Word Sense Disambiguation in Queries

Shuang Liu, Clement Yu
Department of Computer Science
University of Illinois at Chicago
851 South Morgan
Chicago, IL, 60607-7053, U.S.A
{sliu, yu}@cs.uic.edu

Weiyi Meng
Department of Computer Science
Watson School of Engineering
Binghamton University
Binghamton, NY, 13902, U.S.A
meng@cs.binghamton.edu

ABSTRACT

This paper presents a new approach to determine the senses of words in queries by using WordNet. In our approach, noun phrases in a query are determined first. For each word in the query, information associated with it, including its synonyms, hyponyms, hypernyms, definitions of its synonyms and hyponyms, and its domains, can be used for word sense disambiguation. By comparing these pieces of information associated with the words which form a phrase, it may be possible to assign senses to these words. If the above disambiguation fails, then other query words, if exist, are used, by going through exactly the same process. If the sense of a query word cannot be determined in this manner, then a guess of the sense of the word is made, if the guess has at least 50% chance of being correct. If no sense of the word has 50% or higher chance of being used, then we apply a Web search to assist in the word sense disambiguation process. Experimental results show that our approach has 100% applicability and 90% accuracy on the most recent robust track of TREC collection of 250 queries. We combine this disambiguation algorithm to our retrieval system to examine the effect of word sense disambiguation in text retrieval. Experimental results show that the disambiguation algorithm together with other components of our retrieval system yield a result which is 13.7% above that produced by the same system but without the disambiguation, and 9.2% above that produced by using Lesk's algorithm. Our retrieval effectiveness is 7% better than the best reported result in the literature.

Categories and Subject Descriptors

H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing – *Dictionaries, Linguistic Processing* H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval – *Retrieval Models, Relevance Feedback, Query Formulation* I.2.7 [**Artificial Intelligence**]: Natural Language Processing – *Text Analysis*

General Terms

Algorithms, Experimentation, Languages

Keywords

Information Retrieval, Word Sense Disambiguation, WordNet

CIKM'05, October 31–November 5, 2005, Bremen, Germany.
Copyright 2005 ACM 1-59593-140-6/05/0010…$5.00.

1. INTRODUCTION

Word sense disambiguation is the process of choosing the right sense for a word in its occurring context [MS99, JM00]. Robust word sense disambiguation systems using machine learning approaches [Y95, S98, M02] and dictionary based approaches [L86, PBP03, MTF04] have been developed in the past. In spite of these advances, the accuracy of disambiguation is still rather low. The best reported result for word sense disambiguation is 71.2% [M02].

In information retrieval, adding appropriate synonyms and hyponyms to a query can improve retrieval effectiveness [BR99, GVC98, LLYM04, RS95, V94, YM98]. However, some query term has multiple meanings and adding a synonym of the query term which has a different meaning in the context of the query would cause deterioration in retrieval effectiveness. Therefore, determining the sense of each query term is essential for effective retrieval. Once a query term's sense in a query context is determined, synonyms and hyponyms with the same meaning as that of the query term are added to the query so that documents having these synonyms and hyponyms but not the actual term may be retrieved. In the past decade, experiments involving addition of terms to queries and word sense disambiguation have shown rather disappointing results [S94, SOT03, SP95, V93]. This is due to inaccurate disambiguation and incorrect adding of new terms. However, our previous work [LLYM04] shows a promising result of applying word sense disambiguation to information retrieval by automatic query expansion. In this paper, we propose a more elaborate approach to disambiguate word senses in short queries by using WordNet [M90] and the Web. Our experimental results on the most recent TREC query set of 250 queries show that our approach has about 90% accuracy.

Given a query containing multiple words, our aim is to find the precise meaning (sense) of each word in the context of other query words. If the query consists of a single word and the word has multiple meanings, it is usually not possible to determine the sense of the query word. Thus, we concentrate on multi-word queries. A high-level description of a word-sense disambiguation algorithm is as follows.

We first determine the noun phrases of the query [SLLYM05]. Other non-noun phrases may be detected, but noun phrases are of much higher significance in word sense disambiguation. Phrase recognition has been thoroughly studied in [BMSW97, C98, CM02, TZM96] and it is beyond the scope of this paper.

In WordNet, there are several pieces of information associated with each content word and they can be used for word sense disambiguation. These information include its synonyms, hyponyms, hypernyms, definitions of its synonyms and hyponyms, and its domains. By comparing these pieces of information associated with the words which form a phrase, it may be possible to assign senses to these words.

If the above process fails to identify the sense of a query word, then other query words, if exist, are used by going through exactly the same process. If the sense of a query word cannot be determined in this manner, then a guess of the sense of the word is made, if the guess has at least 50% chance of being correct. In WordNet, the meaning or sense of each word is defined by a set of synonyms (called synset) and there is a frequency of use associated with each synset. Based on the frequency information, it is possible to determine if a particular sense of a word is used with at least 50% chance or not. If no sense of the word has 50% or higher chance of being used, then we apply a Web search to assist in the word sense disambiguation process. Thus, our word sense disambiguation process consists of three steps:

Step (1) Utilize WordNet to provide synonyms, hyponyms, their definitions, and other information to determine senses of query terms. If the senses of all query words can be determined, then terminate.
Step (2) Employ the frequencies of use of the synsets supplied by WordNet to make a guess of the senses of query words whose senses have not been determined, if the chance of success is at least 50%.
Step (3) For those words whose senses have not been determined, apply a Web search for the sense determination.

