Measuring Semantic Similarity between Words Using Web Search Engines

Danushka Bollegala
The University of Tokyo
Hongo 7-3-1, Tokyo
113-8656, Japan
danushka@mi.ci.i.u-tokyo.ac.jp

Yutaka Matsuo
National Institute of Advanced Industrial Science and Technology
Sotokanda 1-18-13, Tokyo
101-0021, Japan
y.matsuo@aist.go.jp

Mitsuru Ishizuka
The University of Tokyo
Hongo 7-3-1, Tokyo
113-8656, Japan
ishizuka@i.u-tokyo.ac.jp

ABSTRACT

Semantic similarity measures play important roles in information retrieval and Natural Language Processing. Previous work in semantic web-related applications such as community mining, relation extraction, automatic meta data extraction have used various semantic similarity measures. Despite the usefulness of semantic similarity measures in these applications, robustly measuring semantic similarity between two words (or entities) remains a challenging task. We propose a robust semantic similarity measure that uses the information available on the Web to measure similarity between words or entities. The proposed method exploits page counts and text snippets returned by a Web search engine. We define various similarity scores for two given words P and Q, using the page counts for the queries P, Q and $P\ AND\ Q$. Moreover, we propose a novel approach to compute semantic similarity using automatically extracted lexico-syntactic patterns from text snippets. These different similarity scores are integrated using support vector machines, to leverage a robust semantic similarity measure. Experimental results on Miller-Charles benchmark dataset show that the proposed measure outperforms all the existing web-based semantic similarity measures by a wide margin, achieving a correlation coefficient of 0.834. Moreover, the proposed semantic similarity measure significantly improves the accuracy (F-measure of 0.78) in a community mining task, and in an entity disambiguation task, thereby verifying the capability of the proposed measure to capture semantic similarity using web content.

Categories and Subject Descriptors

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

General Terms

Algorithms

Keywords

semantic similarity, Web mining

1. INTRODUCTION

The study of semantic similarity between words has long been an integral part of information retrieval and natural language processing. Semantic similarity between entities changes over time and across domains. For example, *apple* is frequently associated with *computers* on the Web. However, this sense of apple is not listed in most general-purpose thesauri or dictionaries. A user who searches for *apple* on the Web, may be interested in this sense of apple and not apple as a fruit. New words are constantly being created as well as new senses are assigned to existing words. Manually maintaining thesauri to capture these new words and senses is costly if not impossible.

We propose an automatic method to measure semantic similarity between words or entities using Web search engines. Because of the vastly numerous documents and the high growth rate of the Web, it is difficult to analyze each document separately and directly. Web search engines provide an efficient interface to this vast information. Page counts and snippets are two useful information sources provided by most Web search engines. Page count of a query is the number of pages that contain the query words [1]. Page count for the query $P\ AND\ Q$ can be considered as a global measure of co-occurrence of words P and Q. For example, the page count of the query *"apple"* AND *"computer"* in *Google* [2] is $288,000,000$, whereas the same for *"banana"* AND *"computer"* is only $3,590,000$. The more than 80 times more numerous page counts for *"apple"* AND *"computer"* indicate that *apple* is more semantically similar to *computer* than is *banana*.

Despite its simplicity, using page counts alone as a measure of co-occurrence of two words presents several drawbacks. First, page count analyses ignore the position of a word in a page. Therefore, even though two words appear in a page, they might not be related. Secondly, page count of a polysemous word (a word with multiple senses) might contain a combination of all its senses. For an example, page counts for *apple* contains page counts for *apple* as a fruit and *apple* as a company. Moreover, given the scale and noise in the Web, some words might occur arbitrarily, i.e. by random chance, on some pages. For those reasons, page counts alone are unreliable when measuring semantic similarity.

[1]Page count may not necessarily be equal to the word frequency because the queried word might appear many times on one page

[2]http:://www.google.com

Snippets, a brief window of text extracted by a search engine around the query term in a document, provide useful information regarding the local context of the query term. Semantic similarity measures defined over snippets, have been used in query expansion [36], personal name disambiguation [4] and community mining [6]. Processing snippets is also efficient as it obviates the trouble of downloading web pages, which might be time consuming depending on the size of the pages. However, a widely acknowledged drawback of using snippets is that, because of the huge scale of the web and the large number of documents in the result set, only those snippets for the top-ranking results for a query can be processed efficiently. Ranking of search results, hence snippets, is determined by a complex combination of various factors unique to the underlying search engine. Therefore, no guarantee exists that all the information we need to measure semantic similarity between a given pair of words is contained in the top-ranking snippets.

This paper proposes a method that considers both page counts and *lexico-syntactic patterns* extracted from snippets, thereby overcoming the problems described above.

For example, let us consider the following snippet from Google for the query ***Jaguar*** *AND* ***cat***.

"The ***Jaguar*** *is the largest* **cat** *in Western Hemisphere and can subdue larger prey than can the puma"*

Here, the phrase *is the largest* indicates a hypernymic relationship between the Jaguar and the cat. Phrases such as *also known as, is a, part of, is an example of* all indicate various semantic relations. Such indicative phrases have been applied to numerous tasks with good results, such as hyponym extraction [12] and fact extraction [27]. From the previous example, we form the pattern ***X*** *is the largest* ***Y***, where we replace the two words *Jaguar* and *cat* by two wildcards **X** and **Y**.

Our contributions in this paper are two fold:

- We propose an automatically extracted lexico-syntactic patterns-based approach to compute semantic similarity using text snippets obtained from a Web search engine.
- We integrate different web-based similarity measures using WordNet synsets and support vector machines to create a robust semantic similarity measure. The integrated measure outperforms all existing Web-based semantic similarity measures in a benchmark dataset. To the best of our knowledge, this is the first attempt to combine both WordNet synsets and Web content to leverage a robust semantic similarity measure.

The remainder of the paper is organized as follows. In section 2 we discuss previous works related to semantic similarity measures. We then describe the proposed method in section 3. Section 4 compares the proposed method against previous Web-based semantic similarity measures and several baselines on a benchmark data set. In order to evaluate the ability of the proposed method in capturing semantic similarity between real-world entities, we apply it in a community mining task. Finally, we show that the proposed method is useful for disambiguating senses in ambiguous named-entities and conclude this paper.

