Measuring Semantic Similarity between Words using Web Search Engines

Danushka Bollegala [1,*], Yutaka Matsuo, Mitsuru Ishizuka

The University of Tokyo, Hongo 7-3-1, Tokyo, 113-8656, Japan.

Abstract

Measuring the semantic similarity between words is an important component in various semantic web-related applications such as community mining, relation extraction and automatic meta data extraction. Despite the usefulness of semantic similarity measures in these applications, accurately measuring semantic similarity between two words (or entities) remains a challenging task. We propose a 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 lexical-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 0.867. 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.

Key words: semantic similarity measures, web mining, community extraction

* Corresponding author.
Email addresses: danushka@mi.ci.i.u-tokyo.ac.jp (Danushka Bollegala), matsuo@biz-model.t.u-tokyo.ac.jp (Yutaka Matsuo), ishizuka@i.u-tokyo.ac.jp (Mitsuru Ishizuka).
[1] Research Fellow of the Japan Society for the Promotion of Science (JSPS)

1 Introduction

Accurately measuring the semantic similarity between words has been an important problem in Semantic Web. information retrieval and natural language processing. In Semantic Web applications such as community extraction, ontology generation and entity disambiguation requires the ability to accurately measure semantic relatedness between concepts or entities. In information retrieval one of the main problems is to retrieve a set of documents that is semantically related to a given user query. Efficient measurement of semantically similarity between words is critical for various natural language processing tasks such as word sense disambiguation (WSD), textual entailment and automatic text summarization.

Manually created general-purpose lexical ontologies such as WordNet, groups semantically related words (i.e. synsets) for concepts. However, 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 ontologies 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 not feasible 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. In general, page count may not necessarily be equal to the word frequency because the queried word might appear many times on one page. 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 ignores the position of a word in a page. Therefore, even though two words appear in a page, they might not be actually related. Secondly, page count of a polysemous word (a word with multiple senses) might contain a combination of all its senses. For an example,

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

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.

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 [42], 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.

We 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"*

Fig. 1. A snippet retrieved for the query ***Jaguar*** *AND* ***cat***.

Here, the phrase *is the largest* indicates a hypernymic relationship between Jaguar and 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 hypernym extraction [15] and fact extraction [33]. From the previous example, we form the pattern ***X*** *is the largest* ***Y***, where we replace the two words *Jaguar* and *cat* by two variables **X** and **Y**.

Our contributions are summarized as follows:

- We presnt an automatically extracted lexico-syntactic patterns-based approach to compute semantic similarity between words or entities using text snippets retrieved from a web search engine.
- We integrate different web-based similarity measures using a machine learning approach. We extract synonymous word-pairs from WordNet synsets as positive training instances and automatically generate negative training instances. We then train a two-class support vector machine to classify similar and non-similar word-pairs. 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.
- We apply the proposed semantic similarity measure to identify relations between entities, in particular people, in a community extraction task. In this experiment, the proposed method outperforms the baselines with statistically significant precision and recall values. The results of the experiments show the ability of the proposed method to measure the semantic similarity between not only words but also between named entities, for which manually created lexical ontologies do not exist or incomplete.

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 Section 4.7 we apply the proposed semantic similarity measure in a community mining task to evaluate its ability to capture the semantic similarity between real-world entities.

2 Related Work

Semantic similarity measures are important in many Web-related tasks. In query expansion [5,30,46] 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 [9], community mining [27,23], and keyword extraction for inter-entity relation representation [31]. Cimiano et al., [9] propose PANKOW (Pattern-based Annotation through Knowledge on the Web) system to automatically annotate a web page with metadata. Given a web page to annotate, the PANKOW system first extract candidate phrases such as proper names. It then classify the extracted candidate phrases into a set of given concepts (e.g. Country, Hotel) using the number of hits returned from a web search engine for lexical patterns such as ***X** is a **Y***, *the **X** **Y***, etc. Matsuo et al., [23] 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).

Semantic similarity measures are necessary for various applications in natural language processing such as word-sense disambiguation [38], language modeling [40], synonym extraction [20], and automatic thesauri extraction [10]. Manually compiled taxonomies such as WordNet [3] and large text corpora have been used in previous works on semantic similarity [20,37,16,21]. 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 [17,18]. Resnik and Smith [39] extracted bilingual sentences from the Web to create a parallel corpora for machine translation. Turney [44] 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, [24] 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 [36]. 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 [37] 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., [47] 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 [29] benchmark dataset. They did not evaluate their method in terms of similarities among named entities. Lin [21] defined the similarity between two concepts as the information that is in common to both concepts and the information contained in each individual concept.

