Identifying and Ranking Possible Semantic and Common Usage Categories of Search Engine Queries

Reza Taghizadeh Hemayati[1], Weiyi Meng[1], Clement Yu[2]

[1] Department of Computer Science, Binghamton university, Binghamton, NY 13902, USA
{hemayati,meng}@binghamton.edu
[2] Department of Computer Science, University of Illinois at Chicago, Chicago IL 60607, USA, yu@cs.uic.edu

Abstract. In this paper, we propose a method for identifying and ranking possible categories of any user query based on the meanings and common usages of the terms and phrases within the query. Our solution utilizes WordNet and Wikipedia to recognize phrases and to determine the basic meanings and usages of each term or phrase in a query. The categories are ranked based on their likelihood in capturing the query's intention. Experimental results show that our method can achieve high accuracy.

Keywords: Query categorization, Search engine, Wikipedia, WordNet.

1 Introduction

Current search engines often return too many useless results for users' queries. One way to tackle this problem is to group search results into multiple categories such that all results in the same category correspond to the same meaning of the query. This makes it much easier for users to identify useful results. Most current result clustering techniques are based on word-match similarity. Although a few techniques have used semantic similarity [4, 5], they have various weaknesses. E.g., current techniques do not explicitly and systematically take *usages* of query terms into consideration. Here a term's usage means its use beyond its common meanings in dictionaries. Not considering these usages would lower the quality of search result clustering.

In this paper, we propose a new solution to identify and rank all possible categories of any user query based on both the possible meanings and the possible usages of the terms and/or phrases in the query. Our solution has the following unique features. First, our method utilizes both WordNet and Wikipedia. Second, we apply rule-based techniques to merge the meanings of individual query terms/phrases to increase the clarity of produced categories. Third, we generate candidate categories for a query by considering all combinations that can be formed from different meanings/usages of the terms/phrases in the query. Finally, our method ranks the generated categories by taking into consideration both the importance of each term/phrase in the query and the strength of the relationships between the terms and phrases in the query.

For the rest of the paper, Section 2 reviews related work, Section 3 provides an overview of our approach, Sections 4, 5 and 6 present the main steps of our approach, Section 7 concludes the paper.

2 Related Work

Our work is related to word sense disambiguation (WSD). While we aim to find all possible senses of a term, WSD tries to find the most likely sense only. [15] suggested an unsupervised knowledge-based WSD algorithm. [12, 14] proposed several WSD algorithms based on semantic relatedness using WordNet. In [13], Wikipedia was used for WSD. [1] presented techniques for clustering WordNet word senses and they could be used to merge WordNet senses. In our work, we identify all possible meanings/usages of each query term/phrase using both WordNet/Wikipedia. We also perform sense merging for meanings/usages obtained from both WordNet/Wikipedia. Furthermore, we keep all meanings and usages and rank them.

Some researchers used web directories like Yahoo directory or ODP to categorize/classify user queries. Mapping user queries to hierarchical sequences of topic categories was studied in [8, 14]. [11] proposed to map user queries to categories using a user profile. Our method does not use existing category hierarchies. We provide a systematic method to generate all possible categories for each query.

Some category ranking methods were studied in [6] and the best method is based on the similarity between a query and the clusters/categories. In contrast, our category ranking method is based on the importance of each term/phrase in the query and the strength of the relationships between the terms and phrases in the query.

3 Method Overview

Our method has the following three main steps:

Alternative query generation. For each user query *Q*, this step generates a set of *alternative queries* (AQs). All AQs contain the same set of query terms in *Q* but different phrases. The key task is to identify different possible phrases comprised of the terms in *Q*. For convenience, we call both query terms and phrases as ***concepts***.
Definition category generation. In this paper, a *definition category* (DC) is a combination of *meanings* or *usages* derived from the concepts of an AQ. This step is to generate all possible DCs for each AQ.
Definition category ranking. This step ranks the DCs generated in Step 2.