2. RELATED WORK

Semantic similarity measures are important in many Web-related tasks. In query expansion [5, 25, 40] a user query is modified using synonymous words to improve the relevancy of the search. One method to find appropriate words to include in a query is to compare the previous user queries using semantic similarity measures. If there exist a previous query that is semantically related to the current query, then it can be suggested either to the user or internally used by the search engine to modify the original query.

Semantic similarity measures have been used in Semantic Web related applications such as automatic annotation of Web pages [7], community mining [23, 19], and keyword extraction for inter-entity relation representation [26].

Semantic similarity measures are necessary for various applications in natural language processing such as word-sense disambiguation [32], language modeling [34], synonym extraction [16], and automatic thesauri extraction [8]. Manually compiled taxonomies such as WordNet[3] and large text corpora have been used in previous works on semantic similarity [16, 31, 13, 17]. Regarding the Web as a live corpus has become an active research topic recently. Simple, unsupervised models demonstrably perform better when n-gram counts are obtained from the Web rather than from a large corpus [14, 15]. Resnik and Smith [33] extracted bilingual sentences from the Web to create a parallel corpora for machine translation. Turney [38] defined a point-wise mutual information (PMI-IR) measure using the number of hits returned by a Web search engine to recognize synonyms. Matsuo et. al, [20] used a similar approach to measure the similarity between words and apply their method in a graph-based word clustering algorithm.

Given a taxonomy of concepts, a straightforward method to calculate similarity between two words (concepts) is to find the length of the shortest path connecting the two words in the taxonomy [30]. If a word is polysemous then multiple paths might exist between the two words. In such cases, only the shortest path between any two senses of the words is considered for calculating similarity. A problem that is frequently acknowledged with this approach is that it relies on the notion that all links in the taxonomy represent a uniform distance.

Resnik [31] proposed a similarity measure using information content. He defined the similarity between two concepts C_1 and C_2 in the taxonomy as the maximum of the information content of all concepts C that subsume both C_1 and C_2. Then the similarity between two words is defined as the maximum of the similarity between any concepts that the words belong to. He used WordNet as the taxonomy; information content is calculated using the Brown corpus.

Li et al., [41] combined structural semantic information from a lexical taxonomy and information content from a corpus in a nonlinear model. They proposed a similarity measure that uses shortest path length, depth and local density in a taxonomy. Their experiments reported a Pearson correlation coefficient of 0.8914 on the Miller and Charles [24] benchmark dataset. They did not evaluate their method in terms of similarities among named entities. Lin [17] defined the similarity between two concepts as the information that is in common to both concepts and the information contained in each individual concept.

[3]http://wordnet.princeton.edu/

Recently, some work has been carried out on measuring semantic similarity using Web content. Matsuo et al., [19] proposed the use of Web hits for extracting communities on the Web. They measured the association between two personal names using the overlap (Simpson) coefficient, which is calculated based on the number of Web hits for each individual name and their conjunction (i.e., *AND* query of the two names).

Sahami et al., [36] measured semantic similarity between two queries using snippets returned for those queries by a search engine. For each query, they collect snippets from a search engine and represent each snippet as a TF-IDF-weighted term vector. Each vector is L_2 normalized and the centroid of the set of vectors is computed. Semantic similarity between two queries is then defined as the inner product between the corresponding centroid vectors. They did not compare their similarity measure with taxonomy-based similarity measures.

Chen et al., [6] proposed a double-checking model using text snippets returned by a Web search engine to compute semantic similarity between words. For two words P and Q, they collect snippets for each word from a Web search engine. Then they count the occurrences of word P in the snippets for word Q and the occurrences of word Q in the snippets for word P. These values are combined nonlinearly to compute the similarity between P and Q. This method depends heavily on the search engine's ranking algorithm. Although two words P and Q might be very similar, there is no reason to believe that one can find Q in the snippets for P, or vice versa. This observation is confirmed by the experimental results in their paper which reports zero similarity scores for many pairs of words in the Miller and Charles [24] dataset.

3. METHOD

3.1 Outline

We propose a method which integrates both page counts and snippets to measure semantic similarity between a given pair of words. In section 3.2, we define four similarity scores using page counts. We then describe an automatic lexico-syntactic pattern extraction algorithm in section 3.3. We rank the patterns extracted by our algorithm according to their ability to express semantic similarity. We use two-class support vector machines (SVMs) to find the optimal combination of page counts-based similarity scores and top-ranking patterns. The SVM is trained to classify synonymous word-pairs and non-synonymous word-pairs. We select synonymous word-pairs (positive training examples) from WordNet *synsets*[4]. Non-synonymous word-pairs (negative training examples) are automatically created using a random shuffling technique. We convert the output of SVM into a posterior probability. We define the semantic similarity between two words as the posterior probability that they belong to the synonymous-words (positive) class.

3.2 Page-count-based Similarity Scores

Page counts for the query P *AND* Q, can be considered as an approximation of co-occurrence of two words (or multi-word phrases) P and Q on the Web.

[4]Informally, a synset is a set of synonymous words

However, page counts for the query P *AND* Q alone do not accurately express semantic similarity. For example, Google returns $11,300,000$ as the page count for *"car" AND "automobile"*, whereas the same is $49,000,000$ for *"car" AND "apple"*. Although, *automobile* is more semantically similar to *car* than *apple* is, page counts for query *"car" AND "apple"* are more than four times greater than those for the query *"car" and "automobile"*. One must consider the page counts not just for the query P *AND* Q, but also for the individual words P and Q to assess semantic similarity between P and Q.

We modify four popular co-occurrence measures; Jaccard, Overlap (Simpson), Dice, and PMI (Point-wise mutual information), to compute semantic similarity using page counts.

For the remainder of this paper we use the notation $H(P)$ to denote the page counts for the query P in a search engine. The *WebJaccard* coefficient between words (or multi-word phrases) P and Q, WebJaccard(P,Q), is defined as,

$$\text{WebJaccard}(P,Q) = \begin{cases} 0 & \text{if } H(P\cap Q) \leq c \\ \frac{H(P\cap Q)}{H(P)+H(Q)-H(P\cap Q)} & \text{otherwise.} \end{cases} \quad (1)$$

Therein, $P \cap Q$ denotes the conjunction query P *AND* Q. Given the scale and noise in Web data, it is possible that two words may appear on some pages purely accidentally. In order to reduce the adverse effects attributable to random co-occurrences, we set the WebJaccard coefficient to zero if the page count for the query $P \cap Q$ is less than a threshold c[5].