Cilibrasi and Vitanyi [8] propose a distance metric between words using only page-counts retrieved from a web search engine. The proposed metric is named *Normalized Google Distance* (NGD) and it is given by,

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

$$NGD(x,y) = \frac{\max\{\log H(x), \log H(y)\} - \log H(x,y)}{\log N - \min\{\log H(x), \log H(y)\}}. \quad (1)$$

Here, x and y are the two words between which distance $NGD(x,y)$ is to be computed, $f(x)$ denotes the page-counts for the word x, and $f(x,y)$ is the page-counts for the query *x AND y*. NGD is based on normalized information distance [19] which is defined using Kolmogorov complexity. Because NGD does not take into account the context in which the words co-occur, it suffers from the drawbacks described in the previous section unique to measures that consider only page-counts to compute the relatedness between words.

Sahami et al., [42] 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 The *Co-occurrence Double-Checking* (CODC) measure is defined as,

$$\mathrm{CODC}(P,Q) = \begin{cases} 0 & \text{if } f(P@Q) = 0 \\ \exp\left(\log\left[\frac{f(P@Q}{H(P)} \times \frac{f(Q@P)}{H(Q)}\right]^{\alpha}\right) & \text{otherwise.} \end{cases} \quad (2)$$

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 experimentally set to 0.15. 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 because a search engine considers many other factors besides semantic similarity, such as publication date (novelty) and link structure (authority) when ranking the result set for a query. 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 [29] benchmark dataset.

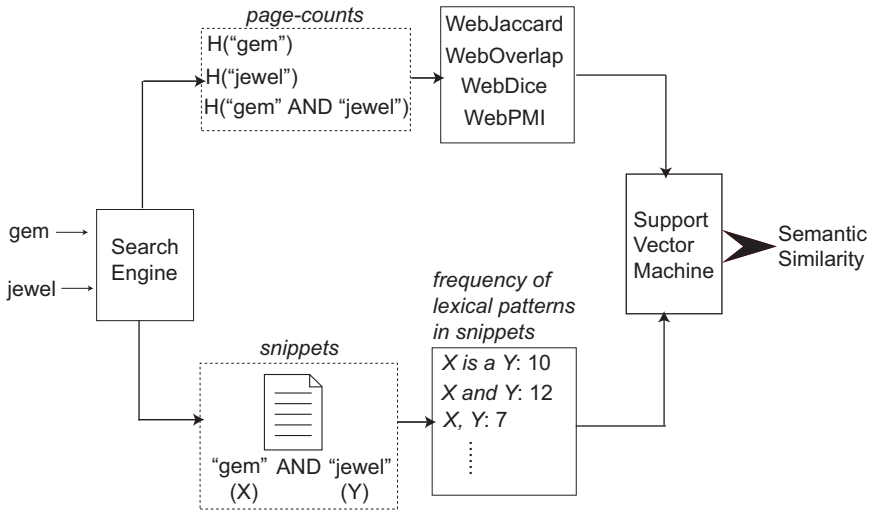

Fig. 2. Outline of the proposed method.

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. Figure 2 illustrates an example of using the proposed method to compute the semantic similarity between two words *gem* and *jewel*. First, we query a Web search engine using the two words between which we must compute semantic similarity and retrieve page-counts for the two words and for their conjunctive (i.e. *AND*) query. In section 3.2, we define four similarity scores using page counts. Second, we find the frequency of numerous lexico-syntactic patterns in snippets returned for the conjunctive query of the two words. The lexical patterns we utilize are extracted using the method described 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 instances) from WordNet *synsets* (i.e. a set of synonymous words). Non-synonymous word-pairs (negative training instances) 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. 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 Pointwise mutual information (PMI), 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, $\mathrm{WebJaccard}(P,Q)$, is defined as,