4 Alternate Query (AQ) Generation

Identify Valid Phrases:
We use Wikipedia and WordNet to recognize phrases. From these sources we can find all dictionary phrases and most well known proper nouns. The order of terms inside each phrase is significant. In this work, terms that are not part of a dictionary phrase or proper names will be considered individually. Given an *n*-word query $Q(w_1, w_2, \ldots, w_n)$, we send *Q* to Wikipedia and WordNet to check whether the *n* words form a phrase. If they do, we stop; else we search for possible (*n*-1)-word phrases, i.e., $(w_1, \ldots, w_{n-1})$ and $(w_2, \ldots, w_n)$. This process is repeated until reaching the two-word candidates. We do not look for sub-phrases inside any valid phrase already found.

To determine whether a sequence of words *p* forms a valid phrase using Wikipedia, we submit *p* as a query to Wikipedia and consider the following three cases:
Case 1: A definition page entitled by *p* is returned. This indicates that *p* is a well known phrase so we consider *p* as a valid phrase.
Case 2: A page saying something like "*p* may refer to the following definitions" is returned. This means that *p* refers to different definitions. In this case, by following the link for each definition, we will be directed to the definition page for that phrase. If one of the linked pages is entitled by *p*, we consider *p* as a valid phrase.
Case 3: If none of the above cases is true, we don't consider *p* as a valid phrase.

To determine whether *p* is a valid phrase using WordNet, we submit *p* to WordNet and consider the following two cases:
Case 1: A page containing some definitions (called synsets in WordNet) is returned, i.e., *p* is defined in the dictionary. In this case, we consider *p* as a valid phrase.
Case 2: If no entry for *p* in WordNet is found, we don't consider *p* as a valid phrase.

Building the Set of Alternative Queries:
In this step we form all possible combinations consisting of the phrases and the terms not appearing in any of the phrases in a query *Q*. Each combination forms an AQ. The original query consisting of individual terms (i.e., no phrase is used) also forms an AQ. Each AQ must satisfy the following: (1) it contains all the terms in *Q*; (2) its phrases do not overlap; (3) each AQ has a unique set of phrases relative to other AQs.

5 Definition Category (DC) Generation

For each AQ, we break this step into three tasks: (1) *Meaning/Usage Generation*, (2) *Meaning/Usage Merging*, and (3) *DC Generation*. Task 1 uses WordNet/Wikipedia to identify all possible meanings/usages of each concept in AQ. Task 2 first processes each concept without considering other concepts in the AQ. The possible meanings or usages retrieved from Wikipedia/WordNet for each concept are compared and the similar ones are merged. To generate DCs for each AQ in Task 3, we use the meanings/usages (including the merged ones) for different concepts to form different combinations. Each combination contains one meaning/usage from each concept of the AQ and forms one DC. Since Task 3 is straightforward, we will focus on the first two tasks only. Query terms having no entries in either WordNet or Wikipedia are assumed to have *unknown* meanings/usages and will be included in each DC of *Q*.

Merging WordNet meanings: Our synset-merging algorithm consists of six merging rules, each of which gives one condition under which two synsets should be merged. These rules have been reported in [9] and will not be repeated here.
Merging Wikipedia meanings/usages: For *meaning/usage generation* from Wikipedia, we send a concept *C* (phrase/term) to Wikipedia and obtain a returned page P. We consider the following cases based on the type of page that is returned.

Case 1: Concept *C* has a unique meaning/usage, i.e., there is no disambiguation link at the top of P and there are no multiple meaning/usage entries. In this case, we represent the meaning/usage of this concept as a vector of terms with weights calculated based on *tf*idf*. The first *n* (say 20) terms appearing P are used to generate the vector because they usually include the main meaning/usage of a concept.

