An Effective Approach to Document Retrieval via Utilizing WordNet and Recognizing Phrases

Shuang Liu, Fang Liu, Clement Yu
Department of Computer Science
University of Illinois at Chicago
851 South Morgan
Chicago, IL, 60607-7053, U.S.A.
{sliu, fliu1, yu}@cs.uic.edu

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

ABSTRACT
Noun phrases in queries are identified and classified into four types: proper names, dictionary phrases, simple phrases and complex phrases. A document has a phrase if all content words in the phrase are within a window of a certain size. The window sizes for different types of phrases are different and are determined using a decision tree. Phrases are more important than individual terms. Consequently, documents in response to a query are ranked with matching phrases given a higher priority. We utilize WordNet to disambiguate word senses of query terms. Whenever the sense of a query term is determined, its synonyms, hyponyms, words from its definition and its compound words are considered for possible additions to the query. Experimental results show that our approach yields between 23% and 31% improvements over the best-known results on the TREC 9, 10 and 12 collections for short (title only) queries, without using Web data.

Categories and Subject Descriptors
H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval – Retrieval Models, Relevance Feedback, Query Formulation. H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing – Dictionaries, Thesauruses.

General Terms
Algorithms, Performance, Experimentation

Keywords
Information Retrieval, Phrase, WordNet, Word Sense Disambiguation

SIGIR'04, July 25–29, 2004, Sheffield, Yorkshire, UK.

1. INTRODUCTION
Techniques involving phrases, general dictionaries (i.e. WordNet [Mill90, Fellbaum98]) and word sense disambiguation have been investigated with mixed results in the past. In this paper, we propose a new approach to processing noun phrases in documents, adapting sense disambiguation techniques to document retrieval, and possibly adding new terms to a query whenever the query terms are disambiguated. Our approach is sketched as follows.

Noun phrases, if exist in a query, are classified into four types: *proper names* of people or organizations; *dictionary phrases* which can be found in dictionaries such as WordNet; *simple phrases* which do not have any embedded phrases; *complex phrases* which are more complicated phrases.

A document has a phrase if all the content words in the phrase are within a window of a certain size, which depends on the type of the phrase. For a proper name, essentially all content words must be adjacent. Content words in a dictionary phrase need not to be adjacent, so its window size can be larger. The window size for a simple phrase is larger than that for a dictionary phrase but smaller than that for a complex phrase.

A phrase is significant if (1) it is a proper name or a dictionary phrase, or (2) it is a simple phrase or a complex phrase whose content words are highly positively correlated in the document collection. We utilize only significant phrases.

We consider phrases to be more important than individual content words when retrieving documents. Consequently, the similarity measure between a query and a document has two components (*phrase-sim*, *term-sim*), where *phrase-sim* is the similarity obtained by matching the phrases of the query against those 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 contributing to the term similarity computation. The latter similarity can be computed by the standard *Okapi similarity function* [RW99]. Documents are ranked in descending order of (*phrase-sim*, *term-sim*). That is, documents with higher *phrase-sim* will be ranked higher. When documents have the same *phrase-sim*, they will be ranked according to *term-sim*.

WordNet is used for word sense disambiguation of query terms. For adjacent query words, the following information from WordNet is utilized: synonym sets, hyponym sets, and their

definitions. When the sense of a query word is determined, its synonyms, words or phrases from its definition, its hyponyms and its compound words are considered for possible addition to the query.

In traditional pseudo-feedback, new terms highly correlated with the original query in the top ranked documents are added to the query [BR99, GF98]. In our framework, we impose an additional constraint, namely a new term is added only if it is highly positively globally correlated with a query term/phrase. A highly positively globally correlated term can also be added to the query if its WordNet definition contains some query terms.

In Section 2, phrases in a query are identified and classified into four types; the window sizes for different types of phrases are determined. In Section 3, phrase similarity is computed for the original query. In Section 4, WordNet is used to do sense disambiguation. After successful disambiguation, additional words from WordNet are considered for addition to the query. Our pseudo-feedback process is given in section 5. In Section 6, phrase similarity is computed for the expanded query. In Section 7, experimental results are given to demonstrate that our approach yields significant improvements (between 23% and 31%) above the existing best-known results of the TREC 9, 10 and 12 collections when Web data are not used. A comparison with previous works is given in Section 8. Concluding remarks are given in Section 9.