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

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 [4]. Similarly, we define *WebOverlap*, $\mathrm{WebOverlap}(P,Q)$, as,

$$\mathrm{WebOverlap}(P,Q) = \begin{cases} 0 & \text{if } H(P \cap Q) \leq c \\ \frac{H(P \cap Q)}{\min(H(P),H(Q))} & \text{otherwise} \end{cases}. \quad (4)$$

WebOverlap is a natural modification to the Overlap (Simpson) coefficient. We define the *WebDice* coefficient as a variant of the Dice coefficient. $\mathrm{WebDice}(P,Q)$ is defined as,

$$\mathrm{WebDice}(P,Q) = \begin{cases} 0 & \text{if } H(P \cap Q) \leq c \\ \frac{2H(P \cap Q)}{H(P)+H(Q)} & \text{otherwise} \end{cases}. \quad (5)$$

Pointwise mutual information (PMI) [7] is a measure that is motivated by information theory; it is intended to reflect the dependence between two probabilistic

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

events. We define *WebPMI* as a variant form of pointwise mutual information using page counts as,

$$\text{WebPMI}(P,Q) = \begin{cases} 0 & \text{if } H(P \cap Q) \leq c \\ \log_2(\frac{\frac{H(P \cap Q)}{N}}{\frac{H(P)}{N}\frac{H(Q)}{N}}) & \text{otherwise} \end{cases}. \quad (6)$$

Here, N is the number of documents indexed by the search engine. Probabilities in Eq. 6 are estimated according to the maximum likelihood principle. To calculate PMI accurately using Eq. 6, 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.

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"*.

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

Fig. 3. Pattern Extraction Example

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 variables *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.

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

Fig. 4. 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

Algorithm 1: EXTRACTPATTERNS(S)

comment: Given a set S of word-pairs, extract patterns.

for each wordpair $(A, B) \in S$

  do $D \leftarrow \text{GetSnippets}(\text{“}A\ B\text{”})$

$N \leftarrow null$

for each snippet $d \in D$

  do $N \leftarrow N + \text{GetNgrams}(d, A, B)$

$Pats \leftarrow \text{CountFreq}(N)$

return $(Pats)$

Fig. 5. Extract patterns from snippets.

snippets we need to process on the Web. 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 5. 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 variables. Let us assume these variables 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 4 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.

To leverage the pattern extraction process, we select synonymous nouns from WordNet [28] synsets. A WordNet synset contains a group of synonymous words. Different senses of a word have different synsets. The WordNet 2.0 database which we used in our experiments contains $114,648$ nouns and $79,689$ synsets. We randomly selected 5000 nouns for which synsets with more than three entries are available. We do not select abbreviations or multi-word nouns. For polysemous nouns we selected the synonyms for the dominant sense. The pattern extraction algorithm described in Figure 5 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

Table 1
Contingency table

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

synonymy. For that purpose, we employ the following procedure.

First, we run the pattern extraction algorithm described in Figure 5 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.

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 [22] value for each pattern as,

$$\chi^2 = \frac{(P+N)(p_v(N-n_v) - n_v(P-p_v))^2}{PN(p_v+n_v)(P+N-p_v-n_v)}. \tag{7}$$

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 [45,26]. 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 [26]. 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

Algorithm 2: GETFEATUREVECTOR(A, B)

comment: Given a word-pair A, B get its feature vector F.

$D \leftarrow \text{GetSnippets}(\text{“}A\ B\text{”})$

$N \leftarrow null$

for each snippet$d \in D$

  do $N \leftarrow N + \text{GetNgrams}(d, A, B)$

$SelPats \leftarrow \text{SelectPatterns}(N, GoodPats)$

$PF \leftarrow \text{Normalize}(SelPats)$

$F \leftarrow [PF, WebJaccard, WebOverlap, WebDice, WebPMI]$

return (F)

Fig. 6. Create a feature vector F for a word-pair (A, B).

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.