Case 2: Concept *C* has multiple meanings/usages. Two sub-cases: (1) There exists a disambiguation link at the top of P saying “For other uses, see *C* (disambiguation).” By following that link we can see the possible meanings/usages of the concept. (2) There is no disambiguation link on P and there are multiple definition entries on P. This case can be identified by noticing that P has “*C* can refer to the following …”. In both subcases, for each meaning/usage of the concept, a short definition is provided by Wikipedia. Sometimes the definitions are organized by categories. Each definition and its corresponding category (if exists) are used to generate the vector for the meaning/usage. Higher weights are given to terms that appear in the category labels.

For each *C*, the vector representations of different meanings/usages generated above are compared to see if some vectors should be merged using the following rule:

Rule 1: If the vector representations of two meanings/usages of *C* have common noun words (excluding *C*), then merge them.

Merging WordNet-Wikipedia meanings/usages: In many cases the same meaning or usage may be retrieved from both Wikipedia and WordNet for a given concept even though their vector representations may be different. In these cases, we merge their vector representations for the meaning/usage using the following rules:

Rule 2: If a synonym or a hyponym of a synset *S* in WordNet appears in the definition *D* of the meaning/usage in Wikipedia, then merge *S* and *D* (i.e., they will be merged into one document; same below).

Rule 3: If a definition *D* in Wikipedia and a synset *S* in WordNet have common content words (not counting the concept itself), then merge *D* and *S*.

6 Ranking the Definition Categories

We use two major weight formulas to rank the DCs, one calculates the importance of each AQ and the other estimates the importance of each DC within each AQ.

Alternative Query Weighting:

Importance of each phrase in AQ: We consider three factors:

Phrase frequency: We give higher weights to phrases that appear in more search result records (SRRs), i.e., have higher frequencies.

Well-knownness: We give higher values of *well-knownness* to phrases that are better known and have less ambiguity. In Section 4 we introduced different cases when we send a phrase to Wikipedia and WordNet. In our experiments, the *well-knownness* of a phrase in Case 1 (WordNet or Wikipedia) = 1, that of a phrase in Case 2 (Wikipedia) = 0.5, and those in Case 3 (Wikipedia) and Case 2 (WordNet) = 0.

Phrase length: This is the number of words in a phrase. We use number of words in the query to normalize this weight. Longer phrases usually have less ambiguity.

To summarize, we use the formula below to compute the weight of a phrase *p*:

$$W_p(p) = (df(p)/max\_df + well\text{-}knownness(p) + |p|/|Q|) / 3$$

where $df(p)$ is the phrase frequency of p in SRRs, max_df is the largest phrase frequency in SRRs among all phrases that appear in the set of AQs, $|p|$ and $|Q|$ are the lengths of p and user query Q, respectively.

Importance of each term *t* in AQ: We consider two factors in assigning weight to *t*.

Co-occurrence: We utilize the co-occurrences of *t* with all phrases in AQ among the SRRs. If AQ does not have phrases then we consider the co-occurrences of *t* with all proper names in AQ. We compute the co-occurrence based weight for term *t* as follows: $cow(t) = \sum_{j=1}^{n} nco(t, p_j)$, where *nco*(*t*, *p*) is the number of co-occurrences of term *t* with phrase *p* among the SRRs and *n* is the number of valid phrases in AQ.

Well-knownness: We give higher values of *well-knownness* to terms that are better known and have less ambiguity. This is similar to assigning a well-knownness value to a phrase as discussed earlier.

We use $W_t(t)$ =(*cow*(*t*)/*max_cow*+*well-knownness*(*t*)) / 2 to compute the weight of *t* in AQ, where *max_cow* is maximum co-occurrence weight for all terms in the SRRs.

Importance of each AQ: Let *AQ* be a given alternative query. We add the weights of the terms and phrases in *AQ* and normalize the sum by |*Q*| to obtain the final weight of *AQ* and denote it as $W_{AQ}(AQ)$. Note that the terms here are only those terms that appear in the original query but not in any of the phrases in *AQ*.

