UIC at TREC2005: Robust Track

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

ABSTRACT

This paper presents a new approach to improve retrieval effectiveness by using concepts, examples, and word sense disambiguation. We also employ pseudo-feedback and web-assisted feedback.

1. INTRODUCTION

A concept appears in a query if it is a phrase consisting of adjacent query words or it is a single content word which cannot be combined with other adjacent query words to form a phrase. To process a query, different concepts in a query are identified. A document has a concept if all content words in the concept are within a window of certain size. For a concept identified in a query, its examples are extracted from WordNet [F98, M90, MLRB93] and possibly other dictionaries. A document has an example of a concept if the exact example appears in that document. Concepts are more important than examples and individual terms; and examples are more important than individual terms. Consequently, the similarity measure between a query and a document has three components (concept-sim, example-sim, term-sim), where concept-sim is the similarity by matching the concepts of the query against those in the document; example-sim is the similarity by matching query concepts against the examples of the concepts in the document; and term-sim is the usual similarity between the query and the document based on term matches, with each term in the query contribute to the term similarity computation. The term-sim is computed by the standard Okapi similarity function [RW00]. Documents are ranked in descending order of (concept-sim, example-sim, term-sim). We employ pseudo-feedback into our retrieval. Besides the traditional pseudo-feedback, we also use the web-assisted feedback to get more highly correlated terms.

We utilize word sense disambiguation in our retrieval system. We use a new approach to determine the senses of terms in queries by using WordNet and Web. For each term 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 terms which form a concept, it may be possible to assign senses to these terms. If the above disambiguation fails, then other query terms, if exist, are used, by going through exactly the same process. If the sense of a query term cannot be determined in this manner, then a guess of the sense of the term is made, if the guess has at least 50% chance of being correct. If no sense of the term has 50% or higher chance of being used, then we apply a Web search to assist in the word sense disambiguation process.

Suppose *w* is a term, and (*w w'*) is a concept in a query. Word sense disambiguation is used to improve retrieval performance in two aspects. First, it helps bring in new terms and concepts to the 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 concepts with *w'* or terms brought in by *w'*. Second, it helps choose feedback terms semantically related to disambiguated query terms. Additional weight is assigned to a feedback term if it can be related to a disambiguated query term *w* through some WordNet relations. Furthermore, the feedback term can be used to form new concept with *w'* if it is related to *w* through synonym, hyponym, or coordinate relations. The newly formed concept contributes to concept similarity computation.

2. CONCEPTS AND EXAMPLES

A concept appears in a query if it is a phrase consisting of adjacent query terms or it is a single content word which cannot be combined with other adjacent query words to form a phrase. Terms in a concept are called components of a concept. To process a query, different concepts in a query are identified [SLLYM2005]. Different concepts in a query are not equally important due to semantic overlap. For the same reason, different components within a concept are not equally important too. Adjustments are made to benefit the more important concepts or components if semantic overlap happens.

New concepts are formed during before and during feedback by using newly brought in terms. The newly formed concepts are treated as synonyms of the original concept.

For a concept identified in a query, examples are extracted from WordNet and other sources like Wekipedia. Terms brought in after pseudo-feedback are treated as examples of a concept if they are related to each component of the concept.

We consider concepts to be more important than examples and individual terms; examples are more important than individual terms. Consequently, the similarity of a document with a query will have three components (*concept-sim*, *example-sim, term-sim*), where *concept-sim* is computed based on the concepts in common between the query and the document; *example-sim* is the similarity by matching query concepts against the examples of the concepts in the document; and *term-sim* is the usual term similarity between the document and the query using the *Okapi* formula [RW99]. Each query term which appears in the document contributes to the term similarity value, irrespective of whether it occurs in a concept or not. Consider, for a given query, two documents *d1* and *d2* having similarities (*x1, y1, z1*) and (*x2, y2, z2*), respectively. *d1* will be ranked higher than *d2* if (1) $x1>x2$; (2) $x1=x2$ and $y1>y2$; (3) $x1=x2$ and $y1=y2$ and $z1>z2$. Note that if $xi>0$, then the individual terms which contribute to *concept-sim* will ensure that $zi>0$.

We now describe how concept similarity is computed. For a document having a concept, its *concept-sim* is the *idf* (inverse document frequency weight) of the concept and is independent of the number of times the concept occurs in the document. However, the multiple occurrences of the concept will contribute a higher value to *term-sim*. If a document has multiple distinct concepts, its *concept-sim* is the sum of the corresponding *idf*s. For a document without any concept, its *concept-sim is* 0.

The *example-sim* is computed similarly to *concept-sim* computation.