Similarly, we define *WebOverlap*, WebOverlap(P,Q), as,

$$\text{WebOverlap}(P,Q) = \begin{cases} 0 & \text{if } H(P\cap Q) \leq c \\ \frac{H(P\cap Q)}{\min(H(P),H(Q))} & \text{otherwise.} \end{cases} \quad (2)$$

WebOverlap is a natural modification to the Overlap (Simpson) coefficient.

We define the *WebDice* coefficient as a variant of the Dice coefficient. WebDice(P,Q) is defined as,

$$\text{WebDice}(P,Q) = \begin{cases} 0 & \text{if } H(P\cap Q) \leq c \\ \frac{2H(P\cap Q)}{H(P)+H(Q)} & \text{otherwise.} \end{cases} \quad (3)$$

We define *WebPMI* as a variant form of PMI using page counts as,

$$\text{WebPMI}(P,Q) = \begin{cases} 0 & \text{if } H(P\cap Q) \leq c \\ \log_2\left(\frac{\frac{H(P\cap Q)}{N}}{\frac{H(P)}{N}\frac{H(Q)}{N}}\right) & \text{otherwise.} \end{cases} \quad (4)$$

Here, N is the number of documents indexed by the search engine. Probabilities in Eq. 4 are estimated according to the maximum likelihood principle. To calculate PMI accurately using Eq. 4, we must know N, the number of documents indexed by the search engine. Although estimating the number of documents indexed by a search engine [2] is an interesting task itself, it is beyond the scope of this work. In the present work, we set $N = 10^{10}$ according to the number of indexed pages reported by Google.

[5]we set $c = 5$ in our experiments

3.3 Extracting Lexico-Syntactic Patterns from Snippets

Text snippets are returned by search engines alongside with the search results. They provide valuable information regarding the local context of a word. We extract lexico-syntactic patterns that indicate various aspects of semantic similarity. For example, consider the following text snippet returned by Google for the query *"cricket" AND "sport"*. Here, the phrase *is a* indicates a semantic relationship between **cricket** and **sport**. Many such phrases indicate semantic relationships. For example, *also known as, is a, part of, is an example of* all indicate semantic relations of different types. In the example given above, words indicating the semantic relation between *cricket* and *sport* appear between the query words. Replacing the query words by wildcards X and Y we can form the pattern X *is a* Y from the example given above. However, in some cases the words that indicate the semantic relationship do not fall between the query words. For example, consider the following example.

*"**Cricket** is a **sport** played between two teams, each with eleven players."*

Figure 1: Pattern Extraction Example

*"**Toyota** and **Nissan** are two major Japanese car manufacturers."*

Figure 2: Pattern Extraction Example

Here, the relationship between *Toyota* and *Nissan* is that they are both *car manufacturers*. Identifying the exact set of words that convey the semantic relationship between two entities is a difficult problem which requires deeper semantic analysis. However, such an analysis is not feasible considering the numerous ill-formed sentences we need to process on the Web. In this paper, we propose a shallow pattern extraction method to capture the semantic relationship between words in text snippets.

Our pattern extraction algorithm is illustrated in Figure 3. Given a set S of synonymous word-pairs, *GetSnippets* function returns a list of text snippets for the query *"A"* AND *"B"* for each word-pair A, B in S. For each snippet found, we replace the two words in the query by two wildcards. Let us assume these wildcards to be X and Y. For each snippet d in the set of snippets D returned by *GetSnippets*, function *GetNgrams* extract word n-grams for $n = 2, 3, 4$ and 5. We select n-grams which contain exactly one X and one Y. For example, the snippet in Figure 2 yields patterns *X and Y, X and Y are, X and Y are two*. Finally, function *CountFreq* counts the frequency of each pattern we extracted. The procedure described above yields a set of patterns with their frequencies in text snippets obtained from a search engine. It considers the words that fall between X and Y as well as words that precede X and succeeds Y.

```
Algorithm 3.1: EXTRACTPATTERNS($S$)

comment: Given a set $S$ of word-pairs, extract patterns.
for each word-pair $(A, B) \in S$
  do $D \leftarrow$ GetSnippets("$A$ $B$")
$N \leftarrow null$
for each snippet $d \in D$
  do $N \leftarrow N +$ GetNgrams$(d, A, B)$
$Pats \leftarrow$ CountFreq$(N)$
return $(Pats)$
```

Figure 3: Extract patterns from snippets.

To leverage the pattern extraction process, we select 5000 pairs of synonymous nouns from WordNet synsets. For polysemous nouns we selected the synonyms for the dominant sense. The pattern extraction algorithm described in Figure 3 yields 4,562,471 unique patterns. Of those patterns, 80% occur less than 10 times. It is impossible to train a classifier with such numerous sparse patterns. We must measure the confidence of each pattern as an indicator of synonymy. For that purpose, we employ the following procedure.

First, we run the pattern extraction algorithm described in Figure 3 with a non-synonymous set of word-pairs and count the frequency of the extracted patterns. We then use a test of statistical significance to evaluate the probable applicability of a pattern as an indicator of synonymy. The fundamental idea of this analysis is that, if a pattern appears a statistically significant number of times in snippets for synonymous words than in snippets for non-synonymous words, then it is a reliable indicator of synonymy.

To create a set of non-synonymous word-pairs, we select two nouns from WordNet arbitrarily. If the selected two nouns do not appear in any WordNet synset then we select them as a non-synonymous word-pair. We repeat this procedure until we obtain 5000 pairs of non-synonymous words.

Table 1: Contingency table

	v	other than v	All
Freq. in snippets for synonymous word pairs	p_v	$P - p_v$	P
Freq. in snippets for non-synonymous word pairs	n_v	$N - n_v$	N

For each extracted pattern v, we create a contingency table, as shown in Table 1 using its frequency p_v in snippets for synonymous word pairs and n_v in snippets for non-synonymous word pairs. In Table 1, P denotes the total frequency of all patterns in snippets for synonymous word pairs ($P = \sum_v p_v$) and N is the same in snippets for non-synonymous word pairs ($N = \sum_v n_v$).

Using the information in Table 1, we calculate the χ^2 [18] value for each pattern as,

$$\chi^2 = \frac{(P+N)(p_v(N-n_v) - n_v(P-p_v))^2}{PN(p_v+n_v)(P+N-p_v-n_v)}. \tag{5}$$

We selected the top ranking 200 patterns experimentally as described in section 4.2, according to their χ^2 values. Some selected patterns are shown in Table 2.