2. PHRASES IDENTIFICATION

Noun phrases in a query are classified into proper names, dictionary phrases, simple phrases and complex phrases. Brill's tagger [Brill] is used to assign a part of speech (POS) to each word in the query. The POS information will be used to recognize simple and complex phrases. To save time, POS tagging is done on queries not on documents. A document has a phrase if the content words in the phrase appear within a window of a certain size (which is the number of words in the window minus the number of words in the phrase). The window sizes of different types of phrases are learned from some training data. Proper names and dictionary phrases are assumed to be significant; a simple phrase or a complex phrase is significant if the terms within the phrase are highly positively correlated. Significant phrases are used during retrieval.

2.1 Determining Phrases Types

Four types of phrases are identified in queries as follows:

Proper names include names of people, places and organizations, and are recognized by the named entity recognizer Minipar [Lin94].

Dictionary phrases are phrases that can be found in dictionaries such as WordNet.

We devise a simple grammar to recognize simple and complex phrases in queries. These two types of phrases are not found in dictionaries.

A simple phrase has no embedded noun phrase, has two to four words, but at most two of them are content words. An example of a simple phrase is "*school uniform*".

A complex phrase either has one or more dictionary phrases and/or simple phrases embedded in it, or there is a noun involved in some complicated way with other words in the phrase. An example of a complex phrase is "*instruments to forecast weather*".

2.2. Determining Window Sizes

A phrase is in a document, if all its content (non-stop) words appear in the document within a certain window size (not including the number of words in the phrase). For each of the four types of phrases mentioned in Section 2.1, the required window size varies. While content words in a proper name are required to be adjacent and in the same order, this condition is not necessarily true for other types of phrases. Accordingly, we impose different proximity conditions for their recognition in documents. Specifically, for a dictionary phrase to be recognized in a document, its content words must be within a certain number of words, say w_1; for a simple phrase, its content words must be within w_2 words, where $w_2 > w_1$; for a complex phrase, they must be within w_3 words, where $w_3 > w_2$. In addition, we require the content words in a simple phrase and a complex phrase to be highly correlated in the documents to be considered as forming the phrase; otherwise, a phrase is not formed and the retrieval will be based on the match of individual content words only. Intuitively, the content words of a dictionary phrase should be close together, say, within 3 words apart. However, this might not be true in practice. For example, for the query "*drugs for mental illness*" which contains the dictionary phrase "*mental illness*", an example relevant document (in a TREC collection) has the two words quite far apart as shown below:

"*… was hospitalized for **mental** problems… and had been on lithium for his **illness** until recently.*"

Based on this observation, suitable distances between content words of different types of phrases should be learned instead of determined arbitrarily based on intuition. Specifically, for a set of training queries, we identify the types of phrases, and the distances of the content words of the phrases in all of the relevant documents and in the irrelevant documents having high similarities with the queries (in the TREC collections). The information is fed into a decision tree (C4.5 [Quin93]). The decision tree produces for each type of phrases a proper distance of d words such that for most relevant documents having the content words of that type of phrases, the content words are within the distance d, and for most irrelevant documents having high similarities with the queries, their content words have distances that exceed d. This information is then applied to a different set of queries for evaluation. The learning results are as follows:

proper name: 0	*dictionary phrase: 16*
simple phrase: 48	*complex phrase: 78*

To avoid over-fitting, we adjust the window sizes so that they are rounded to multiples of 5. After adjustments, the window sizes for proper names, dictionary phrases, simple phrases, and complex phrases are 0, 15, 50 and 80, respectively.

The above window sizes are obtained when TREC9 queries are trained on TREC WT10G data collection. The evaluation result reported in Section 7 is carried out on the query sets TREC10 and TREC12 which are submitted to TREC WT10G and the TREC12 Robust task data collection respectively. When the TREC9 queries are evaluated, the TREC10 queries are used as training data. This produces a set of window sizes, which are similar but not identical to those reported above.