In this paper, we use word sense disambiguation to improve retrieval performance in two aspects. First, it helps bring in new terms and phrases to the query. Suppose *w* is a term, and (*w w'*) is a noun phrase in a query. After the sense of *w* is determined, the selectively chosen synonyms, hyponyms, similar words, and compound concepts of *w* are added to the query. New terms that are brought in by *w* form phrases with *w'* or terms brought in by *w'*. Second, we assign an additional weight to a feedback term if it can be semantically related to some disambiguated query term.

One contribution of this paper is a process which yields high accuracy of determining the senses of words in short queries. This type of queries occurs frequently in the Web search environment. Another contribution is that our experimental results show that word sense disambiguation can improve retrieval performance.

This paper is organized as follows: the first part of Section 2 describes the various cases where the senses of words can be determined using synonyms, hyponyms and their definitions which are supplied by WordNet (Step (1) as indicated above.) Since a query word may satisfy the conditions of different cases, it may be determined to have multiple senses. The second part of Section 2 describes how to choose the best sense if such a situation arises. The description on the use of frequency information to guess the sense of a query word is given in Section 3 (Step (2)). In Section 4, a Web search is performed to assist the word sense disambiguation process (Step (3)). In Section 5, experimental results are given to demonstrate that our disambiguation approach yields high accuracy and significant improvement over the traditional dictionary based method. In addition, by applying our word sense disambiguation algorithm to text retrieval, significant improvements are obtained in 5 TREC collections. Our system using word sense disambiguation algorithm performs 13.7% better than the same system but not using any disambiguation algorithm, and 9.2% better than the same system using Lesk's algorithm. The overall performance is 7% better than the best result reported in the literature. A comparison with previous works is given in Section 6. Concluding remarks are given in Section 7.

2. CASE ANALYSIS AND CASE CONFLICT RESOLUTION

A term *w* has possibly many sets of synonyms, with each set of synonyms representing a meaning of the word. It also has a definition associated with each set of synonyms; the definition explains the meaning of the word; and it may be followed by one or more examples. For each meaning, the word may have a number of hyponym synsets. Each hyponym synset consists of a set of words or phrases which have the same meaning but are narrower than *w* in a specific sense. The hyponym synset has a definition and possibly some examples. For each meaning, the word may belong to a domain.

Thus, a term *w* can be associated with $I(w) = \{S(w)_i, D(w)_i, E(w)_i, \{H(w)_{ij}\ D(H(w)_{ij})\ E(H(w)_{ij})\}\ Dom(w)_i\}$, where $S(w)_i$ is the set of synonyms associated with the *ith* sense of *w*, $D(w)_i$ is the definition of this synonym set, $E(w)_i$ is the set of examples given to illustrate the use of the term in this sense, $H(w)_{ij}$ is the *jth* hyponym synset of the *ith* sense of *w*, $D(H(w)_{ij})$ is its definition, $E(H(w)_{ij})$ is the set of examples associated with this hyponym synset, and $Dom(w)_i$ is the domain associated with the *ith* sense of *w*. As an example, "crime" has 2 senses. Sense 1 has the set of synonyms containing "crime" and "law-breaking"; its definition is "an act punishable by law; usually considered an evil act"; an example is "a long record of crimes". Even within this sense, it has numerous hyponym synsets. One hyponym synset is {attack, attempt} and the definition of this hyponym synset is "the act of attacking". An associated example is "attacks on women increased last year". The word "crime" belongs to "criminal law" domain.

Our aim is to identify for the term *w* the specific synonym set $S(w)_i$ and if possible certain hyponym synsets $\{H(w)_{ij}\}$ which represent the meaning (or narrower meanings) of *w* for the query using the information *I(w)* as well as the information *I(w')* for a term *w'* which forms a phrase with *w* in the query. In most cases, the disambiguation of a term *w* depends on *w'*. Thus, for ease of presentation, we restrict our discussion to the comparison of *I(w)* with *I(w')*.

There are seven pieces of information associated with each of the two terms *w* and *w'*: the synonym sets, their definitions, examples associated with the definitions, the hyponym synsets, their definitions, their examples, and their domains. The examples of the synonyms and those of the hyponyms and the domain information are usually less significant. Thus, for ease of presentation, we concentrate our discussion on the use of the remaining four pieces of information for sense disambiguation.

In section 2.1, we compare the four pieces of information associated with *w* with the four pieces of information associated with *w'*. For each of these 4×4 pair-wise comparisons plus the domains comparison, we examine if *w* and *w'* are related in some way. Among these 17 comparisons, 6 cases are symmetrical with respect to *w* and *w'*. A relationship between *w* and *w'*, if exists in any of the remaining 11 cases, may be used to assign senses to one or both of these two terms. Due to space limit, detailed analysis is provided for some cases only. Since it is possible that a term may satisfy multiple cases (to be described in section 2.1) and therefore may obtain multiple senses, we describe a method to resolve the conflict in section 2.2.

2.1 Case Analysis

Case 1: *w* and *w'* have the same part of speech and they have a common synonym *w''*.

In this case, the sense of *w* is determined to be the synset containing *w* and *w''* and the sense of *w'* is the synset containing *w'* and *w''*. Brill Tagger [Brill] is used to assign part of speech.

Case 2: *w* or one of its synonyms appears in the definition of the *jth* sense of w', $D(w')_j$.

Subcase 1: *w* appears in $D(w')_j$ and both *w* and its occurrence in $D(w')_j$ have the same part of speech.

In this case, the sense of *w'* is determined to be its *jth* sense, but the sense of *w* has yet to be determined.

Example 1: A query is "health and computer terminal" where "computer terminal" is a simple phrase. "Computer" occurs in the definition of sense 3 of "terminal", and they have the same part of speech, then the sense of "terminal" is determined to be sense 3.

Subcase 2: A synonym of the i_{th} sense of *w* appears in $D(w')_j$ and both the synonym of *w* and its occurrence in $D(w')_j$ have the same part of speech.