Before we proceed to the integration of patterns and page-counts-based similarity scores, it is necessary to introduce some constraints to the development of semantic similarity measures. Evidence from psychological experiments suggest that semantic similarity can be context-dependent and even asymmetric [39, 22]. Human subjects have reportedly assigned different similarity ratings to word-pairs when the two words were presented in the reverse order. However, experimental results investigating the effects of asymmetry reports that the average difference in ratings for a word pair is less than 5 percent [22]. In this work, we assume semantic similarity to be symmetric. This is in line with previous work on semantic similarity described in section 2. Under this assumption, we can interchange the query word markers X and Y in the extracted patterns.

3.4 Integrating Patterns and Page Counts

In section 3.2 we defined four similarity scores using page counts. Section 3.3 described a lexico-syntactic pattern extraction algorithm and ranked the patterns according to their ability to express synonymy. In this section we describe leverage of a robust semantic similarity measure through integration of all the similarity scores and patterns described in previous sections.

```
Algorithm 3.2: GetFeatureVector($A, B$)

  comment: Given a word-pair $A, B$ get its feature vector $F$.
  $D \leftarrow$ GetSnippets("$A$ $B$")
  $N \leftarrow null$
  for each snippet$d \in D$
    do $N \leftarrow N +$ GetNgrams$(d, A, B)$
  $SelPats \leftarrow$ SelectPatterns$(N, GoodPats)$
  $PF \leftarrow$ Normalize$(SelPats)$
  $F \leftarrow [PF, WebJaccard, WebOverlap, WebDice, WebPMI]$
  return $(F)$
```

Figure 4: Create a feature vector F for a word-pair (A, B).

For each pair of words (A, B), we create a feature vector F as shown in Figure 4. First, we query Google for *"A" AND "B"* and collect snippets. Then we replace the query words A and B with two wildcards X and Y, respectively in each snippet. Function *GetNgrams* extracts n-grams for $n = 2, 3, 4$ and 5 from the snippets. We select n-grams having exactly one X and one Y as we did in the pattern extraction algorithm in Figure 3. Let us assume the set of patterns selected based on their χ^2 values in section 3.3 to be *GoodPats*. Then, the function *SelectPatterns* selects the n-grams from N which appear in *GoodPats*. In *Normalize*(*SelPats*), we normalize the count of each pattern by diving it from the total number of counts of the observed patterns. This function returns a vector of patterns where each element is the normalized frequency of the corresponding pattern in the snippets for the query "A" "B". We append similarity scores calculated using page counts in section 3.2 to create the final feature vector F for the word-pair (A, B). This procedure yields a 204 dimensional (4 page-counts based similarity scores and 200 lexico-syntactic patterns) feature vector F. We form such feature vectors for all synonymous word-pairs (positive training examples) as well as for non-synonymous word-pairs (negative training examples). We then train a two-class support vector machine with the labelled feature vectors.

Once we have trained an SVM using synonymous and non-synonymous word pairs, we can use it to compute the semantic similarity between two given words. Following the same method we used to generate feature vectors for training, we create a feature vector F' for the given pair of words (A', B'), between which we need to measure the semantic similarity. We define the semantic similarity SemSim(A', B') between A' and B' as the posterior probability Prob$(F'|synonymous)$ that feature vector F' belongs to the synonymous-words (positive) class.

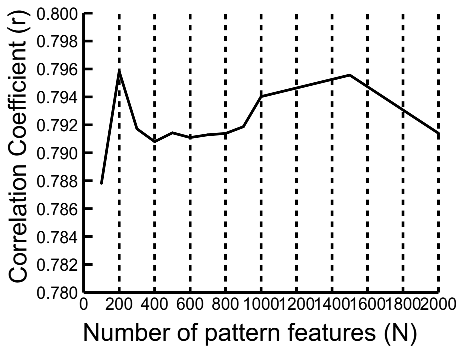

Figure 5: Correlation vs. No of pattern features

$$\text{SemSim}(A', B') = \text{Prob}(F'|synonymous) \quad (6)$$

Being a large-margin classifier, the output of an SVM is the distance from the decision hyper-plane. However, this is not a calibrated posterior probability. We use sigmoid functions to convert this uncalibrated distance into a calibrated posterior probability (see [29] for a detailed discussion on this topic).

4. EXPERIMENTS

We conduct two sets of experiments to evaluate the proposed semantic similarity measure. First we compare the similarity scores produced by the proposed measure against Miller-Charles benchmark dataset. We analyze the behavior of the proposed measure with the number of patterns used as features, the number of snippets used to extract the patterns, and the size of the training dataset. Secondly, we apply the proposed measure in two real-world applications: community mining and entity disambiguation.

4.1 The Benchmark Dataset

We evaluate the proposed method against Miller-Charles [24] dataset, a dataset of 30 word-pairs[6] rated by a group of 38 human subjects. The word pairs are rated on a scale from 0 (no similarity) to 4 (perfect synonymy). Miller-Charles' data set is a subset of Rubenstein-Goodenough's [35] original data set of 65 word pairs. Although Miller-Charles experiment was carried out 25 years later than Rubenstein-Goodenough's, two sets of ratings are highly correlated (pearson correlation coefficient=0.97). Therefore, Miller-Charles ratings can be considered as a reliable benchmark for evaluating semantic similarity measures.

4.2 Pattern Selection

We trained a linear kernel SVM with top N pattern features (ranked according to their χ^2 values) and calculated the correlation coefficient against the Miller-Charles benchmark dataset. Results of the experiment are shown in Figure 5. In Figure 5 a steep improvement of correlation with the number of top-ranking patterns is appearent; it reaches

[6]Because of the omission of two word pairs in earlier versions of WordNet, most researchers had used only 28 pairs for evaluations

Table 2: Features with the highest SVM linear kernel weights

feature	χ^2	SVM weight
WebDice	N/A	8.19
X/Y	33459	7.53
X, Y :	4089	6.00
X or Y	3574	5.83
X Y for	1089	4.49
X . the Y	1784	2.99
with X (Y	1819	2.85
X=Y	2215	2.74
X and Y are	1343	2.67
X of Y	2472	2.56

Table 3: Performance with different Kernels

Kernel Type	Correlation
Linear	0.8345
Polynomial degree=2	0.5872
Polynomial degree=3	0.5387
RBF	0.6632
Sigmoid	0.5277

a maximum at 200 features. With more than 200 patterns correlation drops below this maximum. Considering that the patterns are ranked according to their ability to express semantic similarity and the majority of patterns are sparse, we selected only the top ranking 200 patterns for the remaining experiments.