Definition Category (DC) Weighting:

For ranking DCs, we consider two types of weights: *meaning/usage weight* (MUW) and *relationship weight* (RW). Recall that each DC is a combination of concepts with each concept bound to a specific meaning/usage. The MUW of a meaning/usage *u* of a concept *C* reflects the likelihood that *u* is the correct meaning/usage for *C* without considering other concepts in the same DC. The RW is used to capture the impact of different relationships between concepts or their usages in the same DC on the likelihood of the DC to be the correct DC for the original query. By giving higher weights to those DCs whose concepts are more closely related, those DCs that make little sense (i.e., whose concepts are not related) will be ranked low.

Meaning/Usage Weighting: Let *u*(*C*, *DC*) denote the specific meaning/usage of concept *C* in a definition category *DC*. In WordNet, each *u*(*C*,*DC*) is represented by up to four components: synonyms, definition, example(s), and domain. In Wikipedia, each *u*(*C*, *DC*) is represented by a set of words, which is called its *definition*. In summary, each *u*(*C*, *DC*) has a *representation*. Let *muw*(*u*(*C*, *DC*)) denote the MUW of *u*(*C*, *DC*). This weight indicates the relative importance of *u*(*C*, *DC*) among all possible meanings/usages of *C*.

For a meaning/usage *u*(*C*, *DC*) from Wikipedia, we compute its weight by:

$$muw(u(C, DC)) = \frac{N - Rank(u(C, DC)) + 1}{N}$$

where *N* is the number of meanings/usages for concept *C* from Wikipedia, *Rank*(*u*(*C*,*DC*)) is the rank (order) of the meaning/usage of *C* in *DC* in the list of these *N* meanings/usages.

For a meaning (i.e., synset) *u*(*C*, *DC*) from WordNet, we use the ratio of the *frequenc-of-use* of *u*(*C*, *DC*) (denoted as *f*(*u*(*C*, *DC*))) to the sum of the frequencies-of-use of all synsets of *C* (denoted as *F*(*C*)) to compute *muw*(*u*(*C*, *DC*)).

Also, concepts with an *unknown* meaning will be given zero weight.

Relationship Weighting: Consider a given *DC* and let $u_1 = u(C_1, DC)$ and $u_2 = u(C_2, DC)$ be the (possibly merged) meanings/usages of concepts C_1 and C_2 in *DC*, respectively. The following three cases are possible: Case 1: Both u_1 and u_2 are from WordNet; Case 2: u_1 is from WordNet and u_2 is from Wikipedia; and Case 3: Both u_1 and u_2 are from Wikipedia.

Furthermore, there are two types of basic relationships: **Type 1** is between two specific meanings/usages such as u_1 and u_2; and **Type 2** is between a specific meaning/usage (e.g., u_1) and a concept (e.g., C_2). Since Type 1 relationships are more specific than Type 2 relationships, we assign a higher weight to the former than to the latter. Currently, all Type 1 relationships have the same weight (denoted *rwt1*) and all Type 2 relationships also have the same weight (denoted *rwt2*). In our experiments, *rwt1* = 2 and *rwt2* = 1 are used. We have identified 6 **Type 1** relationships and 4 **Type 2** relationships. Due to space limitation, they cannot be included here but can be found in [10].

It is possible that more than one basic relationship exists between two specific meanings/usages u_1 and u_2 *of* any two concepts C_1 and C_2 in a given DC. Since satisfying more basic relationships usually indicates a stronger overall relationship between u_1 and u_2, we use the sum of all weights of the basic relationships that exist between u_1 and u_2 as the *overall relationship weight* (ORW) between u_1 and u_2, and denote it as $ORW(u_1, u_2)$.