In this case, the sense of *w* is determined to be its i_{th} sense and that of *w'* is determined to be its j_{th} sense.

Example 2: A query is "Greek philosophy Stoicism". A synonym in a synset of "philosophy" is "philosophical system". The phrase "philosophical system" appears in the definition of a sense of "Stoicism". Both occurrences of "philosophical system" are noun phrase, thus "philosophy" and "Stoicism" are disambiguated.

In the above two subcases, *w* and its synonym, and their occurrences in the definition of *w'* have the same part of speech. We call this a full matching. It is possible that the occurrences of *w* and its synonym in the definition of *w'* have different parts of speech. This is called a partial matching. Here is an example.

Example 3: A query is "term limit". "Limit" appears in the definition of sense 2 of "term". "Limit" is a noun in the query, but it is a verb in the definition.

This type of partial matching also occurs in other cases. For simplicity, we leave out its description for the following cases.

Case 3: There are content words in common between the definition of one sense of *w*, $D(w)_i$ and the definition of one sense of *w'*, $D(w')_j$.

Since the definition of a term may contain quite a few words, it is not rare that multiple pairs of the definitions of *w* and *w'* have words in common. To ensure that the common words are significant, they need to be content words and have the same parts of speech. If there are still more than one pair of definitions which have common content words, choose the pair which has the largest number of common content words. An alternative is to place more emphasis on consecutive content words in common. It should be noted that if a common content word is too general, i.e. it is a verb, adjective or adverb and it has numerous senses (say more than 10) in its part of speech, then the common word should be ignored. Another restriction is as follows. Suppose *w* has *n* senses. If a common content word is *h* which is a hypernym of *m* ($m < n$) senses of *w* but it is not a hypernym of any sense of *w'*, then *h* can be counted as a common word towards differentiating the different senses of *w'*; it can also be used to differentiate the *m* senses of *w* (having the hypernym *h*) from the *(n-m)* senses of *w* (not having the hypernym *h*), but it is not useful in differentiating the m senses of *w*. Another situation in which the senses of a term are restricted but not completely disambiguated is that a common content word appears in multiple definitions of the term.

Example 4: A query is "induction and deduction". "Induction" and "deduction" have multiple senses. A pair of definitions of "induction" and "deduction" has the words "reasoning" and "general" in common. These common words have the same parts of speech in the two definitions. This pair of definitions has the largest number of common content words. As a consequence, the senses of the two query terms are determined by these definitions.

Example 5: A query is "Oscar winner selection". Both "winner" and "selection" have multiple senses. The definition of a sense of "winner" has the word "person" in common with the definition of a sense of "selection". However, "person" is a hypernym of all senses of "winner". Thus, although the sense of "selection" is determined to be the sense whose definition contains the word "person", the sense of "winner" has yet to be determined.

Case 4: The term *w* or a synonym of *w* occurs in the definition of a hyponym synset *S* of w', $D(H(w')_{ij})$.

Subcase 1: *w* occurs in the definition of *S*, with both occurrences of *w* having the same part of speech.

Let *K* be the sense of *w'* whose hyponym synset is *S*. Consider each synset *H* (which is a hypernym synset of *S*) lying in the path between *K* and *S*, including the two endpoints. In this case, the senses of *w'* and its descendants from *K* to *S* are determined, while the sense of *w* has yet to be determined.

Example 6: A query is "tropical storm". "Storm" has a hyponym synset consisting of a word "windstorm", which has a hyponym synset consisting of a word "cyclone". In turn, "cyclone" has two hyponym synsets, one of which containing the word "typhoon" and the other containing the word "hurricane". The definitions of both terms contain the word "tropical". Thus, the senses of "storm", "windstorm" and "cyclone" are determined as their senses are decided by their hyponyms "typhoon" and "hurricane".

Subcase 2: A synonym of *w* in the k_{th} synset of *w* occurs in $D(H(w')_{ij})$, with both occurrences of the synonym having the same part of speech.

In this case, not only the sense of *w'* and its descendants down to *S* are determined, but the sense of *w* is determined to be its k_{th} sense.

Example 7: A query is "Toronto film award". "Motion picture" is a synonym in a synset, *G*, of "film"; it also appears in the definition of a hyponym synset of "award", containing "Oscar", both occurrences of "motion picture" being a noun phrase. Thus, the sense of film is determined to be that of *G*, and the sense of "award" is determined to be the sense having hyponym "Oscar".

Case 5: A word in the hyponym synset, *H*, of *w* is *w'* or is a synonym of *w'* and the occurrences of the word have the same part of speech.

Subcase 1: A word in the hyponym synset, *H*, of *w* is *w'*.

In this case, the sense of *w* is the synset, *P*, which contains the term *w* and is a hypernym synset of *H*. All synsets which are hypernyms synsets of *H* from *H* to *P* (and including *H* and *P*) are the senses to be used. The sense of *w'* has yet to be determined.

Example 8: A query is "tobacco cigarette lawsuit". "Tobacco" has a hyponym synet *H* which contains the word "cigarette". Thus, the sense of "tobacco" is the synset *P* which contains "tobacco" and has the hyponym synset *H*; the synsets from *H* to *P* are determined to be related to the query.

Subcase 2: A term in the hyponym synset, *H*, of *w* is a synonym of *w'* in the synset *S(w')*.

In this case, the sense of *w'* is determined to be *S(w')*; the sense of *w* is the synset which contains *w* and is a hypernym synset of *H*.

Case 6: A term *t* in the hyponym synet, *H*, of *w* appears in the definitions of several hyponym synsets of *w'* and the occurrences of *t* have the same part of speech.