3. WORD SENSE DISAMBIGUATION

Word sense disambiguation is the process of choosing the right sense for a word in its occurring context [MS99, JM00, Y95]. 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, [KSR04] and our previous work [LLYM04] shows a promising result of applying word sense disambiguation to information retrieval by automatic query expansion.

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 concepts of the query [SLLYM05]. 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 concpet, 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 concepts to the query. Suppose *w* is a term, and (*w w'*) is a concept 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 concepts 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.

3.1 Disambiguation by Cases Analysis

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 concept 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.

We first 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. Detailed cases analysis is available in [LYM05]. For space limit, we omitted here.

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*; (2) the frequency of use of the sense of *w*, which decides the likelihood that the term is used in this sense; and (3) the historical accuracy of the case which determines the sense of *v*. Then, the sense with the largest total likelihood is chosen. A detailed description is in [LYM05].

3.2 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 $>= \frac{1}{2} x$, 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: 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.

3.3 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: 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".

4. QUERY EXPANSION BY USING WORDNET AND NEW CONCEPT FORMATION

Word sense disambiguation helps bring in new terms and concepts to the query. Suppose *c* is a concept and formed by two adjacent terms t_1 and t_2 in query *q*. Whenever the sense of a given term t_1 is determined to be the synset *S*, its synonyms, its various forms, terms in its definition and associated examples, morphologically derived terms, its hyponyms, its compound are all considered for possible addition to the query. Generally terms brought in by t_1 form synonym concepts of *c* with t_2 or terms brought in by t_2. A term brought in by t_1 can be a synonym concept of *c* by itself if we find the semantic of the new brought in term is close or similar to *c*.

4.1 Query Expansion by Using WordNet

New terms are brought in after the sense of the query term is determined. We discuss some of the cases in this section.

4.1.1 Add Synonyms

Whenever the sense of term t_1 is determined, we examine the possibility of adding synonyms of t_1 in its synset *S* to the query.

(a) For any term *t'* except t_1 in *S*, if *t'* is a single term or a phrase not containing t_1, *t'* is added to the query if either (i) *S* is a dominant synset of *t'* or (ii) *S* is not dominant for *t'* but *t'* is highly globally correlated with t_2, and the correlation value between *t'* and t_2 is at lest as large as the value between t_1 and t_2 (which form a concept in the query). The weight of *t'* is given by:

$$W(t') = f(t', S)/F(t') \tag{1}$$

where $f(t', S)$ is the frequency value of *t'* in *S*, and $F(t')$ is the sum of frequency values of *t'* in all synsets which contain *t'* and have the same POS as *t'*. We interpret the weight of *t'* to be the likelihood that *t'* has the same meaning as *t*. The reason synonyms satisfying condition (ii) are added is that the frequency of use of a synset is computed based on a corpus, which may not reflect the characteristics of the term distribution in the collection of documents. The actual relation between t_1 and *t'* may be reflected by the correlation of *t'* and t_2 in the collection.

In WordNet, an adjective synset *S* containing t_1 may have several satellite synsets which have similar but not necessary identical meaning as the original synset *S*. Consider adjective "modern" in query "modern slavery". In this query, sense "belonging to the modern era" of "modern" is the chosen sense. It has the following satellite synsets: {contemporary, modern-day}, {neo}, {recent}, {read-brick, redbrick}, {ultramodern}, {Moderne}. In this example, the meanings of some satellite terms are so close to term "modern" that they can be considered as synonyms of "modern". So for adjective in the query, we check its satellite synsets to get more synonyms.

(b) Suppose t_1 is an adjective in a query, *t''* is in one of its satellite synset *SS*. *t''* is added to the query if both (i) *SS* is a dominant synset of *t''* and (ii) *t''* is an important satellite term. The weight of *t''* is given by:

$$W(t'') = f(t'', SS)/FS(t_1) \tag{2}$$

where $f(t'', SS)$ is the frequency value of *t''* in *SS*, and $FS(t_1)$ is the sum of the frequency values of all satellite terms in all satellite synsets of *S*. We interpret the weight of *t''* to be the importance of a satellite term *t''* among all satellites terms. *t''* is important if $W(t'')$ is greater than the average importance value over all terms in the satellite synsets.

Example: Satellite terms "contemporary", "neo", and "recent" of query term "modern" are added to the query, since they are dominant over their own synsets and are important satellite terms.

A relational adjective is an adjective that relates to or classifies its noun. For example, "musical" is a relational adjective of noun “music” in the sense of “characterized by or capable of producing music”. In WordNet, a relational adjective and its noun are connected through “of or pertaining to” relation --- a relational adjective is of or pertaining to a noun in a related sense (synset). Still the above example, in WordNet “musical” is connected to “music” in the sense of “an artistic form of auditory communication incorporating instrumental or vocal tones in a structured and continuous manner”. In our system, for a query term which is a relational adjective, we add its pertained noun and the synonyms of the noun to the query, if the pertained noun has unique sense.

