Even Unassociated Features Can Improve Lexical Distributional Similarity

Kazuhide Yamamoto and **Takeshi Asakura**
Department of Electrical Engineering
Nagaoka University of Technology
{yamamoto, asakura}@jnlp.org

Abstract

This paper presents a new computation of lexical distributional similarity, which is a corpus-based method for computing similarity of any two words. Although the conventional method focuses on emphasizing features with which a given word is associated, we propose that even unassociated features of two input words can further improve the performance in total. We also report in addition that more than 90% of the features has no contribution and thus could be reduced in future.

1 Introduction

Similarity calculation is one of essential tasks in natural language processing (1990; 1992; 1994; 1997; 1998; 1999; 2005). We look for a semantically similar word to do corpus-driven summarization, machine translation, language generation, recognition of textual entailment and other tasks. In task of language modeling and disambiguation we also need to semantically generalize words or cluster words into some groups. As the amount of text increases more and more in the contemporary world, the importance of similarity calculation also increases concurrently.

Similarity is computed by roughly two approaches: based on thesaurus and based on corpus. The former idea uses thesaurus, such as WordNet, that is a knowledge resource of hierarchical word classification. The latter idea, that is the target of our work, originates from Harris's distributional hypothesis more than four decades ago (1968), stating that semantically similar words tend to appear in similar contexts. In many cases a context of a word is represented as a feature vector, where each feature is another expression that co-occurs with the given word in the context.

Over a long period of its history, in particular in recent years, several works have been done on distributional similarity calculation. Although the conventional works have attained the *fine* performance, we attempt to further improve the quality of this measure. Our motivation of this work simply comes from our observation and analysis of the output by conventional methods; Japanese, our target language here, is written in a mixture of four scripts: Chinese characters, Latin alphabet, and two Japanese-origin characters. In this writing environment some words which have same meaning and same pronunciation are written in two (or more) different scripts. This is interesting in terms of similarity calculation since these two words are completely same in semantics so the similarity should be ideally 1.0. However, the reality is, as far as we have explored, that the score is far from 1.0 in many *same* word pairs. This fact implies that the conventional calculation methods are far enough to the goal and are expected to improve further.

The basic framework for computing distributional similarity is same; for each of two input words a context (i.e., surrounding words) is extracted from a corpus, a vector is made in which an element of the vector is a value or a weight, and two vectors are compared with a formula to compute similarity. Among these processes we have focused on features, that are elements of

Proceedings of the Second Workshop on NLP Challenges in the Information Explosion Era (NLPIX 2010), pages 32–39,
Beijing, August 2010

the vector, some of which, we think, adversely affect the performance. That is, traditional approaches such as Lin (1998) basically use all of observed words as context, that causes noise in feature vector comparison. One may agree that the number of the characteristic words to determine the meaning of a word is some, not all, of words around the target word. Thus our goal is to detect and reduce such noisy features.

Zhitomirsky-Geffet and Dagan (2009) have same motivation with us and introduced a bootstrapping strategy that changes the original features weights. The general idea here is to promote the weights of features that are common for associated words, since these features are likely to be most characteristic for determining the word's meaning. In this paper, we propose instead a method to using features that are both unassociated to the two input words, in addition to use of features that are associated to the input.

2 Method

The lexical distributional similarity of the input two words is computed by comparing two vectors that express the context of the word. In this section we first explain the feature vector, and how we define initial weight for each feature of the vector. We then introduce in Subsection 2.3 the way to compute similarity by two vectors. After that, we emphasize some of the features by their association to the word, that is explained in Subsection 2.4. We finally present in Subsection 2.5 feature reduction which is our core contribution of this work. Although our target language is Japanese, we use English examples in order to provide better understanding to the readers.

2.1 Feature Vector

We first explain how to construct our feature vector from a text corpus.