In this case, the synset of *w*, say *P*, containing the hyponym synset *H*, determines the sense of *w*. All synsets from *H* to *P* are related to the query. The senses of *w'* are restricted to the multiple hyponym synsets of *w'* whose definitions contain *t*. If these hyponym synsets belong to one synset of *w'*, then this synset of *w'* is the determined sense of *w'*.

Example 9: A query is "alcohol consumption". "Consumption" has a hyponym synset *H* containing the word "drinking", which occurs in the definitions of several hyponym synsets of "alcohol". Thus, the sense of consumption is determined to be the synset (of "consumption") *P* containing *H* and synsets from *H* to *P* are related to the query. "Alcohol" has two senses, both of which have hyponym synsets whose definition contains "drinking". Thus, the sense of "alcohol" has yet to be determined.

Case 7: A term *t* in the hyponym synset, *H*, of *w* appears in the definitions of one or more synsets of *w'* and the occurrences of *t* have the same part of speech.

In this case, the synset of *w*, say *P*, containing the hyponym synsets is the determined sense for w. All synsets from *H* to *P* have determined senses and are related to the query. The senses of *w'* are restricted to the multiple synsets of *w'* whose definitions contain *t*. If there is only one such definition, the sense of *w'* is determined.

Example 10: A query is "Greek philosophy Stoicism". The definition of one sense of "Stoicism" contains "teaching", which is a hyponym of "philosophy", with both "teaching" having the same part of speech. Thus, the sense of "stoicism" and "philosophy" are determined.

Case 8: There are content words in common between the definition of a synset, say *S*, of *w* and the definition of a hyponym synset , say *S'*, of *w'*. (This is very similar to Case 3.)

Case 9: There is a common term in a hyponym synset of *w* and a hyponym synset of *w'*.

Case 10: There are content words in common between the definition of a hyponym synset of *w* and that of a hyponym synset of *w'*. (This is very similar to Case 3.)

Case 11: One or more senses of *w* and *w'* belong to the same domain.

Example 11: A query is "white collar crime sentence". The first sense of "crime" belongs to the domain "criminal law", and the second sense of "sentence" belongs to the same domain. Then the senses of "crime" and "sentence" are determined to be the first sense of "crime" and the second sense of "sentence" respectively.

2.2 Case Conflict Resolution

There are situations where a term *w* may satisfy the conditions of several cases. In such a situation, *w* may be assigned different senses. To resolve the conflict, the information needs to be aggregated in order to make a determination. Suppose the sense of the term is determined to be its sense s_i due to its satisfaction of the conditions in Case *k* involving a sense of term *v*, but its sense is also determined to be the sense s_j due to its satisfaction of the condition in a different case, say Case *t*, involving either the same sense of term *v*, a different sense of term *v*, or a sense of another term *v'*. Then, the sense s_i is chosen, if the Case *k* historically has a higher accuracy than Case *t*. More elaborate solution consists of utilizing the frequencies of use of the term *w* in sense s_i and sense s_j, as well as the accuracies of the cases involved in determining the senses of the terms *v* and *v'*. More precisely, for each sense of the term *w* which satisfies the conditions of some cases, the likelihood that this sense is chosen is a function of three parameters: (1) the historical accuracy of the case in determining the sense of *w* (this is referred to as the case weight in Section 2.2.1); (2) the frequency of use of the sense of *w*, which decides the likelihood that the term is used in this sense (this will be referred to as the sense weight in Section 2.2.2); and (3) the historical accuracy of the case which determines the sense of *v* (this is referred to as the supporting weight, as *v* is used to support for the disambiguation of the term *w*). Then, the sense with the largest total likelihood is chosen. A detailed description is given as follows.

2.2.1 Case Weight

The case weight is given by the historical accuracy of each case. Let $C_{K\text{-}F}$ and $C_{K\text{-}P}$ represent the full match and partial match cases respectively, where *K* varies among the 11 cases. The accuracy is:

$$case\_wt(C_{K-X}) = \frac{\#words\ C_{K-X}\ correctlly\, disambiguates}{\#words\ C_{K-X}\ disambiguates} \quad (1)$$

where *X* is *F*(ull) or *P*(artial). To avoid a weight of 0, 0.01 will be automatically assigned to a case if there is no training data for it. Additionally, the weight of each case is normalized so that the sum of weights is 1.

The set of queries is divided into 5 subsets; each containing 50 queries (see Section 5). For each 4 subsets of queries, the weights of the cases are determined and then an average weight of each case (one for full matching and another for partial matching) is obtained. (We take this elaborate process is that the accuracy of our disambiguation is determined by a 5-fold cross validation.)

We note that in Table 1, the weight of a full matching case is usually significantly higher than that of the corresponding partial matching case, indicating that a full matching usually yields higher accuracy. An exception occurs in Case 6, possibly due to insufficient number of terms satisfying that case.

2.2.2 Sense Weight

The sense weight of a sense of term *w* is given in formula (2):

$$sense\_wt(w, s_i) = f(w, s_i) / F(w) \quad (2)$$

Table 1. Cases Weights

	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11
Full	0.001	0.083	0.069	0.081	0.091	0.071	0.084	0.066	0.091	0.080	0.091
Partial	0.001	0.037	0.012	0.001	0.001	0.077	0.032	0.001	0.028	0.001	0.001

Table 2. A Disambiguation Example

w=”terminal” *Cases*	*S_1* *sense_wt(terminal,s_1) = 0.5*	*S_3* *sense_wt(terminal,s_3) = 0.25*
C_{2-F} case_wt = 0.083		*sp_wt(computer) =1*
C_{4-F} case_wt = 0.081		*sp_wt(computer) = 1*
C_{8-F} case_wt = 0.066	*sp_wt(computer, s_1) = 0.146*	
C_{10-F} case_wt = 0.080	*sp_wt(computer, s_1) = 0.146*	
disam_wt(terminal,s_i)	*$0.5\times(0.066\times0.146+0.080\times0.146) = 0.0107$*	*$0.25\times(0.083\times1+0.081\times1)$ =* ***0.041***

where *f(w, s_i)* is the frequency of use of the sense s_i of *w* and *F(w)* is the sum of the frequencies over all senses of *w*.