(c) If t''' is the noun of relational adjective t_1, t''' is a single term or a phrase not containing t_1 in a synset *SN*, t''' is added to the query if *SN* is the unique synset of t'''. Synonyms of t''' in *SN* can also be added to the query if *SN* is dominant for them. The weights of t''' and its synonyms are given by formula (1).

Example: Query term “soviet” is a relational adjective and “Soviet Union” is the pertained noun of “soviet”. “Soviet Union” has unique synset {Soviet Union, Russia, Union of Soviet Socialist Republics, USSR}. This synset is unique for “Union of Soviet Socialist Republics” and “USSR” and dominant for “Russia”. So terms that don’t containing “soviet” including “Russia” and “USSR” are added to the query with weights 0.72 and 1 respectively.

4.1.2 Add Variations of Each Query Term

We add various spelling of a query term including its singular, plural, and different tenses that have different stem forms, and with/without hyphen forms. WordNet has singular and plural for nouns, and different tenses for verb. If a word has a hyphen, we replace the hypen with a space, or delete the hyphen. A variation of the query term has the same weight as that of the query term. Here are some examples:

Example: *Query is “woman in parliament”. The plural of “woman” is “women” which has different stem root with “woman”. It will be added to the query.*

4.1.3 Add Morphologically Derived Words

We add morphologically derivationally related word of query term t_1 to the query. In WordNet, there are derivational morphology links between noun and verb forms: the derivationally related form of a noun is a verb, and vice verse. Suppose the derivationally related form of t_1 is *t* which belongs to a synset *SD*.

(a) If *SD* is dominant for *t*, *t* can be added to the query. The weight of *t* is given by formula (1). If *SD* is the unique sense of a synonym t' (t' != t) in *SD*, t' can be added to the query with the same weight as *t*.

Example: *Query is “police death”. The derivationally related verb of “death” is “die”. It belongs to synset {die, decease, perish, go, exit, pass away, expire, pass} which is dominant for “die”. The weight of “die” is 0.90. This synset is unique for term “decease” and “perish”, so these two terms can be added to the query with the same weight as “die”.*

4.1.4 Add Hyponyms

Suppose *U* is a hyponym synset of t_1. A synonym in *U* is added to the query if one of the following conditions is satisfied:

(a) *U* is the unique hyponym synset of the determined synset of t_1. For each term t' in *U*, t' is added to the query, with a weight similar to that given by the Formula (1), if *U* is dominant in the synsets of t'.

(b) *U* is not a unique hyponym synset of the determined synset of t_1, but the definition of *U* contains term t_2 or its synonyms. For each term t' in *U*, if *U* is dominant in the synsets of t'; t' is added to the query with a weight given by Formula (1). Hyponym synset between the chosen synset of t_1 and *U* are also added to the query. For terms from *U*, they carry the semantic of both t_1 and t_2 (the concept *c*). They can be used the same as concept *c* except their concept weights are given by their term weights.

Example: *Query is “tropical storm”. The definition of “hurricane” which is hyponym of “storm” contains “tropical”, and “hurricane” is the only element in this synset. Thus, “hurricane” is added to the query.*

4.1.5 Add Compounds

Given a term *t*, we can retrieve its compound concepts using WordNet. A compound concept is either a word having term *t* as a sub-string or a dictionary phrase containing term *t*.

Suppose *cp* is a compound concept of a query term t_1 and *cp* has a dominant synset *V*. The compound *cp* can be added to the query if it satisfies one of the following conditions. In each case, its weight is given by formula (1):

(a) The definition of *V* contains t_1 as well as all terms that form a concept with t_1 in the query. *cp* can be treated as synonym concept of *c* as it carry the semantic of *c* and its concept weight is the same as its term weight.

Example: *A term is “nobel”, and a query is “Nobel Prize winner”. Both “nobelist” and “nobel laureate” are compound of "nobel”. Their definition (they are synonyms) is “winner of a Nobel Prize”, which contains all query terms in the concept “Nobel Prize Winner”.*

(b) A compound *cp* consists of t_1 and t_3, if the definition of t_3 contains t_2 --- another query term. *cp* can be treated the same as query concept *c* since it carry the semantic of *c* and its concept weight is the same as its term weight.

In above conditions, if the compound concept has unique sense, its synonyms are also considered for addition if the synset of the compound concept synset is dominant for the synonyms.

4.2 New Concept Formation

Let *c* be a concept formed by t_1 and t_2. Terms brought in by t_1 are used to form new concept with t_2 or terms brought in by t_2. The newly formed concepts are treated as synonym concept of *c*. Suppose t_{1i} and t_{2j} are terms brought in by t_1 and t_2 respectively. Generally the following new concepts are formed: (t_{1i} t_2), (t_{1i} t_{2j}), (t_1 t_{2j}). The weight of a newly formed concept is the product of the weights of the terms that form it.