We now discuss how to compute the weight for a *definition category DC* within the AQ from which the *DC* is derived. In general, a *DC* may contain multiple concepts $C_1, \ldots, C_k$, $k \geq 1$, with each having a specific meaning/usage in the *DC*. If $k = 1$, there will be no basic relationship. If $k > 1$, we need to consider all pairs of meanings/usages. In summary, we use the formula below to compute the weight of defintion category *DC*:

$$W_{DC}(DC) = \begin{cases} muw(u_1), & if \quad k = 1 \\ \sum_{1 \le i < j \le k} ORW(u_i, u_j) * (muw(u_i) + muw(u_j)), & if \quad k > 1 \end{cases}$$

To normalize the weight between 0 and 1, we divide $W_{DC}(DC)$ by the maximum $W_{DC}(DC)$ among all DCs in the corresponding AQ.

Final Definition Category Ranking:

Recall that each DC comes from a particular alternative query (AQ). In our DC ranking model, the importance of the AQ from which a DC is derived can impact the final ranking of the DC. We consider the following two general ways to combine $W_{AQ}(AQ)$ and $W_{DC}(DC)$ to obtain a final ranking for all DCs:

AQ-DC Ordering: First, all AQs are ordered in non-ascending value of $W_{AQ}(AQ)$; next, within each AQ, its DCs are ordered in non-ascending value of $W_{DC}(DC)$. For AQs with the same $W_{AQ}(AQ)$, their DCs are considered together and are ordered in non-ascending value of $W_{DC}(DC)$.

Weighted-Sum: The final weight of *DC*, denoted as $FW_{DC}(DC)$, is computed by:

$$FW_{DC}(DC) = c_1 * W_{AQ}(AQ) + c_2 * W_{DC}(DC) \qquad (1)$$

where c_1 and c_2 are non-negative weight parameters satisfying $c_1 + c_2 = 1$, and each DC in (1) is derived from the AQ in (1). Finally the DCs are ordered in non-ascending value of $FW_{DC}(DC)$.

7 Evaluation

Our dataset contains 50 queries. Among these queries, the numbers of queries having 1, 2, 3 and 4 terms are 9, 28, 10 and 3, respectively. 37 of the 50 queries have at least one phrase and 13 have no phrase; 44 queries are ambiguous (at least one term or phrase has more than one meaning/usage) and 6 are not. We submit each test query to the Yahoo search engine to collect the top 50 search result records.

We report two evaluation tests here: (1) evaluate our method for generating DCs; and (2) evaluate our algorithm for ranking the DCs. In each test, the results generated by the proposed methods are compared against the ideal results (the golden standard) judged by human expert based on the intention and meaning of each test query.

DC Generation Accuracy:

A DC is considered to be correct if it does not contain unrelated meanings/usages and does not miss related meanings/usages from the concepts in this DC. Our results can be summarized as follows: Our algorithm has an average *precision*, *recall*, and *F1-measure* of 0.97, 0.96 and 0.96, respectively. Our results show that our algorithm can provide very accurate DCs for the test queries. When phrases in a query can be recognized, the average *precision*, *recall*, and *F1-measure are 0.98*, *0.99* and *0.99*, showing the importance of recognizing phrase(s) in a query.

DC Ranking Accuracy:

To evaluate the quality of the ranking of DCs produced by our ranking method, the list of ranked DCs by our method σ_1 is compared against the ideal list generated by human expert σ_2. We use *scaled/normalized Spearman's footrule distance* (denoted by *NFr*) [7] to compare two lists of rankings of the same set *S*. It is the sum (over all elements *i* from *S*) of the absolute difference between the ranks of *i* in the two lists divided by some normalization value.

Table 1. NFr Performance of DC Ranking

	AQ-DC Ordering	c_1=0, c_2=1	c_1=0.2 c_2=0.8	c_1=0.5 c_2=0.5	c_1=0.8 c_2=0.2
no phrase*	0.01	0.01	0.01	0.01	0.01
phrase	0.05	0.096	0.078	0.055	0.051
1 term	0.02	0.02	0.02	0.02	0.02
2 terms	0.01	0.01	0.01	0.01	0.01
3 terms	0.01	0.138	0.094	0.025	0.01
4 terms	0.1	0.26	0.19	0.13	0.106
overall	0.02	0.056	0.04349	0.0258	0.022

* This row does not contain results for 1-term queries.