2.3 Determining Significant Phrases

Proper names and dictionary phrases are assumed to be significant. A simple phrase or a complex phrase is significant if the content words within the phrase are highly positively correlated in the collection of documents. The correlation value of the terms in a phrase is given by:

$$correlation(t_1, t_2, ..., t_n) = \frac{P(phrase) - \prod_{t_i \in phrase} P(t_i)}{\prod_{t_i \in phrase} P(t_i)} \quad (1)$$

where t_1 to t_n are the terms in the phrase, P(phrase) is the probability of a document having the phrase and $\prod_{t_i \in phrase} P(t_i)$ is the probability that a document has all terms in the phrase, assuming that these terms are independent. If

$$correlation(t_1, t_2, ..., t_n) > 5 \quad (2)$$

then the phrase is significant. In other words, only when the content words of a potential phrase are much more highly correlated than independent, they form an actual phrase. Additionally, formula (1) is also used to solve parsing ambiguity. For example, the query "t_1 *and* t_2 *and* t_3" can be parsed to either "((t_1 *and* t_2) *and* t_3)" or "(t_1 *and* (t_2 *and* t_3))". The phrases "t_1 *and* t_2"and "t_2 *and* t_3" overlap. If both phrases are significant, the one with a higher correlation value will be chosen.

3. PHRASE SIMILARITY COMPUTATION

We first identify the significant phrases inside a query. Content words not in any phrases are treated as individual terms. In order to have a significant phrase, a document must have all the content words of the phrase within the required window size.

We consider phrases to be more important than individual terms. Consequently, the similarity of a document with a query will have two components (*phrase-sim, term-sim*), where *phrase-sim* is computed based on the significant phrases in common between the query and 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 significant phrase or not. Consider, for a given query, two documents d_1 and d_2 having similarities (x_1, y_1) and (x_2, y_2), respectively. d_1 will be ranked higher than d_2 if either (1) $x_1 > x_2$, or (2) $x_1 = x_2$ and $y_1 > y_2$. Note that if $x_i > 0$, then the individual terms which contribute to *phrase-sim* will ensure that $y_i > 0$. We now describe how phrase similarity is computed.

For a document having a single proper name, a dictionary phrase or a simple significant phrase, its *phrase-sim* is the *idf* (inverse document frequency weight) of the phrase and is independent of the number of times the phrase occurs in the document. However, the multiple occurrences of the phrase will contribute a higher value to *term-sim*. If a document has multiple distinct significant phrases, its *phrase-sim* is the sum of the corresponding *idf*s. For a document having a single significant complex phrase, the *phrase-sim* is the *idf* of the complex phrase plus the *idf*s of the embedded significant phrases. For a document without any significant phrase, its *phrase-sim is* 0.

4. WORD SENSE DISAMBIGUATION AND QUERY EXPANSION USING WORDNET

In WordNet [Mill90, Fellbaum98], synonyms with the same meaning are grouped together to form a synset. Each synset represents one sense. Associated with each synset of a word *t*, there is a definition and a "frequency value" that indicates the extent the word *t* is utilized in this sense. For example, the frequency values of "*baby*" in synsets {*baby, infant*} and {*baby, sister*} are 611 and 27, respectively. Then, the noun "*baby*" is more likely to be used in the sense of "*infant*" than the sense of "*sister*". The synsets in WordNet are linked to each other through different relations including *hyponyms, part of* and *member of.*

Suppose a word with a POS has multiple senses. If one of its synsets *S* has a frequency value higher than the sum of the frequency values of all other synsets of the same word with the same POS, *S* is called the ***dominant synset*** of the word with the given POS.

In Section 4.1, we describe word sense disambiguation using WordNet; in Section 4.2, we consider adding new words to the query.

4.1 Word Sense Disambiguation

Word sense disambiguation makes use of adjacent words in the query. When a given query *q* is parsed, the POS of each word as well as the phrases in *q* are recognized. Suppose two adjacent terms t_1 and t_2 in *q* form a phrase *p*. From WordNet, the following information can be obtained. Each of t_1 and t_2 has (I) one or more synsets; (II) a definition for each synset in (I); (III) one or more hyponym synsets of each synset in (I) (containing IS-A relationships; for example, the synset {*male child, boy*} *is a hypo*nym synset of {*male, male person*}); (IV) definitions of the synsets of the hyponyms in (III). These four items (I), (II), (III) and (IV) can be used to determine the sense of the term t_1 by executing the following steps in the given order. The sense of t_2 is determined in a similar manner.