Example: *Query is "police death", "die" is the derivational related form of "death". It can be used to form concept with "police", which is "police die". The weight of the concept is 1 which is the weight product of "police" and "die".*

As we have seen in some cases in section 4.1.4 and 4.1.5, there is some newly added term that carries the semantic of the query concept *c*. In these situations, the term itself can be treated as synonym of concept *c* no matter which term brings it in, in another words, it is not used to form concept with others. The concept weight of this term is the same as the term weight assigned to it during expansion.

5. QUERY EXPANSION AND CONCEPT FORMATION BY FEEDBACK

5.1 Bring in New Terms By Feedback

We use the same feedback and web-assisted feedback system [K03, K04, YCCLT03] as we described in last year's TREC [LSY04]. A new *t* brought in by pseudo-feedback and/or Web-feedback may be related to some query term t_1 (i) by being a synonym of the **determined sense** of t_1; (ii) by being a hyponym of the **determined sense** of t_1; (iii) by being a coordinate term of **the determined sense** of t_1; (iv) by being a direct hypernym of the **determined sense** of t_1; (v) having a definition which contains t_1 or (vi) the definition of **the determined sense** of t_1 contains *t*. In the cases of (i), *t* is a non-dominant synonym of t_1; otherwise. In all these cases, the weight of *t* is given by how it relates to t_1, using the same computation as discussed in section 2.

The weight described above is given to term *t* based on its relation with t_1. It has another weight based on its correlation with the query in the top retrieved documents in the given collection of documents or Web documents. These two weights should be added together but the sum is bounded by 1.

Feedback terms related to query term t_1 through synonyms, hyponyms, and coordinate terms relations are used to form new concept with t_2. The weight of a newly formed concept is the product of the weights of the terms that form it.

5.1 Add Terms after Feedback

Feedback terms related to query term through synonyms, hyponyms, and coordinate terms relations are considered to be highly related to the query terms. The highly related term are used to bring in new terms after feedback.

Suppose *U* is one of t_1's hyponyms whose definitions contain feedback terms highly correlated to t_2. *U* and other similar hyponyms synsets form a clique *C* through direct hyponym and hypernym relations. *t* is a term in *U*, *t* and its hyponyms can be added to the query if *U* is the dominant sense of *t*. The weight of *t* is the product of *t*'s frequency weight and weight of the feedback terms its definition contains. Since *t* is related to both t_1 and t_2, *t* is considered as examples of concept c. The example weight of *t* is the same as its term weight.

6. ROBUST TRACK

In the robust track [Robust], we submit only 1 run to test our system. This run use title only. WordNet is used to disambiguate word senses and supply synonyms, hyponyms, definition words, and compound concepts. Pseudo-feedback and web-assisted feedback are applied. Table 2 gives the evaluation of this run of the entire 50 topics over the data collection Aquatics.

Table 1. Evaluations for UIC TREC 2005 Robust Track

Geometric MAP	MAP	P(10)	Prediction
0.2326	0.3096	0.5920	1.0479

The geometric mean [V04] of the individual topics' average precision scores is equivalent to taking the log of the individual topics' average precision scores, computing the arithmetic mean of the logs, and exponentiating back for the final geometric mean average precision (MAP) score. MAP is the traditional mean average precision. P(10) give the precision at 10 documents. The prediction value is the area between best and predicted MAP curves. We are No.1 in geometric MAP and Precision at 10 evaluations.

7. CONCLUSION

Our TREC-2005 experiment shows that robust retrieval result can be achieved by: (1) effective use of concept, (2) a new similarity function capturing the use of concepts (3) word sense disambiguation and the utilization of synonyms, hyponyms, definition words and compound concepts which are properly chosen. (4) Web does help retrieval.

8. REFERENCES

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

[F98] C. Fellbaum, WordNet An Electronic Lexical Database. The MIT Press, 1998.

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

[GF98] David Grossman and Ophir Frieder, Ad Hoc Information Retrieval: Algorithms and Heuristics, Kluwer Academic Publishers, 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

[LYM05] Shuang Liu, Clement Yu, and Weiyi Meng. Word Sense Disambiguation in Queries. Proceedings of ACM Conference on Information and Knowledge Management (CIKM2005), Bremen, Germany

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

[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.

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

[RW00] Robertson S E, Walker S. Okapi/Keenbow at TREC-8. TREC-8, 2000

[Robust] Robust Track Guidelines

http://trec.nist.gov/tracks.html

[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

[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

[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*, 2000