A word is represented by a feature vector, where features are collection of syntactically dependent words co-occurred in a given corpus. Thus, we first collect syntactically dependent words for each word. This is defined, as in Lin (1998), as a triple (w, r, w'), where w and w' are words and r is a syntactic role. As for definition of *word*, we use not only words given by a morphological analyzer but also compound words. Nine case particles are used as syntactic roles, that roughly express subject, object, modifier, and so on, since they are easy to be obtained from text with no need of semantic analysis. In order to reduce noise we delete triples that appears only once in the corpus.

We then construct a feature vector out of collection of the triples. A feature of a word is an another word syntactically dependent with a certain role. In other words, given a triple (w, r, w'), a feature of w corresponds to a dependent word with a role (r, w').

2.2 (Initial) Filtering of Features

There are several weighting functions to determine a value for each feature element. As far as we have investigated the literature the most widely used feature weighting function is pointwise mutual information (MI), that is defined as follows:

$$MI(w, r, w') = \log_2 \frac{freq(w, r, w')S}{freq(w)freq(r, w')} \quad (1)$$

where $freq(r, w')$ is the frequency of the co-occurrence word w' with role r, $freq(w)$ is the independent frequency of a word w, $freq(w, r, w')$ is the frequency of the triples (w, r, w'), and S is the number of all triples.

In this paper we do not discuss what is the best weighting functions, since this is out of target. We use mutual information here because it is most widely used, i.e., in order to compare performance with others we want to adopt the standard approach.

As other works do, we filter out features that have a value lower than a minimal weight thresholds α. The thresholds are determined according to our preliminary experiment, that is explained later.

2.3 Vector Similarity

Similarity measures of the two vectors are computed by various measures. Shibata and Kurohashi (2009) have compared several similarity measures including Cosine (Ruge, 1992), (Lin,

(input word) w: boy
(feature) v: $\underline{\text{guard}_{\text{OBJ}}}$
(synonyms of w, shown with its similarity to w) $Syn(w)$ =
{ child(0.135), girl(0.271), pupil(0.143), woman(0.142), young people(0.147) }

(feature vectors V):
V(boy) = { $\text{parents}_{\text{MOD}}$, $\text{runaway}_{\text{SUBJ}}$, $\text{reclaim}_{\text{OBJ}}$, $\text{father}_{\text{MOD}}$, $\underline{\text{guard}_{\text{OBJ}}}$, $\cdots$ }

V(child) = { $\underline{\text{guard}_{\text{OBJ}}}$, look_{OBJ}, $\text{bring}_{\text{OBJ}}$, $\text{give birth}_{\text{OBJ}}$, care_{OBJ} , $\cdots$ }
V(girl) = { $\text{parents}_{\text{MOD}}$, $\underline{\text{guard}_{\text{OBJ}}}$, $\text{father}_{\text{MOD}}$, $\text{testify}_{\text{SUBJ}}$, look_{OBJ}, $\cdots$ }
V(pupil) = { $\text{target}_{\text{OBJ}}$, $\underline{\text{guard}_{\text{OBJ}}}$, care_{OBJ}, aim_{OBJ}, $\text{increase}_{\text{SUBJ}}$, $\cdots$ }
V(woman) = { name_{MOD}, $\text{give birth}_{\text{OBJ}}$, $\text{group}_{\text{MOD}}$, together+with, $\text{parents}_{\text{MOD}}$, $\cdots$ }
V(young people) = { $\text{harmful}_{\text{TO}}$, $\text{global}_{\text{MOD}}$, $\text{reclaim}_{\text{OBJ}}$, $\text{wrongdoing}_{\text{MOD}}$, $\cdots$ }

(words that has feature v) $Asc(v)$ = {boy, child, girl, pupil, $\cdots$}

$weight(w, v) = weight$ (boy, $\text{guard}_{\text{OBJ}}$) $= \sum_{w_f \in Asc(v) \cap Syn(w)} sim(w, w_f)$
$= 0.135 + 0.271 + 0.143 = 0.549$

Figure 1: Example of feature weighting for word *boy*.