For each pair of words (A, B), we create a feature vector F as shown in Figure 6. First, we query Google for *“A” AND “B”* and collect snippets. Then we replace the query words A and B with two variables 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 5. 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 $\mathbf{x_i}$ for the word-pair (A_i, B_i). 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 labeled 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 $\mathbf{x}^*$ for the given pair of words (A^*, B^*), between which we must measure the semantic similarity. We define the semantic similarity $\mathrm{SemSim}(A^*, B^*)$ between A^* and B^* as the posterior probability $P(y^* = 1|f(\mathbf{x}^*))$ that feature vector $\mathbf{x}^*$ corresponding to the word-pair (A^*, B^*) belongs to the synonymous-words class. We denote the label assigned to a feature vector $\mathbf{x_i}$ by $y_i \in \{-1, 1\}$. Here, $y_i = 1$ denotes the synonymous-words class and $y_i = -1$ denotes the non-synonymous words class. $\mathrm{SemSim}(A^*, B^*)$ is given by,

$$\mathrm{SemSim}(A^*, B^*) = P(y^* = 1|f(\mathbf{x}^*)). \tag{8}$$

Here, $\mathbf{f}(\mathbf{x}^*)$ is the distance to the instance x^* from the classification hyperplane. It is given by,

$$f(\mathbf{x}^*) = h(\mathbf{x}^*) + b.$$

Here, b is the bias term and the hyperplane, $\mathbf{h}(\mathbf{x})$, is given by,

$$h(\mathbf{x}^*) = \sum_i y_i \alpha_i K(\mathbf{x_i}, \mathbf{x}^*).$$

Here, α_i is the Lagrange multiplier corresponding to the support vector $\mathbf{x_i}$ [5], and $K(\mathbf{x_i}, \mathbf{x}^*)$ is the value of the kernel function for a training instance $\mathbf{x_i}$ and the instance to classify, $\mathbf{x}^*$.

Because it is 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. Following Platt [35], we use sigmoid functions to convert this uncalibrated distance into a calibrated posterior probability. The probability, $P(y = 1|f(\mathbf{x}))$, is computed using a sigmoid function defined over $f(\mathbf{x})$ as follows,

$$P(y = 1|f(\mathbf{x})) = \frac{1}{1 + \exp(Af(\mathbf{x}) + B)}.$$

[5] From K.K.T. conditions it follows that the Lagrange multipliers corresponding to non-support vectors become zero

Here, A and B are parameters which are determined by maximizing the likelihood of the training data. Log-likelihood of the training data is given by,

$$L(A,B) = \sum_{i=1}^{N} \log P(y_i|\mathbf{x_i}; A, B) \tag{9}$$
$$= \sum_{i=1}^{N} \{t_i \log(p_i) + (1-t_i)\log(1-p_i)\}.$$

Here, to simplify the formula we have used the notations $t_i = (y_i + 1)/2$ and $p_i = P(y_i = 1|\mathbf{x_i})$. The maximization in Formula 9 with respect to parameters A and B can be done using various optimization algorithms [32]. Platt [35] used a model-trust minimization algorithm [13] for the optimization.

4 Experiments

Evaluating a semantic similarity measure is a difficult task because the notion of semantic similarity varies across domains and from person to person. To evaluate the proposed method we conduct two types of experiments. First, we compare the similarity scores produced by the proposed measure against a benchmark dataset of semantic similarity. The benchmark dataset is detailed in section 4.1. The degree of correlation between a benchmark dataset of semantic similarity and the similarity scores produced by an automatic similarity measure, can be considered as a measurement of how well the automatic similarity measure captures the notion of semantic similarity held by humans. 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. This enables us to evaluate the performance of the proposed method in measuring semantic similarity between named entities for which no manually created lexical resources such as dictionaries exist.

4.1 The Benchmark Dataset

We evaluate the proposed method against Miller-Charles [29] 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

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

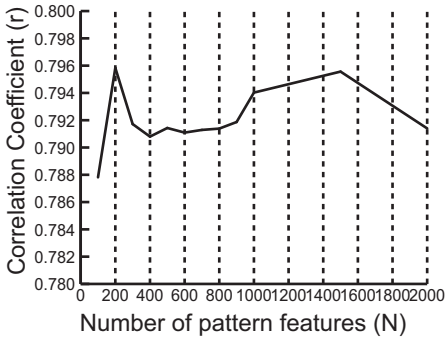

Fig. 7. Correlation vs. No of pattern features

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

a subset of Rubenstein-Goodenough's [41] 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 is 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 7. In Figure 7 a steep improvement of correlation with the number of top-ranking patterns is apparent; it reaches a maximum at 200 features. With more than 200 patterns correlation drops below this maximum. Considering that the patterns are ranked according to