Example 12: The noun “terminal” has 3 senses. The sense weight of each sense is given as follows: *sense_wt(terminal, s_1) = 0.5; sense_wt(terminal, s_2) = 0.25; sense_wt(terminal, s_3) = 0.25.*

2.2.3 Supporting Weight

Suppose the term *w* is disambiguated to sense s_i using term *v* with sense v_j. We want to assign the supporting weight to term *v* with sense v_j, which reflects the historical accuracies of the cases which are used to disambiguate *v* to sense v_j. This supporting weight, denoted by *sp_wt(v_j)*, is simply the sum of the weights of the cases which disambiguate *v* to v_j.

$$sp\_wt(v_j) = \sum_{C_{T-X}\ choses\ v_j} case\_wt(C_{T-X}) \qquad (3)$$

If there are multiple senses of *v* that support the disambiguation case, then the supporting weight of multiple senses of *v* is the sum of each sense’s supporting weight. If *v* cannot be disambiguated but *v* can be used to disambiguate *w*, then the supporting weight of *v* is 1, as all senses of *v* can be used for the disambiguation.

Example 13: A query is “health and computer terminal”. “Computer” has 2 senses and can be disambiguated to sense 1 by case C_{8-F}, and C_{10-F}. The supporting weight of “computer” and its senses are: *sp_wt(computer) = 1, sp_wt(computer, s_1) = case_wt(C_{8-F})+ case_wt(C_{10-F}) = 0.146, sp_wt(computer, s_2) = 0.*

2.2.4 Maximize the Likelihood

If several cases are used to disambiguate the term *w* to sense s_i by utilizing different senses of *v*, then the disambiguation weight of sense s_i of term *w*, which is the likelihood that *w* is disambiguated correctly, is given as in formula (4):

$$disam\_wt(w,s_i) = sense\_wt(w,s_i)\times \sum_{C_{K-X}\ choses\ s_i} (case\_wt(C_{K-X})\times[\sum_{v_j\ supports\ s_i} sp\_wt(v_j)]) \qquad (4)$$

The sense with the maximum disambiguation weight will be chosen as the sense of *w* as in formula (5):

$$sense(w) = \arg\max_{s_i}[\ disam\_wt(w,s_i)] \qquad (5)$$

2.2.5 An Example

A query is “health and computer terminal” where “computer” and “terminal” form a simple phrase. The word “terminal” can be disambiguated to 2 senses s_1, and s_3 by 4 cases. Table 2 shows its disambiguation information. Cases weights in Table 1 are used.

In Table 2, the first row gives the sense weights of the 2 involved senses of “terminal”. The leftmost column lists disambiguation cases and their weights. The other entries in Table 2, except the last row, are the supporting weights from “computer” for the different cases. For example, *sp_wt(computer, s_1)* in row 4 and column 2 is the supporting weight of sense s_1 of “computer”; in case C_{8-F}, sense s_1 of “computer” is used to disambiguate the word “terminal” to sense s_1. *sp_wt(computer)* in row 2 and column 3 is the supporting weight of word “computer”; in case C_{2-F}, “computer” is used to disambiguate the word “terminal” to sense s_3. The last row gives the disambiguation weights of 2 senses of “terminal”. Since sense s_3 of “terminal” has the highest disambiguation weight, it is the chosen sense.

3. GUESS THE SENSE OF A TERM BASED ON ITS FREQUENCIES OF USE

Suppose none of the cases identified in section 2 is satisfied by a term *w*. In that situation, a guess based on the frequencies of use of the senses of *w* can be made.

In WordNet, the sense of each term is associated with its frequency of use. The higher the frequency of use of a sense of *w*, the more likely that this sense is used in the absence of other information. Suppose the sum of the frequencies of use of the senses of *w* is *x* and the first sense (which is the sense with highest frequency of use) has frequency >= ½ *x*, then using sense 1 without any additional information has at least 50% chance of being right. The first sense is called a dominant sense of the given term.

Example 14: An adjective “modern” has 5 senses. The overall frequency of use is 77. The first sense of “modern” has a frequency of use 62, which is greater than half of the overall frequency. So the first sense of “modern” is the dominant sense.

4. WEB ASSISTED DISAMBIGUATION

If no case can be applied to disambiguate a query term *w* (as described in Section 2) and the frequency of use of the first sense of *w* is lower than 50% of the sum of frequencies of use of all senses of *w* (as described in Section 3), then a Web search engine such as Google may be employed to disambiguate the sense of the query term. First, the query is submitted to Google and the top 20 documents are retrieved. For each such document, find a window of *y* words, say 50, which contains all query terms. Then, all content words in the window, with the exception of the term to be disambiguated, namely *w*, are used to form a vector. The vectors from the windows of the top 20 documents are put together to form a vector *V*. The definition of each sense of the term also forms a vector. The sense of the term whose vector has the highest similarity (say, using the standard cosine function) with *V* is the determined sense of *w*. Here is an example.

