick, and is able to meet the speed requirement of RTQE methods. And it is quite new and different from other RTQE methods. Most of the how-to questions in the 20 categories can be successfully expanded and the search results are satisfying.

But sometimes the testers feel inconvenience when doing search because currently the RTQE method can only handle verb, noun and attributes. And this method is category sensitive. When searching a query which is not in the given category, the expansion result is not good.

4 Conclusion

This paper proposed a novel RTQE method concerning user query intention. We used a large corpus from www.ehow.com to build a dependency relation network of verb, noun and attributes, then designed a RTQE method using this network. The proposed RTQE method is proved to be effective in representing user query intention and hence improve retrieval performance.

In the future, we will work on reducing the limit on the types of part of speech and do more research on offering personalized expanded results to represent the user intention better.

References

1. White, R.W., Marchionini, G.: Examining the effectiveness of real-time query expansion. Inf. Process. Manage. 43(3), 685–704 (2007)
2. Ji, S., Li, G., Li, C., Feng, J.: Efficient interactive fuzzy keyword search. In: WWW, pp. 371–380 (2009)
3. Barouni-Ebrahimi, M., Ghorbani, A.A.: On Query Completion in Web Search Engines Based on Query Stream Mining. In: Web Intelligence, pp. 317–320 (2007)
4. Arias, M., Cantera, J.M., Vegas, J., Fuente, P.D.L., Alonso, J.C., Bernardo, G.G., Llamas, C., Zubizarreta, Á.: Context-Based Personalization for Mobile Web Search. In: PersDB, pp. 33–39 (2008)
5. Bar-Yossef, Z., Kraus, N.: Context-sensitive query auto-completion. In: WWW, pp. 107–116 (2011)
6. Strohmaier, M., Kroll, M., Korner, C.: Intentional Query Suggestion: Making User Goals More Explicit During Search. In: WSCD, pp. 68–74 (2009)
7. Broder, A.Z.: A taxonomy of web search. SIGIR Forum, 3–10 (2002)
8. He, K., Chang, Y., Lu, W.: Improving Identification of Latent User Goals through Search-Result Snippet Classification. In: Web Intelligence, pp. 683–686 (2007)
9. Duan, R., Wang, X., Hu, R., Tian, J.: Dependency Relation Based Detection of Lexicalized User Goals. In: Yu, Z., Liscano, R., Chen, G., Zhang, D., Zhou, X. (eds.) UIC 2010. LNCS, vol. 6406, pp. 167–178. Springer, Heidelberg (2010)