Features with the highest linear kernel weights are shown in Table 2 alongside their χ^2 values. The weight of a feature in the linear kernel can be considered as a rough estimate of the influence it imparts on the final SVM output. WebDice has the highest kernel weight followed by a series of pattern-based features. WebOverlap (rank=18, weight=2.45), WebJaccard (rank=66, weight=0.618) and WebPMI (rank=138, weight=0.0001) are not shown in Table 2 because of space limitations. It is noteworthy that the pattern features in Table 2 agree with intuition. Lexical patterns (e.g., *X or Y, X and Y are, X of Y*) as well as syntax patterns (e.g., bracketing, comma usage) are extracted by our method.

We experimented with different kernel types as shown in Table 3. Best performance is achieved with the linear kernel. When higher degree kernels such as quadratic (Polynomial degree=2) and cubic (Polynomial degree=3) are used, correlation with the human ratings decreases rapidly. Second best is the Radial Basis Functions (RBFs), which reports a correlation coefficient of 0.6632. For the rest of the experiments in this paper we use the linear kernel.

4.3 Semantic Similarity

We score the word pairs in Miller-Charles' dataset using the page-count-based similarity scores defined in section 3.2, Web-based semantic similarity measures proposed in previous work (Sahami [36], Chen [6]) and the proposed method (SemSim). Results are shown in Table 4. All figures, except those for the Miller-Charles ratings, are normalized into values in $[0, 1]$ range for ease of comparison. Pearson's correlation coefficient is invariant against a linear transformation. Proposed method earns the highest correlation of 0.834 in our experiments. It shows the highest similarity score for the word-pair *magician* and *wizard*. Lowest similarity is reported for *cord* and *smile*[7]. Our reimplementation of Co-occurrence Double Checking (CODC) measure [6] indicates the second-best correlation of 0.6936. The CODC measure is defined as,

$$\text{CODC}(P, Q) = \begin{cases} 0 & \text{if } f(P@Q) = 0 \\ e^{\log\left[\frac{f(P@Q}{H(P)} \times \frac{f(Q@P)}{H(Q)}\right]^{\alpha}} & \text{otherwise.} \end{cases} \quad (7)$$

Therein, $f(P@Q)$ denotes the number of occurrences of P in the top-ranking snippets for the query Q in Google. $H(P)$ is the page count for query P. α is a constant in CODC model and it is set to 0.15 according to Chen et al., [6]. CODC measure reports zero similarity scores for many word-pairs in the benchmark. One reason for this sparsity in CODC measure is that even though two words in a pair (P, Q) are semantically similar, we might not always find Q among the top snippets for P (and vice versa). As might be appearent from the definition of the CODC measure in Eq. 7, it returns zero under these conditions. Ranking of snippets, (hence the value of $f(P@Q)$), depends directly upon the search engine's specifications. A search engine considers various factors such as novelty, authority, link structure, user preferences when ranking search results. Consequently, CODC measure is influenced by these factors.

Similarity measure proposed by Sahami et al. [36] is placed third, reflecting a correlation of 0.5797. This method use only those snippets when calculating semantic similarity. Among the four page-counts-based measures, WebPMI garners the highest correlation ($r = 0.5489$). Overall, the results in Table 4 suggest that similarity measures based on snippets are more accurate than the ones based on page counts in capturing semantic similarity.

4.4 Taxonomy-Based Methods

Table 5: Comparison with taxonomy-based methods

Method	Correlation
Human replication	0.9015
Resnik (1995)	0.7450
Lin (1998)	0.8224
Li et al. (2003)	0.8914
Edge-counting	0.664
Information content	0.745
Jiang & Conrath (1998)	0.8484
Proposed	0.8129

Table 5 presents a comparison of the proposed method to the WordNet-based methods. The proposed method outperforms simple WordNet-based approaches such as Edge-counting and Information Content measures. It is comparable with Lin (1998) [17] Jiang & Conrath (1998) [13] and Li (2003) [41] methods. Unlike the WordNet based methods, proposed method requires no a hierarchical taxonomy of concepts or sense-tagged definitions of words.

[7]We did not use any of the words in the benchmark dataset or their synsets for training

Table 4: Semantic Similarity of Human Ratings and Baselines on Miller-Charles' dataset

Word Pair	Miller-Charles'	Web Jaccard	Web Dice	Web Overlap	Web PMI	Sahami [36]	CODC [6]	Proposed SemSim
cord-smile	0.13	0.102	0.108	0.036	0.207	0.090	0	0
rooster-voyage	0.08	0.011	0.012	0.021	0.228	0.197	0	0.017
noon-string	0.08	0.126	0.133	0.060	0.101	0.082	0	0.018
glass-magician	0.11	0.117	0.124	0.408	0.598	0.143	0	0.180
monk-slave	0.55	0.181	0.191	0.067	0.610	0.095	0	0.375
coast-forest	0.42	0.862	0.870	0.310	0.417	0.248	0	0.405
monk-oracle	1.1	0.016	0.017	0.023	0	0.045	0	0.328
lad-wizard	0.42	0.072	0.077	0.070	0.426	0.149	0	0.220
forest-graveyard	0.84	0.068	0.072	0.246	0.494	0	0	0.547
food-rooster	0.89	0.012	0.013	0.425	0.207	0.075	0	0.060
coast-hill	0.87	0.963	0.965	0.279	0.350	0.293	0	0.874
car-journey	1.16	0.444	0.460	0.378	0.204	0.189	0.290	0.286
crane-implement	1.68	0.071	0.076	0.119	0.193	0.152	0	0.133
brother-lad	1.66	0.189	0.199	0.369	0.644	0.236	0.379	0.344
bird-crane	2.97	0.235	0.247	0.226	0.515	0.223	0	0.879
bird-cock	3.05	0.153	0.162	0.162	0.428	0.058	0.502	0.593
food-fruit	3.08	0.753	0.765	1	0.448	0.181	0.338	0.998
brother-monk	2.82	0.261	0.274	0.340	0.622	0.267	0.547	0.377
asylum-madhouse	3.61	0.024	0.025	0.102	0.813	0.212	0	0.773
furnace-stove	3.11	0.401	0.417	0.118	1	0.310	0.928	0.889
magician-wizard	3.5	0.295	0.309	0.383	0.863	0.233	0.671	1
journey-voyage	3.84	0.415	0.431	0.182	0.467	0.524	0.417	0.996
coast-shore	3.7	0.786	0.796	0.521	0.561	0.381	0.518	0.945
implement-tool	2.95	1	1	0.517	0.296	0.419	0.419	0.684
boy-lad	3.76	0.186	0.196	0.601	0.631	0.471	0	0.974
automobile-car	3.92	0.654	0.668	0.834	0.427	1	0.686	0.980
midday-noon	3.42	0.106	0.112	0.135	0.586	0.289	0.856	0.819
gem-jewel	3.84	0.295	0.309	0.094	0.687	0.211	1	0.686
Correlation	1	0.259	0.267	0.382	0.548	0.579	0.693	0.834

