Query Term Ranking based on Dependency Parsing of Verbose Queries

Jae-Hyun Park and W. Bruce Croft
Center for Intelligent Information Retrieval
Department of Computer Science
University of Massachusetts, Amherst, MA 01003
{jhpark,croft}@cs.umass.edu

ABSTRACT

Query term ranking approaches are used to select effective terms from a verbose query by ranking terms. Features used for query term ranking and selection in previous work do not consider grammatical relationships between terms. To address this issue, we use syntactic features extracted from dependency parsing results of verbose queries. We also modify the method for measuring the effectiveness of query terms for query term ranking.

Categories and Subject Descriptors

H.3.3 [**Information System**]: Information Search and Retrieval

General Terms

Algorithm, Experimentation, Performance

Keywords

Query Term Ranking, Query Formulation, Dependency Parse

1. INTRODUCTION

Most search engines have a tendency to show better retrieval results with keyword queries than with verbose queries. Verbose queries contain more redundant terms when compare to keyword queries. These redundant terms have grammatical meaning for communication between humans to identify keywords from questions or sentences. However, searching engines do not explicitly use these terms to consider grammatical roles of terms within queries. For example, given a verbose query, *"Identify positive accomplishments of the Hubble telescope since it was launched ..."*, search engines cannot recognize that *"Hubble telescope"* is the key concept of the query whereas *"accomplishments"* should be considered as a complementary concept, while people can readily identify this by analyzing the grammatical structure of the query. Therefore, search engines potentially need a method for using the grammatical structure of queries.

In this work, we rank terms in a verbose query and reformulate a new query with highly ranked terms. Good selection methods should be able to use the grammatical roles of terms within a query. To do this , we use syntactic features extracted from dependency parsing trees of queries. A typed dependency parse provides dependent relationships based on grammatical relations [3]. For example, a "direct object" type of dependent relationship tells us a verb and a noun are related and the type of relationship. By using syntactic features extracted from typed dependency parsing trees, we expect that a ranking function should be able to reflect grammatical roles of terms.

Also, we suggest a new method for measuring effectiveness of query terms in query term ranking. A previous approach [6] ranks sets of terms in order to take account of underlying relations between terms. This approach increases a query term weighting list and causes a data sparseness problem. Thus, we estimate ranks of terms by comparing effectiveness with different combinations of terms.

2. QUERY TERM RANKING

2.1 Features extracted from Dependency Parsing

We use syntactic features extracted from dependency parsing to capture the grammatical properties of terms for a query. Features used by previous work in query term ranking [1, 5, 6] are inadequate to reflect these characteristics. The limitation of these features is that they are based on individual terms. Therefore, features such as tf, idf, part-of-speech(PoS) tag, etc. will not change even if the role of the term changes according to the structures of queries. Features collected from sub-queries [5] are also unlikely to reflect grammatical characteristics because they are not affected by the structure of quries.

We propose to overcome this limitation by using dependency parsing trees. A dependency parse provides dependent information between individual words within a query. A typed dependency parse labels dependencies with grammatical relations such as subject or indirect object [3]. Figure 1 shows an example of typed dependency parse trees. The advantage of syntactic features collected from dependency parsing results is that syntactic features have the ability to relate grammatically connected terms [2].
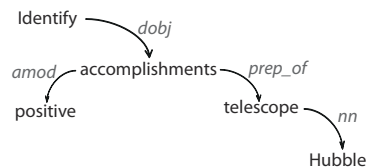
Figure 1: An example of dependency parsing trees. Labels attached to arcs are types of dependencies.

Identify —dobj→ **accomplishments** (a) * —dobj→ **accomplishments** (b) Identify —*→ **accomplishments** (c)
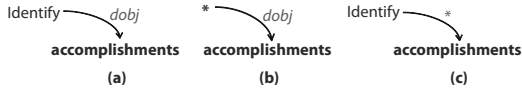
Figure 2: Three types of syntactic features for the term *"accomplishments"*. (a) An original syntactic feature (b) The word are generalized to a * (c) The type of the dependency are generalized to a *

It is infeasible to use all dependency parse tree fragments as syntactic features. We limited the number of arcs in syntactic features to two arcs. Even if we limit the number of arcs, some of collected tree fragments are too specific to have a reliable amount of training data and not all of them are useful. We generalize syntactic features which consist of arcs labeled with types of dependencies nodes with words which are dependent. Figure 2 shows an example of an original syntactic feature and its generalized features. In the figure, "*" means any words or any types of a dependency.