Step 1. If t_2 or a synonym of t_2 is found in the definition of a synset of t_1, say *S*, *S* is determined to be the sense of t_1.

> **Example 4.1:** *Suppose a query contains the phrase "incandescent light". In WordNet, the definition of a synset of "incandescent" contains the word "light". Thus, this synset of "incandescent" is used.*

Step 2. The definition of each synset of t_1 is compared against the definition of each synset of t_2. The combination of the synset of t_1 and the synset of t_2 that have the maximum positive number of content words in common yields the sense for t_1 and the sense for t_2.

> **Example 4.2:** *Suppose the query is "induction and deduction". Each of the two terms has a number of senses. The definitions of the two terms which have the maximum overlap of two content words, namely "general" and "reasoning" are their determined senses. For "induction" and "deduction", the definitions of the determined synsets are "**reasoning** from detailed facts to **general** principles" and "**reasoning** from the **general** to the particular (or from cause to effect)", respectively.*

Step 3. If t_2 or one of its synonyms appears in the definition of a synset *S* containing a hyponym of t_1, then the sense of t_1 is

determined to be the synset S_1 which contains t_1 and has the descendant *S*.

> **Example 4.3:** *Suppose the query is "tropical storm". A hyponym of the synset {storm, violent storm} is "hurricane" whose definition contains the word "tropical". As a result, the sense of "storm" is determined.*

Step 4. Let t_1 be contained by a synset S_1. S_1 have a hyponym synset *U* that contains a term *h*. If *h* appears in the definition of a synset S_2 containing t_2, then the sense of t_1 is determined to be S_1 and S_2 is the sense of t_2.

Step 5. If all the preceding steps fail, consider the surrounding terms in the query. In other words, we first use those terms that form simple phrases for sense determination. If it fails, then use terms forming complex phrases and finally surrounding query terms to execute the above steps in the same order.

Step 6. If the sense of t_1 has yet to be determined, then decide whether there is a dominant synset *S* for t_1. If there is, the sense of t_1 is assumed to be *S*.

4.2 Query Expansion Using WordNet

Whenever the sense of a given term is determined to be the synset S, its synonyms, words or phrases from its definition, its hyponyms and compound words (see case (4)) of the given term are considered for possible addition to the query as shown in the following four cases, respectively. As in Section 4.1, terms t_1 and t_2 in *q* are adjacent and they form a phrase *p*.

(1) Add Synonyms.

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

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 (a) *S* is a dominant synset of *t'* or (b) *t'* is highly globally correlated with t_2, and the correlation value between *t'* and t_2 is greater than the value between t_1 and t_2. The weight of *t'* is given by

$$W(t') = f(t', S)/F(t') \qquad (3)$$

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*.

> **Example 4.4:** *In Example 4.1, the synset containing "incandescent" also contains "candent". It can be verified that the synset is dominant for "candent" and therefore "candent" is added to the query.*

(2) Add Definition Words.

We select words from the definition of S.

If t_1 is a single sense word, the first shortest noun phrase from the definition can be added to the query if it is highly globally correlated with t_1.

> **Example 4.5:** For *query term "euro" whose definition is "the basic monetary unit of …", the noun phrase "monetary unit" from the definition can be added to the query if it is highly globally correlated with "euro".*

(3) 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 given by the Formula (3), 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 (3).

> **Example 4.6:** *In Example 4.3, the definition of the hyponym synset of "hurricane" contains "tropical", and "hurricane" is the only element in this synset. Thus, "hurricane" is added to the query.*

(4) Add Compound Words.

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

Suppose *c* is a compound word of a query term t_1 and *c* has a dominant synset *V*. The compound word *c* can be added to the query if it satisfies one of the following conditions:

(a) The definition of *V* contains t_1 as well as all terms that form a phrase with t_1 in the query.