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

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 [42], Chen [6]), Normalized Google Distance (NGD) [8], and the proposed method (SVM). Results are shown in Table 4. Because NGD is a distance measure and all other measures compared in Table 4 are similarity scores, for consistency with other measures, we consider the value after deducting NGD from one in the NGD column in Table 4. All figures, except those for the Miller-Charles ratings, are normalized into values in $[0, 1]$ range for the 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

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

Word Pair	MC	Web Jaccard	Web Dice	Web Overlap	Web PMI	NGD	Sahami [42]	CODC [6]	SVM
automobile-car	3.920	0.650	0.664	0.831	0.427	0.466	0.225	0.008	0.918
journey-voyage	3.840	0.408	0.424	0.164	0.468	0.556	0.121	0.005	1.000
gem-jewel	3.840	0.287	0.300	0.075	0.688	0.566	0.052	0.012	0.817
boy-lad	3.760	0.177	0.186	0.593	0.632	0.456	0.109	0.000	0.958
coast-shore	3.700	0.783	0.794	0.510	0.561	0.603	0.089	0.006	0.975
asylum-madhouse	3.610	0.013	0.014	0.082	0.813	0.782	0.052	0.000	0.794
magician-wizard	3.500	0.287	0.301	0.370	0.863	0.572	0.057	0.008	0.997
midday-noon	3.420	0.096	0.101	0.116	0.586	0.687	0.069	0.010	0.987
furnace-stove	3.110	0.395	0.410	0.099	1.000	0.638	0.074	0.011	0.878
food-fruit	3.080	0.751	0.763	1.000	0.449	0.616	0.045	0.004	0.940
bird-cock	3.050	0.143	0.151	0.144	0.428	0.562	0.018	0.006	0.867
bird-crane	2.970	0.227	0.238	0.209	0.516	0.563	0.055	0.000	0.846
implement-tool	2.950	1.000	1.000	0.507	0.297	0.750	0.098	0.005	0.496
brother-monk	2.820	0.253	0.265	0.326	0.623	0.495	0.064	0.007	0.265
crane-implement	1.680	0.061	0.065	0.100	0.194	0.559	0.039	0.000	0.056
brother-lad	1.660	0.179	0.189	0.356	0.645	0.505	0.058	0.005	0.132
car-journey	1.160	0.438	0.454	0.365	0.205	0.410	0.047	0.004	0.165
monk-oracle	1.100	0.004	0.005	0.002	0.000	0.579	0.015	0.000	0.798
food-rooster	0.890	0.001	0.001	0.412	0.207	0.568	0.022	0.000	0.018
coast-hill	0.870	0.963	0.965	0.263	0.350	0.669	0.070	0.000	0.356
forest-graveyard	0.840	0.057	0.061	0.230	0.495	0.612	0.006	0.000	0.442
monk-slave	0.550	0.172	0.181	0.047	0.611	0.698	0.026	0.000	0.243
coast-forest	0.420	0.861	0.869	0.295	0.417	0.545	0.060	0.000	0.150
lad-wizard	0.420	0.062	0.065	0.050	0.426	0.657	0.038	0.000	0.231
cord-smile	0.130	0.092	0.097	0.015	0.208	0.460	0.025	0.000	0.006
glass-magician	0.110	0.107	0.113	0.396	0.598	0.488	0.037	0.000	0.050
rooster-voyage	0.080	0.000	0.000	0.000	0.228	0.487	0.049	0.000	0.052
noon-string	0.080	0.116	0.123	0.040	0.102	0.488	0.024	0.000	0.000
Correlation	**1.000**	**0.260**	**0.267**	**0.382**	**0.549**	**0.205**	**0.580**	**0.694**	**0.867**

wizard. Lowest similarity is reported for *cord* and *smile* We did not use any of the words in the benchmark dataset or their synsets for training. Our reimplementation of Co-occurrence Double Checking (CODC) measure [6] indicates the second-best correlation of 0.6936. 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 apparent from the definition of the CODC measure in Eq. 2, 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. [42] 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 ($r = 0.5489$). NGD considers only page-counts and it reports a low correlation (0.205) with Miller-Charles' ratings. 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 between words.