2.2 Measuring Effectiveness of Terms

Our approach aims to rank individual query terms among $T = \{t_1, t_2, ..., t_n\}$ from a verbose query and to select effective terms to formulate a query. Bendersky and Croft [1] manually annotate key concepts that have the most impact on effectiveness. Lee et al. [6] rank terms based on effectiveness of retrieval.

Lee et al. [6] pointed out that there are underlying dependencies between terms in a query. For example, in the verbose query, *"Find articles containing contents from reports on the decline of the unemployment rate as South Korea overcame the foreign exchange crisis."*, *unemployment rate*, *reports*, and *contents* are the most effective terms individually. However, if we combine terms in a proper way, *unemployment rate*, *South Korea*, and *foreign exchange* will be the top ranked effective terms. To capture these underlying relationships between terms, they extend candidates from a single term to a set of terms $c_m = \{t_i | t_i \in T\}$.

The problem is that, when we rank a set of query terms, a query term ranking list will increase exponentially. To avoid this problem, we used a set of terms c_m for measuring the effectiveness of terms instead of using it as candidates of query term ranking. The modified method for measuring effectiveness is as follows:

$$E(t_i) = \frac{1}{N} \cdot \sum_{c \in C_m} (\varphi(c, t_i) - \varphi(c)) \qquad (1)$$

where C_m is all possible combinations of terms where $m = |c|$. N is $m = |C_m|$ and $\varphi(\cdot)$ is a effectiveness measure function.

3. EXPERIMENTS AND ANALYSIS

We evaluated our proposed method using two TREC collections: Robust 2004 (topic numbers are 301-450 and 601-700) and Wt10g (topic numbers are 450-550). The average number of nouns, adjectives and verbs in queries of Robust2004 and Wt10g are 8.7 and 6.5 per a query, respectively. We used the language model framework with Dirichlet smoothing(μ set to 1,500). Indexing and retrieval were conducted using the Indri toolkit.

To rank query terms, we used RankSVM [4]. We trained query term ranking models for each queries using leave-one-

Table 1: Mean Average Precision(MAP) of Robust04 and Wt10g collections, Key-Concept: using key concept [1] as labels of training data, Auto: using effectiveness in retrieval as labels of training data

		Robust04	Wt10g
<title>		25.17	18.55
<desc>		24.07	17.52
Key-Concept	binary	23.98	18.55
	weight	24.24	19.45
Auto	binary	25.40	17.91
	weight	26.21	19.15

out cross-validation in which one query was used for a test set and the others were used for a training set. We labeled training data based on Key concepts [1] and the effectiveness measured by Eq. 1 in which we chose nDCG as the effectiveness measure. We used syntactic features in addition to tf, idf, and PoS tag features.

When we combined selected terms with original queries, we used two approaches. First, we assigned uniform weights to selected terms. Alternatively, we used query term ranking scores as the weight for selected terms.

Table 1 shows the results. They show that selected terms based on query term ranking have better performance than description queries except for one result in which we used key concepts and uniform weighting. In this case, only the most important concepts in queries are labeled, whereas the effectiveness in retrieval is measured for all terms in queries. This difference makes the method using the effectiveness of terms(**Auto**) is superior for the relatively longer queries in Robust2004, and the method using key concepts(**Key Concept**) better for the shorter queries in Wt10g.

4. CONCLUSIONS

In this paper, we propose a query term ranking method that uses syntactic features extracted from dependency parsing trees. By using syntactic features, we can take account of grammatical relationships between terms in addition to the characteristics of individual terms themselves. We also modify the query term ranking method to measure the effectiveness of terms based on combinations of terms. Experimental results showed that the terms selected by the query term ranking method improved retrieval performance.

5. ACKNOWLEDGMENTS

6. REFERENCES

[1] M. Bendersky and W. B. Croft. Discovering key concepts in verbose queries. In *Proc. ACM SIGIR*, pages 491–498, 2008.

[2] A. Chanen. *A comparison of human and computationally generated document features for automatic text classification*. PhD thesis, The University of Sydney.

[3] M. De Marneffe, B. MacCartney, and C. Manning. Generating typed dependency parses from phrase structure parses. In *Proc. LREC 2006*, 2006.

[4] T. Joachims. Optimizing Search Engines Using Clickthrough Data. In *Proc. ACM KDD*, pages 133–142, 2002.

[5] G. Kumaran and V. Carvalho. Reducing long queries using query quality predictors. In *Proc. ACM SIGIR*, pages 564–571, 2009.

[6] C. Lee, R. Chen, S. Kao, and P. Cheng. A term dependency-based approach for query terms ranking. In *Proc. CIKM*, pages 1267–1276, 2009.