Example 15: A query is "Islamic Revolution", in which the word "revolution" cannot be disambiguated by any case nor by the frequencies of use. It has the following 3 senses in WordNet: {revolution -- a drastic and far-reaching change in ways of thinking and behaving}, {revolution -- the overthrow of a government by those who are governed}, and {rotation, revolution, gyration -- a single complete turn (axial or orbital)}. Each definition of revolution forms a vector. Let them be *V1*, *V2* and *V3*. We first submit the query to the web and get the top ranked 20 documents to extracts words to form a vector *V*. By computing the similarity between *V* and *Vi* using the cosine similarity function, it is found that *V2* has the highest similarity, and sense 2 is chosen as the correct sense for "revolution".

5. EXPERIMENT EVALUATION

5.1 Experiment on WSD

5.1.1 Experiment Setup

Experiments are performed on the most recent TREC queries in the robust track. This set consists of 250 queries [V04]. Each query has three portions: 1) title, which is a short query, 2) description, which describes the intention of the user in more detail, and 3) narrative, which gives the characteristics of the relevant documents of the query. Since the title portion resembles a typical user query, we only use the title queries in all experiments. A 5-fold cross validation is used to compute the disambiguation accuracy. That is, the set of queries is divided into 5 subsets; each containing 50 queries. Each four subsets of queries are used as the training data to obtain the cases weights, which are applied to the remaining subset to compute the disambiguation accuracy. This is repeated for the 5 sets each containing 4 subsets of queries and the average disambiguation accuracy is computed. Two word sense disambiguation algorithms employed in the experiments are sketched as follows:

Lesk: The Lesk's algorithm [L86] is one of the earliest word sense disambiguation algorithms applied in open text. The main idea behind the original algorithm is to disambiguate word senses by finding the overlap among their sense definitions. This corresponds to Case 3 as described in Section 2.

Our Algorithm: The 3-step disambiguation algorithm.

5.1.2 Result

There are 258 unique sense terms and 333 ambiguous terms in the 250 queries. Table 3 shows the performances of the two algorithms over 333 ambiguous terms on 250 queries.

The Lesk's algorithm can be applied to only 79 ambiguous terms, which is only 23.7% of 333 ambiguous terms. Its accuracy for these 79 ambiguous terms is 77%. In our algorithm, 217 terms can be handled by case analysis and case conflict resolution as described in section 2; the applicability is 65%, which is 174% more than that of the Lesk's algorithm. 194 of them are correctly disambiguated. The disambiguation accuracy is 89.4%. Out of the remaining 116 terms, 100 terms have dominant senses, which is 30% of the ambiguous terms, and 93 of them are correct, so the accuracy is 93%. We use web-assisted disambiguation method to disambiguate the remaining 16 terms, which constitute 5% of the ambiguous terms. 13 of them are corrected disambiguated; the disambiguation accuracy is 81%. In summary, our method can be applied to all 333 ambiguous terms and the overall disambiguation accuracy is 90%.

Table 3. Disambiguation Evaluation of Different Algorithms

	Lesk	Our algorithm		
		cases	frequency	Web
disam terms #	79	217	100	16
applicability	23.7%	65%	30%	5%
correct terms #	61	194	93	13
Accuracy	77%	89.4%	**93%**	81%
overall accuracy	18%	**90%**		

5.2 Experiment on Retrieval Effectiveness

5.2.1 Our Retrieval Model

Noun phrases in queries are automatically identified by [SLLYM2005] and used for retrieval. Our retrieval model is somewhat non-traditional in the sense that we consider that phrases are more important than individual words. As a consequence, phrase matching between a query and a document has higher priority than term matching [LLYM04]. Disambiguated query terms bring in new terms from WordNet. New phrases are also formed by using the new terms.

We also use pseudo-feedback [BR99, YM98] to expand query. The basic idea is to assume a small number of top ranked documents from initial retrieval to be relevant, and use them to refine the original query. Specifically, for each query, its top 20 documents are assumed to be relevant and the most correlated 20 terms are extracted from these documents. Additionally, a web-assisted pseudo-feedback is also adopted to get more expansion terms in our system [K03, LSY04, YCCLT03]. An important technique we used is to assign additional weights to feedback terms that are semantically related to the disambiguated query terms.

5.2.2 Experiment Setup

Experiments are performed on the TREC disk 4 and 5 (except for the Congressional Record) collections. The same 5 sets of queries described in section 5*.1.1* are used except query 672 which has no relevant documents. 150 queries are from TREC6, TREC7 and TREC8 ad hoc task; 99 queries are from TREC12 and TREC13 robust task. For the same reason as we described in section *5.1.1*, only the title portion of each query is used. We set up 3 sets of experiments over the 249 queries to examine how word sense disambiguation affects retrieval effectiveness. The three sets of experiments are denoted as follows:

NDisam: no disambiguation is performed on ambiguous terms.
LeskDisam: Lesk's algorithm is used to disambiguate the ambiguous terms.
OurDisam: our disambiguation algorithm is used.

These 3 sets of experiments follow the same retrieval model as we described in section *5.2.1*.

5.2.3 Result

Table 4 shows the retrieval performances of executing 5 TREC query sets on the document collections when adopting none or different disambiguation algorithms. The results are measured by mean average precision (MAP) of top 1000 documents for each query. 50 extremely difficult queries that are picked from TREC6, TREC7 and TREC8 query sets are measured separately in the table. For easy comparison with the latest TREC (TREC13) robust task, besides the overall performance of 249 queries from 5 query sets, we also have the overall performances of the 200 queries (among the 250 queries) from TREC6 TREC7, TREC8, and TREC12. Improvements in average precision of each disambiguation algorithm relative to the baseline are given next to the average precision. They are followed by the disambiguation accuracy.