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) [21] Jiang & Conrath (1998) [16] and Li (2003) [47] methods. Unlike the WordNet based methods, proposed method requires no a hierarchical taxonomy of

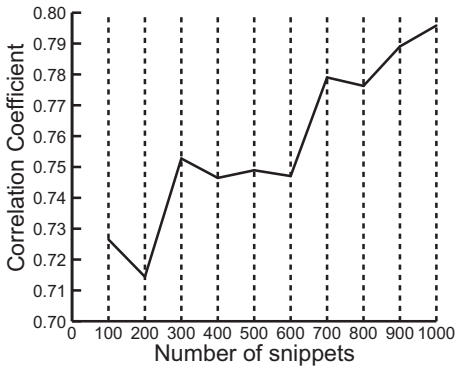

Fig. 8. Correlation vs. No of snippets

concepts or sense-tagged definitions of words. 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. We started with 100 snippets and increased this number by 100 snippets at a time. Because of the constraints imposed by Google on the maximum number of snippets that can be collected for a query, we were limited to a maximum of 1000 snippets. The experimental results are presented in Figure 8. From Figure 8 it is apparent 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 9. Maximum correlation coefficient of 0.8345 is achieved with 1900 positive

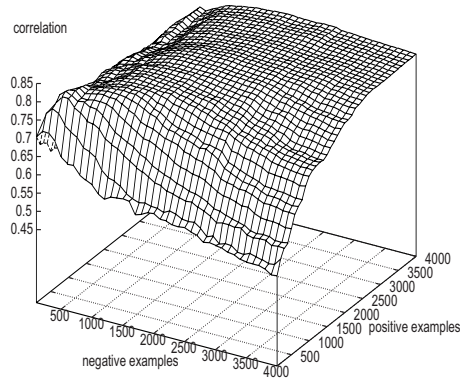

Fig. 9. Correlation vs. No of positive and negative training instances

training examples and 2400 negative training examples. Moreover, Figure 9 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.

4.7 Community Mining

Measuring semantic similarity between named entities is vital in many applications such as query expansion [42], entity disambiguation (e.g. namesake disambiguation) and community mining [23]. 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) [7] . 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, $\mathrm{Corr}(\Gamma)$ between two clusters A and B is defined as the fol-

[7] http://dmoz.org

lowing,

$$\mathrm{Corr}(\Gamma) = \frac{1}{2}\frac{1}{|\Gamma|(|\Gamma|-1)}\sum_{(u,v)\in\Gamma}\mathrm{sim}(u,v) \tag{10}$$

Here, Γ is the merger of the two clusters A and B. $|\Gamma|$ denotes the number of elements (persons) in Γ and $\mathrm{sim}(u,v)$ 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 $\mathrm{sim}(u,v)$ 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,

$$\mathrm{Precision}(p) = \frac{\text{No. of people in C(p) with affiliation A(p)}}{\text{No. of people in C(p)}}, \tag{11}$$

$$\mathrm{Recall}(p) = \frac{\text{No. of people in C(p) with affiliation A(p)}}{\text{Total No. of people with affiliation A(p)}}. \tag{12}$$

Since, we selected 10 people from each of the five categories, the denominator in Formula 12 is 10 for all the names p.

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

$$\mathrm{F}(p) = \frac{2\times\mathrm{Precision}(p)\times\mathrm{Recall}(p)}{\mathrm{Precision}(p)+\mathrm{Recall}(p)}. \tag{13}$$

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

$$\mathrm{Precision} = \frac{1}{N}\sum_{p\in DataSet}\mathrm{Precision}(p) \tag{14}$$

$$\mathrm{Recall} = \frac{1}{N}\sum_{p\in DataSet}\mathrm{Recall}(p) \tag{15}$$

$$F\text{−Score} = \frac{1}{N}\sum_{p\in DataSet}\mathrm{F}(p) \tag{16}$$

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 [42]	0.6384	0.668	0.6426
Chen [6]	0.4763	0.624	0.4984
Proposed	0.7958	0.804	0.7897

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. [42]'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 [11], social network extraction [23,3,4], Word Sense Disambiguation (WSD) [25], citation matching [14] and cross-document co-reference resolution [34,12].

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 [43] 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

Table 7
Entity Disambiguation Results

Method	Jaguar			Java		
	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 [42]	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

similarity $\mathrm{SIM}(S_a, S_b)$ between two snippets S_a,$S_b$as follows,

$$\mathrm{SIM}(S_a, S_b) = \frac{1}{|S_a||S_b|} \sum_{a \in S_a, b \in S_b} sim(a, b) \tag{17}$$

In Formula 17 $|S|$ denotes the number of words in snippet S. We used different semantic similarity measures for sim in Formula 17 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.