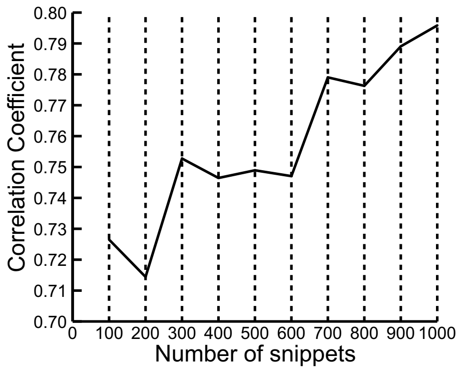

Figure 6: Correlation vs. No of snippets

Therefore, in principle the proposed method could be used to calculate semantic similarity between named entities, etc, which are not listed in WordNet or other manually compiled thesauri. However, considering the high correlation between human subjects (0.9), there is still room for improvement.

4.5 Accuracy vs. Number of Snippets

We computed the correlation with the Miller-Charles ratings for different numbers of snippets to investigate the effect of the number of snippets used to extract patterns upon the semantic similarity measure. The experimental results are presented in Figure 6. From Figure 6 it is appeerant that overall the correlation coefficient improves with the number of snippets used for extracting patterns. The probability of finding better patterns increases with the number of processed snippets. That fact enables us to represent each pair of words with a rich feature vector, resulting in better performance.

4.6 Training Data

We used synonymous word pairs extracted from WordNet synsets as positive training examples and automatically generated non-synonymous word pairs as negative training examples to train a two-class support vector machine in section 3.4. To determine the optimum combination of positive and negative training examples, we trained a linear kernel SVM with different combinations of positive and negative training examples and evaluated accuracy against the human ratings in the Miller-Charles benchmark dataset. Experimental results are summarized in Figure 7. Maximum correlation coefficient of 0.8345 is achieved with 1900 positive training examples and 2400 negative training examples. Moreover, Figure 7 reveals that correlation does not improve beyond 2500 positive and negative training examples. Therefore, we can conclude that 2500 examples are sufficient to leverage the proposed semantic similarity measure.

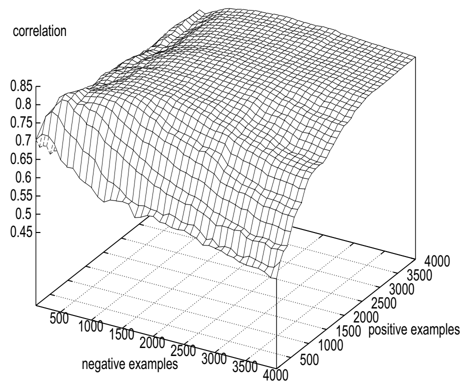

Figure 7: Correlation vs. No of positive and negative training instances

4.7 Community Mining

Measuring semantic similarity between named entities is vital in many applications such as query expansion [36], entity disambiguation (e.g. namesake disambiguation) and community mining [19]. Because most named entities are not covered by WordNet, similarity measures that are based on WordNet cannot be used directly in these tasks. Unlike common English words, named entities are being created constantly. Manually maintaining an up-to-date taxonomy of named entities is costly, if not impossible. The proposed semantic similarity measure is appealing for these applications because it does not require pre-compiled taxonomies.

In order to evaluate the performance of the proposed measure in capturing the semantic similarity between named-entities, we set up a community mining task. We select 50 personal names from 5 communities: *tennis players*, *golfers*, *actors*, *politicians* and *scientists* , (10 names from each community) from the open directory project (DMOZ)[8]. For each pair of names in our data set, we measure their similarity using the proposed method and baselines. We use group-average agglomerative hierarchical clustering (GAAC) to cluster the names in our dataset into five clusters.

Initially, each name is assigned to a separate cluster. In subsequent iterations, group average agglomerative clustering process, merges the two clusters with highest correlation. Correlation, Corr(Γ) between two clusters A and B is defined as the following,

$$\text{Corr}(\Gamma) = \frac{1}{2}\frac{1}{|\Gamma|(|\Gamma|-1)}\sum_{(u,v)\in\Gamma}\text{sim}(u,v) \quad (8)$$

Here, Γ is the merger of the two clusters A and B. $|\Gamma|$ denotes the number of elements (persons) in Γ and sim(u,v) is the semantic similarity between two persons u and v in Γ. We terminate GAAC process when exactly five clusters are formed. We adopt this clustering method with different semantic similarity measures sim(u,v) to compare their accuracy in clustering people who belong to the same community.

We employed the *B-CUBED* metric [1] to evaluate the clustering results. The B-CUBED evaluation metric was originally proposed for evaluating cross-document co-reference chains. We compute precision, recall and F-score for each name in the data set and average the results over the dataset. For each person p in our data set, let us denote the cluster that p belongs to by $C(p)$. Moreover, we use $A(p)$ to denote the affiliation of person p, e.g., $A($"Tiger Woods"$) =$"Tennis Player". Then we calculate precision and recall for person p as,

[8]http://dmoz.org

Table 6: Results for Community Mining

Method	Precision	Recall	F Measure
WebJaccard	0.5926	0.712	0.6147
WebOverlap	0.5976	0.68	0.5965
WebDice	0.5895	0.716	0.6179
WebPMI	0.2649	0.428	0.2916
Sahami [36]	0.6384	0.668	0.6426
Chen [6]	0.4763	0.624	0.4984
Proposed	0.7958	0.804	0.7897

$$\text{Precision}(p) = \frac{\text{No. of people in C(p) with affiliation A(p)}}{\text{No. of people in C(p)}}, \quad (9)$$

$$\text{Recall}(p) = \frac{\text{No. of people in C(p) with affiliation A(p)}}{\text{Total No. of people with affiliation A(p)}}. \quad (10)$$

Since, we selected 10 people from each of the five categories, the denominator in Formula 10 is 10 for all the names p.