Table 4. Retrieval Performance by MAP

	NDisam	**LeskDisam**			**OurDisam**		
	MAP	MAP	Acc (%)	Imp (%)	MAP	Acc (%)	Imp (%)
TREC6	0.2805	0.2898	78	3.3	0.3280	91	17.0
TREC7	0.2522	0.2815	**100**	**11.6**	0.3093	**92**	**22.6**
TREC8	0.2896	0.2888	62.5	-0.3	0.3225	**92**	11.4
TREC12	0.3714	0.3825	72	3.0	0.4105	88	10.5
TREC13	0.3798	0.3938	84	3.7	**0.4177**	88	10.0
Hard 50	0.1791	0.1897	83	6.0	**0.2056**	**92**	14.7
Old 200	0.2983	0.3106	75	4.1	**0.3426**	91	14.9
Overall	0.3144	0.3270	77	**4.0**	**0.3574**	**90**	**13.7**

As indicated in Table 4, when the disambiguation accuracy is low for LeskDisam (62.5%, TREC8), the retrieval effectiveness deteriorates. But, when the disambiguation accuracy is high (100%, TREC7), the improvement in retrieval effectiveness is significant (11.6%). Since our disambiguation algorithm maintains a high accuracy (88% or higher), the improvement in retrieval effectiveness is significant (10% to 22.6%). Note that the improvement in retrieval effectiveness, though significantly influenced by the disambiguation accuracy, is not proportional to it. The reasons are (a) the percentage of words which are disambiguated is another factor; (b) the number of terms which are brought in by the disambiguated words also impacts retrieval effectiveness.

The best-known average precision result in the total 249 queries was 0.3333 [K04, V04]. Our result is 0.3573, which is 7% better. The best-know result in the old 200 queries was 0.3165 [K04, V04]. Our result is 0.3426, which is 8% better. The best-known result in the extremely hard 50 queries is 0.1941 in our last year's TREC result [LSY04, V04]. Now our result is 0.2056, which is 5.9% better. In TREC13, the best-known result is 0.4019 [K04, V04]. Our result is 0.4177, which is 4% better.

6. RELATED WORK

A number of methods using WordNet to help word sense disambiguation have been proposed in the past few years. [M02] presents a bootstrapping approach. This approach first learns patterns from a sense-tagged corpus SemCor [MLRB93], WordNet definitions, and a generated corpus; then it disambiguates word senses by using the learned patterns with automatic feature selection. While using WordNet, in addition to the examples in each word sense definition, a word's synonyms, hypernyms or hyponyms having unique sense are used; words are disambiguated based on synonym and hypernymy relations only. The accuracy of disambiguation is reported to be 71.2%.

[MTF04] proposes an unsupervised knowledge-based word sense disambiguation algorithm. It applies PageRank-style algorithms to a WordNet-based concepts graph, in which word senses are vertices and relations are edges. The PageRank algorithm is applied to this graph to get the highest rank "node". Additionally, the definitions of the context words and the first sense of each word in WordNet are used in conjunction of the page rank algorithm. The accuracy of disambiguation is reported to be 70.32%.

[PBP03] gives a class of word sense disambiguation algorithms based on semantic relatedness measured by using WordNet. The algorithm with the best performance is an extended Lesk's algorithm which uses the definitions of words to be disambiguated as well as the definitions of their hyponyms. The accuracy is reported to be 39.1%.

Compared with above algorithms, our method has the following differences: first and the most important, we use WordNet in a more elaborated way: we explore synonyms, hyponyms, their definitions, hypernyms, and domain information. In addition, whenever a term can be disambiguated into different senses, a rather elaborate scheme is invoked to determine the proper sense. We also use the dominant sense of each term as well as the Web in case the sense of a term cannot be determined. The accuracy of our method is 90%, which is significantly higher than what has been reported in the literature. However, we note that the query collection we utilize in our experiment is different from the data used by other researchers. Both [MTF04, PBP03] disambiguate certain words in documents instead of queries.

Applying word sense disambiguation to information retrieval has been investigated for a long time. The effects of word sense disambiguation in information retrieval are well discussed in [S94]. [SOT03] disambiguates word senses by applying collocations, co-occurrence statistics, and prior sense frequency in a stepwise way. Documents and queries are represented with sense vectors, and documents are retrieved using the traditional $tf \times idf$ term weighting method. There are two potential problems in their system: first, their supervised disambiguation algorithm uses the sense-tagged corpus SemCor that limits the number of applicable words; second their retrieval effectiveness is not good (MAP 0.055). [KSR04] applies word sense disambiguation to information retrieval as well. They disambiguate word senses coarsely but consistently for documents and queries terms by using co-occurrence information constructed automatically. Experiments show some promising result. The problem is that the results cannot be compared with other works. Although TREC7 and TREC8 queries are used, only 2 sub-collections of TREC disk4 are used for evaluation.

Besides disambiguation, a number of attempts have been done to explore WordNet for text retrieval purpose [GVC98, RS95, V04]. One of the most frequently addressed aspects is to enrich the query with semantic-related terms (synonyms and hyponyms belonging to the same sense with the query term). In the past, improvement in retrieval effectiveness was only achieved by using WordNet manually [GVC98, V04]. Automatic query expansion using WordNet [RS95, V93, V94] on short queries, which are typical queries submitted by users, has not resulted in higher retrieval effectiveness. The main difficulty is that a word usually has multiple synonyms with somewhat different meanings and it is not easy to automatically find the correct synonyms to use.

Compared with the above retrieval systems, we have the following advantages: (1) Our word sense disambiguation approach can be applied to any word beyond the training corpus; (2) Unlike [GVC98, SP95, SOT03, KSR04], we do not disambiguate terms in documents which can be very time consuming as it involves millions of documents. (3) The disambiguated information is also used during feedback. (4) Significant improvement in retrieval effectiveness is achieved via the word sense disambiguation sub-system.