> **Example 4.7:** *A term is "nobel", and a query is "Nobel Prize winner". Both "nobelist" and "nobel laureate" are compound words of "nobel". Their definition (they are synonyms) is "winner of a Nobel Prize", which contains all query terms in the phrase "Nobel Prize Winner".*

(b) The definition of *V* contains term t_1, and *c* relates to t_1 through a "*member of*" relation.

5. PSEUDO RELEVANCE FEEDBACK

Pseudo-feedback has been employed with considerable success. However, some extraneous terms are usually brought in. To solve this problem, we propose two methods that are modifications of existing techniques. The first method makes use of global correlation information, WordNet and query context: a new word that is highly globally correlated with a query concept (e.g. any single term, proper name, or dictionary phrase in the query) is added to the query, if it has a unique sense and its definition contains some other query terms. The second method brings in terms from the top ranked documents if they are highly globally correlated with a query concept.

5.1 Using Global Correlations and WordNet

The global correlation between a query concept s and another concept t_i can be measured by the co-occurrences of the two concepts and their *idf* weights as follows.

$$global\_correlation(t_i, s) = idf(s) \times \log(dev(t_i, s)) \qquad (4)$$

where $dev(t_i, s) = \frac{co-occurrence(t_i, s) - df_i \times sdf / N}{df_i \times sdf / N}$, df_i and *sdf* are document frequencies of the term t_i and *s*, *N* is number of

documents in the collection, *idf(s)* is the inverse document frequency weight of *s*, *co-occurrence(t_i, s)* is the number of documents containing t_i and *s and dev(t_i, s)* indicates the extent that t_i and *s* deviate (in the sense of positively correlated) from independence. Since *idf(s)* appears in formula (4), terms that correlate with low frequency query terms are likely to have higher global correlations. If t_i and *s* are independent, then *global_correlation* is negative infinity.

A term or a phrase among the top 10 most highly globally correlated terms with a query concept *s* is added to the query if (1) it has a single sense and (2) its definition contains some other query terms. For example, for the query of "postmenopausal estrogen", "*osteoporosis*" is a top globally related term with "*estrogen*", has a single sense and its definition contains the query term "*postmenopausal*". Therefore, "*osteoporosis*" can be added to the query.

5.2 Combining Local and Global Correlations

A term is brought in if it correlates highly with the query in the top ranked documents and globally with a query concept in the collection. The computation of the correlation of a term with a query in the top-ranked documents is similar to the one described in the Q*uery Expansion Through Local Clustering* [BR99] and will not be repeated here. This yields a set of potential terms. Among these terms, only terms having sufficiently high global correlations (greater than 1 in formula (4)) with at least one query concept will be added.

6. MODIFICATION OF THE QUERY AND THE SIMILARITY FUNCTION

Query expansion may result in a final Boolean query. Consider a query that consists of two terms t_1 and t_2. Suppose t_1 and t_2 bring in new terms t_1' and t_2' respectively. If t_1 and t_2 in the query form a phrase, the expanded query is equivalent to (t_1 AND t_2) or (t_1 AND t_2') or (t_1' AND t_2') or (t_1' AND t_2). In such case, when computing the *phrase-sim* of a document, any occurrence of t_1' and t_2, or t_1 and t_2', or t_1' and t_2' is equivalent to an occurrence of t_1 and t_2. Each query term which appears in the document contributes to the *term-sim* value.

As an example, if a document has both t_1 and t_2', then its *phrase-sim* would be the same as if it had t_1 and t_2. However, its *term-sim* is computed based on the occurrences of both t_1 and t_2'.

7. EXPERIMENTAL RESULTS

7.1 Experiment Setup

Experiments are performed on the WT10G and the TREC disk 4 and 5 (except for the Congressional Record) collections. 100 queries from the TREC9 and the TREC10 ad hoc queries sets and 100 queries from the TREC12 robust queries set are used [Hawk00, HawkCras01, Voor03,]. (Note: TREC11 had neither the ad hoc nor the robust track.) Each query has three portions: title, which is a short query, description, which describes the intention of the user in more detail, and narrative, which gives characteristics of relevant documents of the query. Since the title portion resembles a typical user query, we only use the title in all experiments. The algorithms used in the experiments are denoted as follows:

SO: The Standard *Okapi* formula for passage retrieval [RS76, RW99] is applied, which is $\sum_{T \in Q} w^{(1)} \times \frac{(k_1+1)\times tf}{K+tf} \times \frac{(k_3+1)\times qtf}{k_3+qtf}$, where *avgpl* is the average length of a passage in number of words, $K = k_1 \times ((1-b) + b \times \frac{pl}{avgpl})$ and the other symbols are described in [RW99].