Then, the F-score of person p is defined as,

$$\text{F}(p) = \frac{2 \times \text{Precision}(p) \times \text{Recall}(p)}{\text{Precision}(p) + \text{Recall}(p)}. \quad (11)$$

Overall precision, recall and F-score are computed by taking the averaged sum over all the names in the dataset.

$$\text{Precision} = \frac{1}{N}\sum_{p\in DataSet}\text{Precision}(p) \quad (12)$$

$$\text{Recall} = \frac{1}{N}\sum_{p\in DataSet}\text{Recall}(p) \quad (13)$$

$$F\text{-Score} = \frac{1}{N}\sum_{p\in DataSet}\text{F}(p) \quad (14)$$

Here, *DataSet* is the set of 50 names selected from the open directory project. Therefore, $N = 50$ in our evaluations.

Experimental results are shown in Table 6. The proposed method shows the highest entity clustering accuracy in Table 6 with a statistically significant ($p \leq 0.01$ Tukey HSD) F score of 0.7897. Sahami et al. [36]'s snippet-based similarity measure, WebJaccard, WebDice and WebOverlap measures yield similar clustering accuracies.

4.8 Entity Disambiguation

Disambiguating named entities is important in various applications such as information retrieval [9], social network extraction [19, 3, 4], Word Sense Disambiguation (WSD) [21], citation matching [11] and cross-document co-reference resolution [28, 10].

For example, *Jaguar* is a cat, a car brand and also an operating system for computers. A user who searches for *Jaguar* on the Web, may be interested in either one of these different senses of Jaguar. However, only the first sense (Jaguar as a cat) is listed in WordNet. Considering the number of new senses constantly being associated to the existing words on the Web, it is costly, if not impossible to maintain sense tagged dictionaries to cover all senses.

Contextual Hypothesis for Sense [37] states that the context in which a word appears can be used to determine its sense. For example, a Web page discussing Jaguar as a car, is likely to talk about other types of cars, parts of cars etc. Whereas, a Web page on Jaguar the cat, is likely to contain information about other types of cats and animals. In this section, we utilize the clustering algorithm described in section 4.7 to cluster the top 1000 snippets returned by Google for two ambiguous entities *Jaguar* and *Java*. We represent each snippet as a bag-of-words and calculate the similarity $\text{SIM}(S_a, S_b)$ between two snippets S_a,$S_b$as follows,

$$\text{SIM}(S_a, S_b) = \frac{1}{|S_a||S_b|} \sum_{a \in S_a, b \in S_b} sim(a, b) \qquad (15)$$

In Formula 15 $|S|$ denotes the number of words in snippet S. We used different semantic similarity measures for *sim* in Formula 15 and employed the group average agglomerative clustering explained in section 4.7. We manually analyzed the snippets for queries *Java* (3 senses: programming language, Island, coffee) and *Jaguar* (3 senses: cat, car, operating system) and computed precision, recall and F-score for the clusters created by the algorithm.

Our experimental results are summarized in Table 7. Proposed method reports the best results among all the baselines compared in Table 7. However, the experiment needs to be carried out on a much larger data set of ambiguous entities in order to obtain any statistical guarantees.

4.9 Conclusion

In this paper, we proposed a measure that uses both page counts and snippets to robustly calculate semantic similarity between two given words or named entities. The method consists of four page-count-based similarity scores and automatically extracted lexico-syntactic patterns. We integrated page-counts-based similarity scores with lexico syntactic patterns using support vector machines. Training data were automatically generated using WordNet synsets. Proposed method outperformed all the baselines including previously proposed Web-based semantic similarity measures on a benchmark dataset. A high correlation (correlation coefficient of 0.834) with human ratings was found for semantic similarity on this benchmark dataset. Only 1900 positive examples and 2400 negative examples are necessary to leverage the proposed method, which is efficient and scalable because it only processes the snippets (no downloading of Web pages is necessary) for the top ranking results by Google. A contrasting feature of our method compared to the WordNet based semantic similarity measures is that our method requires no taxonomies, such as WordNet, for calculation of similarity. Therefore, the proposed method can be applied in many tasks where such taxonomies do not exist or are not up-to-date. We employed the proposed method in community mining and entity disambiguation experiments. Results of our experiments indicate that the proposed method can robustly capture semantic similarity between named entities. In future research, we intend to apply the proposed semantic similarity measure in automatic synonym extraction, query suggestion and name alias recognition.

5. REFERENCES

[1] A. Bagga and B. Baldwin. Entity-based cross document coreferencing using the vector space model. In *Proc. of 36th COLING-ACL*, pages 79–85, 1998.

[2] Z. Bar-Yossef and M. Gurevich. Random sampling from a search engine's index. In *Proceedings of 15th International World Wide Web Conference*, 2006.

[3] R. Bekkerman and A. McCallum. Disambiguating web appearances of people in a social network. In *Proceedings of the World Wide Web Conference (WWW)*, pages 463–470, 2005.

[4] D. Bollegala, Y. Matsuo, and M. Ishizuka. Disambiguating personal names on the web using automatically extracted key phrases. In *Proc. of the 17th European Conference on Artificial Intelligence*, pages 553–557, 2006.

[5] C. Buckley, G. Salton, J. Allan, and A. Singhal. Automatic query expansion using smart: Trec 3. In *Proc. of 3rd Text REtreival Conference*, pages 69–80, 1994.

[6] H. Chen, M. Lin, and Y. Wei. Novel association measures using web search with double checking. In *Proc. of the COLING/ACL 2006*, pages 1009–1016, 2006.

[7] P. Cimano, S. Handschuh, and S. Staab. Towards the self-annotating web. In *Proc. of 13th WWW*, 2004.

[8] J. Curran. Ensemble menthods for automatic thesaurus extraction. In *Proc. of EMNLP*, 2002.