7. CONCLUSION

In this paper, we provided an effective approach to disambiguate word senses in short queries and demonstrated that it can be applied to 100% of ambiguous terms and achieve an accuracy of 90%, which is significantly better than any existing method. Furthermore, our experimental results show that by incorporating our term disambiguation technique into our retrieval model, the retrieval effectiveness can be significantly improved. In fact, on average, the effectiveness of our system is 7% better than the best result reported in the literature.

8. REFERENCES

[BR99] Ricardo Baeza-Yates, Berthier Ribeiro-Neto: *Modern Information Retrieval,* Addison-Wesley, 1999.

[BMSW97] Daniel M. Bikel, Scott Miller, Richard L. Schwartz, Ralph M. Weischedel: Nymble: a High-Performance Learning Name-finder. ANLP 1997: 194-201

[Brill] Brill Tagger: http://www.cs.jhu.edu/~brill/

[CM02] James P. Callan, Teruko Mitamura: Knowledge-based extraction of named entities. CIKM 2002: 532-537

[C98] Nancy Chinchor: “Overview of MUC-7”, MUC-7, (1998)

[GVC98] Julio Gonzalo, Felisa Verdejo, Irina Chugur, Juan M. Cigarran: Indexing with WordNet synsets can improve Text Retrieval CoRR cmp-lg/9808002: (1998)

[JM00] Daniel Jurafsky, James H. Martin: *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition,* Prentice-Hall, 2000

[KSR04] Sang-Bum Kim, Hee-Cheol Seo, Hae-Chang Rim: Information retrieval using word senses: root sense tagging approach. SIGIR 2004: 258-265

[K03] K. Kwok, L. Grunfeld, N. Dinstl, P. Deng, TREC 2003 Robust, HARD, and QA Track Experiments using PIRCS, *TREC12*, 2003.

[K04] K.L. Kwok, L. Grunfeld, H.L. Sun, P. Deng, TREC 2004 Robust Track Experiments Using PIRCS, TREC13, 2004

[LLYM04] Shuang Liu, Fang Liu, Clement Yu, Weiyi Meng: An effective approach to document retrieval via utilizing WordNet and recognizing phrases. SIGIR 2004: 266-272

[LSY04] Shuang Liu, Chaojing Sun, Clement Yu: UIC at TREC 2004: Robust Track. TREC13, 2004

[L86] Michael Lesk: Automatic Sense Disambiguation Using Machine Readable Dictionaries: how to tell a pine cone from an ice cream cone. ACM SIGDOC, 1986.

[MS99] Christopher D. Manning, Hinrich Schütze: *Foundations of Statistical Natural Language Processing*, MIT Press. Cambridge, MA: May 1999.

[MTF04] Rada Mihalcea, Paul Tarau, Elizabeth Figa: PageRank on Semantic Networks, with application to Word Sense Disambiguation, COLING 2004, Switzerland, Geneva, 2004

[M02] Rada Mihalcea: Word Sense Disambiguation Using Pattern Learning and Automatic Feature Selection. Journal of Natural Language and Engineering, 2002.

[M90] George A. Miller. Special Issue. WordNet: An On-line Lexical Database, International Journal of Lexicography, 1990.

[MLRB93] George A. Miller, Claudia Leacock, Randee I. Tengi, R. Bunker: A Semantic Concordance. 3 DARPA Workshop on Human Language Technology, p303-308, 1993.

[PBP03] Siddharth Patwardhan, Satanjeev Banerjee, Ted Pedersen: Using Measures of Semantic Relatedness for Word Sense Disambiguation. CICLing 2003: 241-257

[RS95] R. Richardson, A. Smeaton: Using WordNet in a knowledge-based approach to information retrieval. *BCS-IRSG Colloquium on Information Retrieval*, 1995

[S94] Mark Sanderson: Word Sense Disambiguation and Information Retrieval, ACM SIGIR, 1994

[SP95] Hinrich Schütze, Jan O. Pedersen: Information retrieval based on word senses. In Proceedings of the 4th Annual Symposium on Document Analysis and Information Retrieval, pages 161--175, Las Vegas, NV, 1995

[S98] Hinrich Schütze: Automatic Word Sense Discrimination. Computational Linguistics 24(1): 97-123 (1998)

[SLLYM2005] C. Sun, S. Liu, F. Liu, C. Yu, W. Meng, Recognition and Classification of Noun Phrases in Queries for Effective Retrieval, Technique Report, UIC, 2005,

[SOT03] Christopher Stokoe, Michael P. Oakes, John Tait: Word sense disambiguation in information retrieval revisited. SIGIR 2003: 159-166

[TZM96] Xiang Tong, ChengXiang Zhai, Natasa Milic-Frayling, David A. Evans: Evaluation of Syntactic Phrase Indexing -- CLARIT NLP Track Report. TREC 1996

[V93] Ellen M. Voorhees: Using WordNet to Disambiguate Word Senses for Text Retrieval. SIGIR 1993: 171-180

[V94] Ellen M. Voorhees: Query Expansion Using Lexical-Semantic Relations. SIGIR 1994: 61-69

[V04] Ellen M. Voorhees: Overview of the TREC 2004 Robust Retrieval Track, TREC13, 2004.

[Y95] David Yarowsky: Unsupervised Word Sense Disambiguation Rivaling Supervised Methods. ACL 1995: 189-196

[YCCLT03] D.L. Yeung, C.L.A. Clarke, G.V. Cormack, T.R. Lynam, E.L. Terra, Task-Specific Query Expansion (MultiText Experiments for TREC 2003), *TREC12*, 2003.

[YM98] Clement Yu, Weiyi Meng: Principles of database query processing for advanced applications. San Francisco, Morgan Kaufmann, 1998.