NO: The *Okapi* formula is modified by replacing *pl* and *avgpl* by *Norm* and *AvgNorm* respectively, where *Norm* is the ***L2***-*Norm* of a passage and *AvgNorm* is the average norm of all passages in the collection. We use *Norm* and *AvgNorm* instead of the traditional length factors because passages differ more in norms than in lengths.

NO+P: In addition to use the modified *Okapi* formula, this algorithm computes *phrase-sim* after processing phrases in documents as described in Sections 2 and 3.

NO+P+D: In addition to NO+P, this algorithm employs word sense disambiguation techniques to add terms to the query as described in Section 4.

NO+P+D+F: In addition to NO+P+D, pseudo-feedback techniques as described in Section 5 are employed.

7.2 Results

Table 1 shows the overall performances of the five algorithms executing the three different queries sets on the document collections and the improvements in average precision of each algorithm relative to its preceding algorithm. (The TREC12 queries are partitioned into two subsets TREC12 old and TREC12 new.)

The modified *Okapi* formula (NO) gives an average improvement of between 3.5% and 16% above the baseline (SO), where the improvement varies from one collection to another. The phrase algorithm (NO+P) obtains additional improvements, ranging from 3% to 16% against algorithm (NO). The word sense disambiguation algorithm (NO+P+D) brings in further improvements, ranging from 4% to 34%. Finally, the pseudo-feedback algorithm gains an additional 5% to 9% improvement.

The best-known average precision result in TREC9 was 0.2078 [OMNH00] and the best official result was 0.2011 [Fujita00, Hawk00]. Our result is 0.2613, which is respectively 25.7% and 30% better than the results stated above. The best-known results in TREC10 were 0.2226 as posted at TREC site [Ama01] and 0.2225[AR02]. Our result is 0.2752, which is about 23.6% above these results. In TREC12, systems were allowed to use the description portions of the queries. For those systems that reported results obtained by using titles only, the best result was 0.2692 [Yeung03]. However, it used Web data. Using titles only but without the use of Web data, the best-known result was 0.2052 [ZhaiTao03]. Our result that uses titles only and does not make use of Web data is 0.2685. This is 31% better than the best result (0.2052) under the same condition of performing the experiments. Our result that does not utilize Web data is comparable to the best-known result that utilizes Web data (0.2692). (Note: If both Web data and the descriptions were used, the best-known result is 0.2900[Kwok03].)

Table 1. Mean Average Precision (MAP) Results by Different Algorithms

Algorithm	Query Set (Collection)									
	TREC9 (WT10G)		**TREC10 (WT10G)**		**TREC12 old (disk4&5)**		**TREC12 new (disk4&5)**		**TREC12 overall (disk4&5)**	
SO	0.1872	n/a	0.1834	n/a	0.1009	n/a	0.2779	n/a	0.1894	n/a
NO	0.2096	12%	0.2130	16%	0.1061	5%	0.2860	3%	0.1961	3.5%
NO+P	0.2347	12%	0.2358	11%	0.1228	16%	0.2939	3%	0.2103	7%
NO+P+D	0.2434	4%	0.2574	9%	0.1641	34%	0.3346	14%	0.2494	19%
NO+P+D+F	**0.2613**	7%	**0.2752**	7%	**0.1729**	5%	**0.3641**	9%	**0.2685**	7.6%

8. RELATED WORKS

The use of phrases in document retrieval has been investigated with mixed results [Fag87, CTL91, MBSC97]. Our approach of utilizing phrases is different in the sense that different proximity (window size) requirements are imposed for different types of phrases. In contrast, earlier research usually utilized windows of one size. For simple and complex phrases, we also require that the content words in these phrases to be highly positively correlated in the collection. Our similarity function has two components in which phrases are deemed more important than individual terms.