[9] D. R. Cutting, J. O. Pedersen, D. Karger, and J. W. Tukey. Scatter/gather: A cluster-based approach to browsing large document collections. In *Proceedings SIGIR '92*, pages 318–329, 1992.

[10] M. Fleischman and E. Hovy. Multi-document person name resolution. In *Proceedings of 42nd Annual Meeting of the Association for Computational Linguistics (ACL), Reference Resolution Workshop*, 2004.

[11] H. Han, H. Zha, and C. L. Giles. Name disambiguation in author citations using a k-way spectral clustering method. In *Proceedings of the International Conference on Digital Libraries*, 2005.

[12] M. Hearst. Automatic acquisition of hyponyms from large text corpora. In *Proc. of 14th COLING*, pages 539–545, 1992.

[13] J. Jiang and D. Conrath. Semantic similarity based on corpus statistics and lexical taxonomy. In *Proc. of the International Conference on Research in Computational Linguistics ROCLING X*, 1998.

[14] F. Keller and M. Lapata. Using the web to obtain frequencies for unseen bigrams. *Computational Linguistics*, 29(3):459–484, 2003.

[15] M. Lapata and F. Keller. Web-based models ofr natural language processing. *ACM Transactions on Speech and Language Processing*, 2(1):1–31, 2005.

[16] D. Lin. Automatic retreival and clustering of similar words. In *Proc. of the 17th COLING*, pages 768–774, 1998.

Table 7: Entity Disambiguation Results

	Jaguar			Java		
Method	Precision	Recall	F	Precision	Recall	F
WebJaccard	0.5613	0.541	0.5288	0.5738	0.5564	0.5243
WebOverlap	0.6463	0.6314	0.6201	0.6228	0.5895	0.56
WebDice	0.5613	0.541	0.5288	0.5738	0.5564	0.5243
WebPMI	0.5607	0.478	0.5026	0.7747	0.595	0.6468
Sahami [36]	0.6061	0.6337	0.6019	0.751	0.4793	0.5761
CODC [6]	0.5312	0.6159	0.5452	0.7744	0.5895	0.6358
Proposed	0.6892	0.7144	0.672	0.8198	0.6446	0.691

[17] D. Lin. An information-theoretic definition of similarity. In *Proc. of the 15th ICML*, pages 296–304, 1998.

[18] C. D. Manning and H. Schütze. *Foundations of Statistical Natural Language Processing*. The MIT Press, Cambridge, Massachusetts, 2002.

[19] Y. Matsuo, J. Mori, M. Hamasaki, K. Ishida, T. Nishimura, H. Takeda, K. Hasida, and M. Ishizuka. Polyphonet: An advanced social network extraction system. In *Proc. of 15th International World Wide Web Conference*, 2006.

[20] Y. Matsuo, T. Sakaki, K. Uchiyama, and M. Ishizuka. Graph-based word clustering using web search engine. In *Proc. of EMNLP 2006*, 2006.

[21] D. McCarthy, R. Koeling, J. Weeds, and J. Carroll. Finding predominant word senses in untagged text. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL'04)*, pages 279–286, 2004.

[22] D. Medin, R. Goldstone, and D. Gentner. Respects for similarity. *Psychological Review*, 6(1):1–28, 1991.

[23] P. Mika. Ontologies are us: A unified model of social networks and semantics. In *Proc. of ISWC2005*, 2005.

[24] G. Miller and W. Charles. Contextual correlates of semantic similarity. *Language and Cognitive Processes*, 6(1):1–28, 1998.

[25] M. Mitra, A. Singhal, and C. Buckley. Improving automatic query expansion. In *Proc. of 21st Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval*, pages 206–214, 1998.

[26] J. Mori, Y. Matsuo, and M. Ishizuka. Extracting keyphrases to represent relations in social networks from web. In *Proc. of 20th IJCAI*, 2007.

[27] M. Pasca, D. Lin, J. Bigham, A. Lifchits, and A. Jain. Organizing and searching the world wide web of facts - step one: the one-million fact extraction challenge. In *Proc. of AAAI-2006*, 2006.

[28] X.-H. Phan, L.-M. Nguyen, and S. Horiguchi. Personal name resolution crossover documents by a semantics-based approach. *IEICE Transactions on Information and Systems*, E89-D:825–836, 2005.

[29] J. Platt. Probabilistic outputs for support vector machines and comparison to regularized likelihood methods. *Advances in Large Margin Classifiers*, pages 61–74, 2000.

[30] R. Rada, H. Mili, E. Bichnell, and M. Blettner. Development and application of a metric on semantic nets. *IEEE Transactions on Systems, Man and Cybernetics*, 9(1):17–30, 1989.

[31] P. Resnik. Using information content to evaluate semantic similarity in a taxonomy. In *Proc. of 14th International Joint Conference on Aritificial Intelligence*, 1995.

[32] P. Resnik. Semantic similarity in a taxonomy: An information based measure and its application to problems of ambiguity in natural language. *Journal of Aritificial Intelligence Research*, 11:95–130, 1999.

[33] P. Resnik and N. A. Smith. The web as a parallel corpus. *Computational Linguistics*, 29(3):349–380, 2003.

[34] R. Rosenfield. A maximum entropy approach to adaptive statistical modelling. *Computer Speech and Language*, 10:187–228, 1996.

[35] H. Rubenstein and J. Goodenough. Contextual correlates of synonymy. *Communications of the ACM*, 8:627–633, 1965.

[36] M. Sahami and T. Heilman. A web-based kernel function for measuring the similarity of short text snippets. In *Proc. of 15th International World Wide Web Conference*, 2006.

[37] H. Schutze. Automatic word sense discrimination. *Computational Linguistics*, 24(1):97–123, 1998.

[38] P. D. Turney. Minning the web for synonyms: Pmi-ir versus lsa on toefl. In *Proc. of ECML-2001*, pages 491–502, 2001.

[39] A. Tversky. Features of similarity. *Psychological Review*, 84(4):327–352, 1997.

[40] B. Vlez, R. Wiess, M. Sheldon, and D. Gifford. Fast and effective query refinement. In *Proc. of 20th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval*, pages 6–15, 1997.

[41] D. M. Y. Li, Zuhair A. Bandar. An approch for measuring semantic similarity between words using multiple information sources. *IEEE Transactions on Knowledge and Data Engineering*, 15(4):871–882, 2003.