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.

References

[1] A. Bagga, B. Baldwin, Entity-based cross document coreferencing using the vector space model, in: Proc. of 36th COLING-ACL, 1998.

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

[3] R. Bekkerman, A. McCallum, Disambiguating web appearances of people in a social network, in: Proceedings of the World Wide Web Conference (WWW), 2005.

[4] D. Bollegala, Y. Matsuo, M. Ishizuka, Disambiguating personal names on the web using automatically extracted key phrases, in: Proc. of the 17th European Conference on Artificial Intelligence, 2006.

[5] C. Buckley, G. Salton, J. Allan, A. Singhal, Automatic query expansion using smart: Trec 3, in: Proc. of 3rd Text REtreival Conference, 1994.

[6] H. Chen, M. Lin, Y. Wei, Novel association measures using web search with double checking, in: Proc. of the COLING/ACL 2006, 2006.

[7] K. Church, P. Hanks, Word association norms, mutual information and lexicography, Computational Linguistics 16 (1991) 22–29.

[8] R. Cilibrasi, P. Vitanyi, The google similarity distance, IEEE Transactions on Knowledge and Data Engineering 19 (3) (2007) 370–383.

[9] P. Cimano, S. Handschuh, S. Staab, Towards the self-annotating web, in: Proc. of 13th WWW, 2004.

[10] J. Curran, Ensemble menthods for automatic thesaurus extraction, in: Proc. of EMNLP, 2002.

[11] D. R. Cutting, J. O. Pedersen, D. Karger, J. W. Tukey, Scatter/gather: A cluster-based approach to browsing large document collections, in: Proceedings SIGIR '92, 1992.

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

[13] P. Gill, W. Murray, M. Wright, Practical optimization, Academic Press, 1981.

[14] H. Han, H. Zha, 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.

[15] M. Hearst, Automatic acquisition of hyponyms from large text corpora, in: Proc. of 14th COLING, 1992.

[16] J. Jiang, 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.

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

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

[19] M. Li, X. Chen, X. Li, B. Ma, P. Vitanyi, The similarity metric, IEEE Transactions on Information Theory 50 (12) (2004) 3250–3264.

[20] D. Lin, Automatic retreival and clustering of similar words, in: Proc. of the 17th COLING, 1998.

[21] D. Lin, An information-theoretic definition of similarity, in: Proc. of the 15th ICML, 1998.

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

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

[24] Y. Matsuo, T. Sakaki, K. Uchiyama, M. Ishizuka, Graph-based word clustering using web search engine, in: Proc. of EMNLP 2006, 2006.

[25] D. McCarthy, R. Koeling, J. Weeds, J. Carroll, Finding predominant word senses in untagged text, in: Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL'04), 2004.

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

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

[28] G. Miller, Wordnet: a lexical database for english, Commun. ACM 38 (11) (1995) 39–41.

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

[30] M. Mitra, A. Singhal, C. Buckley, Improving automatic query expansion, in: Proc. of 21st Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval, 1998.

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

[32] J. Nocedal, S. Wright, Numerical optimization, Springer, 2000.

[33] M. Pasca, D. Lin, J. Bigham, A. Lifchits, 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.

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

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

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

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

[38] 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 (1999) 95–130.

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

[40] R. Rosenfield, A maximum entropy approach to adaptive statistical modelling, Computer Speech and Language 10 (1996) 187–228.

[41] H. Rubenstein, J. Goodenough, Contextual correlates of synonymy, Communications of the ACM 8 (1965) 627–633.

[42] M. Sahami, 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.

[43] H. Schutze, Automatic word sense discrimination, Computational Linguistics 24 (1) (1998) 97–123.
URL `citeseer.ist.psu.edu/schutze98automatic.html`

[44] P. D. Turney, Minning the web for synonyms: Pmi-ir versus lsa on toefl, in: Proc. of ECML-2001, 2001.

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

[46] B. Vlez, R. Wiess, M. Sheldon, D. Gifford, Fast and effective query refinement, in: Proc. of 20th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval, 1997.

[47] 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) (2003) 871–882.