An early work which applies word sense disambiguation to information retrieval can be found in [Sander94]. The use of WordNet for document retrieval has been attempted by various researchers [Gonzalo98, Krov92, Rich95, Voor93, Voor94]. However, improvement in retrieval effectiveness was only achieved by using WordNet **manually** [Gonzalo98, Voor94]. Automatic query expansion using WordNet [Rich95, Voor93, Voor94] 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. Our word sense disambiguation technique is capable of determining the correct senses for most query terms. Unlike previous works [Gonzalo98, MihaMold00, Stokoe03], we do not disambiguate terms in documents.

Automatic word sense disambiguation has also been considered in [BaPe02, BaPe03, Lesk86, Miha02]. A significant difference between our work and previous works is that our emphasis is on improving retrieval effectiveness and therefore useful terms are added to the query whereas previous works [BaPe02, BaPe03, Lesk86, Miha02] concentrated on word sense disambiguation without the addition of new terms. In [MihaMold00, Voor98, Voor94], all synonyms in the synset containing a disambiguated query term as well as substantial hyponyms of the query term are added to the query. In contrast, we add selective synonyms, hyponyms and compound words as given in Section 4.2. Furthermore, our algorithm for word sense disambiguation is not the same as existing ones. As an example, in a method of disambiguating the senses of two terms t_1 and t_2 in [BaPe02, BaPe03], all definitions of the hyponym synsets of t_1 are unioned and then compared against the union of the definitions of the hyponym synsets of t_2. In contrast, we compare the definition of each hyponym synset of t_1 against the definition of each hyponym synset of t_2. This allows more precise identification of the proper hyponym synset of t_1 to match against the proper hyponym synset of t_2. Our technique is also different from [Miha02]. While we utilize WordNet only, [Miha02] employs training data as well. When use WordNet for disambiguation, we examine more cases. For example, in [Miha02], the definitions of the terms are not utilized for sense disambiguation.

9. CONCLUSIONS

In this paper, we provide an effective approach to process typical short queries and demonstrate that it yields significant improvements over existing algorithms in three TREC collections under the same experimental conditions (no utilization of Web data and using titles only). We plan to make use of Web data to achieve further improvement.

10. ACKNOWLEDGMENTS

11. REFERENCES

[Ama01] G. Amati, C. Carpineto, G. Romano. FUB at TREC-10 Web Track: A Probabilistic Framework for Topic Relevance Term Weighting. *TREC10*, 2001.

[AR02] G. Amati and C. J. Van Rijsbergen. Probabilistic Models of Information Retrieval Based on Measuring the Divergence from Randomness. *ACM TOIS*, 2002.

[BaPe02] S. Banerjee, and T. Pedersen. An Adapted Lesk Algorithm for Word Sense Disambiguation Using WordNet. *International Conference on Computational Linguistics and Intelligent Text Processing*, 2002.

[BaPe03] S. Banerjee, and T. Pedersen. Extended Gloss Overlaps as a Measure of Semantic Relatedness. *International Joint Conference on Artificial Intelligence*, 2003.

[BR99] R. Baeza-Yates and B. Ribeiro-Neto, *Modern Information Retrieval*, Addison-Wesley, 1999.

[Brill] Eric Brill. Penn Treebank Tagger, Copyright by M.I.T and University of Pennsylvania.

[BS95] C. Buckley and G. Salton. Optimization of Relevance Feedback Weights. *ACM SIGIR*, 1995.

[CTL91] W. Croft, H. Turtle, and D. Lewis. The use of phrases and structured queries in information retrieval. *ACM SIGIR*, 1991.

[Fag87] J. Fagan. *Experiments in Automatic Phrase Indexing for Document Retrieval: A Comparison of Syntactic and Non-Syntactic Methods*. PhD thesis, 1987, Department of CS, Cornell University.

[Fellbaum98] C. Fellbaum, *WordNet An Electronic Lexical Database*. The MIT Press, 1998.

[Fujita00] Sumio FUJITA. Reflections on "Aboutness" Evaluation Experiments at Justsystem, *TREC9*, 2000.

[GF98] D. Grossman and O. Frieder, *Ad Hoc Information Retrieval: Algorithms and Heuristics*, Kluwer Academic Publishers, 1998.