In Table 1, the second column shows the DC ranking result when *AQ-DC Ordering* is used, and columns 3-6 show the results when different weights of c_1 and c_2 (See Formula 1) are used. Note that $c_1 = 0$ means that the ranking is based on DC weight only while the impact of AQ is ignored. It can be observed that the overall accuracy improves when more emphasis is given to the AQ weight, i.e., when c_1 increases. *AQ-DC Ordering* gives the best result. We can see that the accuracy for 4-term queries is noticeably lower than those for other queries; this is due to the lack of enough basic relationships among some of the 4-term queries used. The lower accuracy for 4-term queries also has a negative impact on the accuracy for queries with phrases (see the

3rd row in Table 1). Note that for queries without phrases (e.g., 1-term queries), only one AQ will be generated and thus there is no difference in performance for different ranking methods.

8 Conclusion

In this paper we studied the problem of generating all possible categories of any user query and ranking these categories according to their match with the intention of the user. These categories can be used to categorize search result records returned from search engines in response to user queries. Our approach focuses on leveraging the meanings and usages of terms/phrases in each query and their relationships. We utilize both WordNet and Wikipedia to identify phrases in queries, the basic meanings/usages of each term/phrase, and the basic relationships between each pair of terms/phrases and their meanings/usages.

References

1. E. Agirre, E. Alfonseca, O. Lopez. Approximating Hierarchy-based Similarity for WordNet Nominal Synsets Using Topic Signatures. Global WordNet Conf., 2004.
2. R. Al-Kamha, D. W. Embley: Grouping search-engine returned citations for person-name queries. WIDM 2004.
3. S. Beitzel. Varying Approaches to Topical Web Query Classification. ACM SIGIR, 2007
4. E. de Luca, A. Nürnberger, O. von-Guericke. Ontology-Based Semantic Online Classification of Documents: Supporting Users in Searching the Web. University of Magdeburg, Germany, AMR, 2004.
5. T. de Simone and D. Kazakov. Using WordNet Similarity and Antonymy Relations to Aid Document Retrieval. Recent Advances in Natural Language Processing (RANLP), 2005.
6. G. Demartini, P.-A. Chirita, I. Brunkhorst and W. Nejdl, Ranking Categories for Web Search. ECIR 2008.
7. P. Diaconis, R. L. Graham, Spearman's Footrule as a Measure of Disarray. Journal of the Royal Statistical Society, Series B (Methodological), 39, 1977, 262-268.
8. M. He, M. Cutler, K. Wu. Categorizing Queries by Topic Directory. WAIM 2008.
9. R. Hemayati, W. Meng. Semantic-Based Grouping of Search Engine Results. In Introduction to the Semantic Web: Concepts, Technologies and Applications, edited by Gabriel Fung, iConcept Press, 2010.
10. R**.** Hemayati, W**.** Meng, C**.** Yu**.** Identifying and Ranking Possible Semantic and Common Usage Categories of Search Engine Queries**.** Technical report, 2010.
11. F. Liu, C. Yu, W. Meng. Personalize Web Search by Mapping User Queries to Categories. ACM CIKM, 2002.
12. S. Liu, C. Yu, W. Meng. Word Sense Disambiguation in Queries. ACM CIKM, 2005.
13. R. Mihalcea. Using Wikipedia for Automatic Word Sense Disambiguation. NAACL HLT 2007, pp.196–203.
14. M. Sanderson. Word Sense Disambiguation and Information Retrieval, ACM SIGIR, 1994.
15. S. Patwardhan, S. Banerjee, T. Pedersen. Using Measures of Semantic Relatedness for Word Sense Disambiguation. CICLing 2003: pp 241-257.