[Gonzalo98] J. Gonzalo, F. Verdejo, I. Chugur and J. Cigarran Indexing with WordNet synsets can improve Text Retrieval, *COLING/ACL '98 Workshop on Usage of WordNet for NLP*, 1998.

[Hawk00] D. Hawking. Overview of the Web Track, *TREC9*, 2000

[HawkCras01] D. Hawking, N. Craswell, Overview of the Web Track, TREC11, 2001.

[Krov92] R. Krovetz and W. Croft. Lexical ambiguity and information retrieval. *ACM TOIS*, 1992.

[Kwok03] K. Kwok, L. Grunfeld, N. Dinstl, P. Deng, TREC 2003 Robust, HARD, and QA Track Experiments using PIRCS, *TREC12*, 2003.

[Lesk86] M. Lesk. Automatic Sense Disambiguation Using Machine Readable Dictionaries: how to tell a pine cone from an ice cream cone. *ACM SIGDOC*, 1986.

[Lin94] D. Lin, 1994. PRINCIPAR---An Efficient, broad-coverage, principle-based parser. *COLING*. 1994.

[MBSC97] M. Mitra, C. Buckley, A. Singhal, C. Cardie. An Analysis of Statistical and Syntactic Phrases, *RIAO*, 1997.

[Miha02] R. Mihalcea, Word Sense Disambiguation Using Pattern Learning and Automatic Feature Selection. *Journal of Natural Language and Engineering*, 2002.

[MihaMold00] R. Mihalcea and D. Moldovan. Semantic indexing using WordNet senses. *ACL Workshop on IR & NLP*, 2000.

[Mill90] G. Miller. Special Issue. WordNet: An On-line Lexical Database, International Journal of Lexicography, 1990.

[OMNH00] Y. Ogawa, H. Mano, M. Narita, S. Honma: Structuring and Expanding Queries in the Probabilistic Model. *TREC9*, 2000.

[Qiu93] Y. Qiu. Concept Based Query Expansion. *ACM SIGIR*, 1993.

[Quin93] J. Ross Quinlan, *C4.5: programs for machine learning*, Morgan Kaufmann, 1993.

[Rich95] R. Richardson and A. Smeaton. Using WordNet in a knowledge-based approach to information retrieval. *BCS-IRSG Colloquium on Information Retrieval*, 1995

[RS76] S. Robertson and K. Sparck Jones. Relevance weighting of search terms. *JASIS*, 1976.

[RW99] S. Robertson, S. Walker Okapi/Keenbow at TREC-8, 1999.

[Sander94] M. Sanderson. Word Sense Disambiguation and Information Retrieval. *ACM SIGIR*, 1994.

[SB90] G. Salton, and C. Buckley. Improving retrieval performance by relevance feedback. *JASIS*, 1990

[Stokoe03] C. Stokoe, M. Oakes, J. Tait, Word sense disambiguation in information retrieval revisited, *ACM SIGIR*, 2003.

[Voor93] E. Voorhees. Using WordNet to Disambiguate Word Sense for Text Retrieval. *ACM SIGIR*, 1993.

[Voor94] E. Voorhees. Query expansion using lexical-semantic relations. *ACM SIGIR*, 1994.

[Voor98] E. M. Voorhees. Using WordNet for text retrieval. *In WordNet, an Electronic Lexical Database*, MIT Press, 1998.

[Voor03] E. Voorhees, Overview of the TREC 2003 Robust Retrieval Track, *TREC12*, 2003.

[XuCro96] J. Xu and W. Croft. Query Expansion Using Local and Global Document Analysis. *ACM SIGIR*, 1996.

[Yeung03] D.Yeung, C. Clarke, G. Cormack, T. Lynam, E. Terra,Task-Specific Query Expansion (MultiText Experiments for TREC 2003), 2003.

[YM98] C. Yu and W. Meng, *Principles of database query processing for advanced applications*. San Francisco, Morgan Kaufmann, 1998.

[ZhaiTao03] C. Zhai, T. Tao, H. Fang, Z. Shang, Improving the Robustness of Language Models--UIUC TREC 2003 Robust and Genomics Experiments, *TREC12*, 2003.