1998), (Lin, 2002), Simpson, Simpson-Jaccard, and conclude that Simpson-Jaccard index attains best performance of all. Simpson-Jaccard index is an arithmetic mean of Simpson index and Jaccard index, defined in the following equation:

$$sim(w_1, w_2) = \frac{1}{2}(sim_J(w_1, w_2) + sim_S(w_1, w_2)) \quad (2)$$

$$sim_J(w_1, w_2) = \frac{|V_1 \cap V_2|}{|V_1 \cup V_2|} \quad (3)$$

$$sim_S(w_1, w_2) = \frac{|V_1 \cap V_2|}{min(|V_1|, |V_2|)} \quad (4)$$

where V_1 and V_2 is set of features for w_1 and w_2, respectively, and $|A|$ is the number of set A. It is interesting to note that both Simpson and Jaccard compute similarity according to degree of overlaps of the two input sets, that is, the reported best measure computes similarity by ignoring the weight of the features. In this paper we adopt Simpson-Jaccard index, sim, which indicates that the weight of features that is explained below is only used for feature reduction, not for similarity calculation.

2.4 Feature Weighting by Association

We then compute weights of the features of the word w according to the degree of semantic association to w. The weight is biased because all of the features, i.e., the surrounding words, are not equally characteristic to the input word. The core idea for feature weighting is that a feature v in w is more weighted when more synonyms (words of high similarity) of w also have v.

Figure 1 illustrates this process by examples. Now we calculate a feature $\text{guard}_{\text{OBJ}}$ for a word *boy*, we first collect synonyms of w, denoted by $Syn(w)$, from a thesaurus. We then compute similarities between w and each word in $Syn(w)$ by Equation 2. The weight is the sum of the similarities of words in $Syn(w)$ that have feature v, defined in Equation 5.

$$weight(w, v) = \sum_{w_f \in Asc(v) \cap Syn(w)} sim(w, w_f) \quad (5)$$

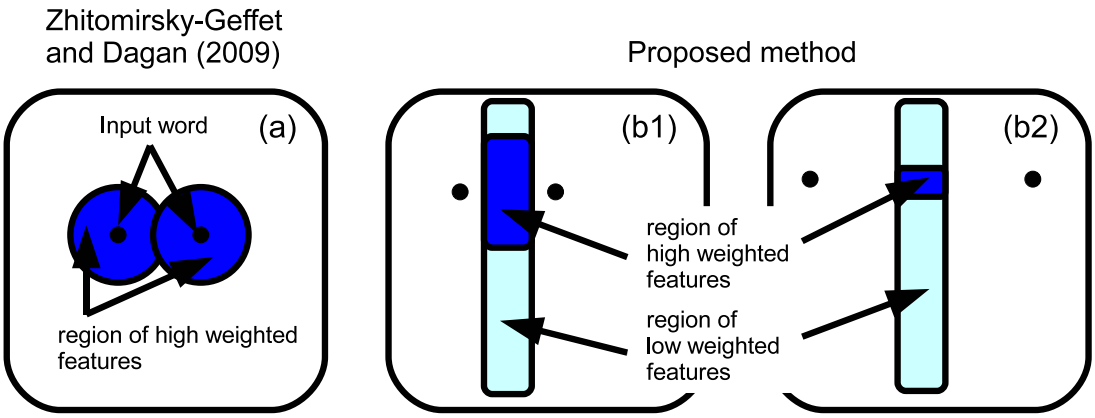

Figure 2: An illustration of similarity calculation of Zhitomirsky-Geffet and Dagan (2009) (a) and the proposed method (b1 and b2) in feature space. In order to measure the distance of the two words (shown in black dots) they use only associated words, while we additionally use unassociated words in which the distances to the words are similar.

2.5 Feature Reduction

We finally reduce features according to the difference of weights of each feature in words we compare. In computing similarity of two words, w_1 and w_2, a feature v satisfying Equation 6 is reduced.

$$abs(weight(w_1, v) - weight(w_2, v)) > \beta \quad (6)$$

where $abs()$ is a function of absolute value, and β is a threshold for feature reduction.

Figure 2 illustrates our idea and compares the similar approach proposed by Zhitomirsky-Geffet and Dagan (2009). Roughly speaking, Zhitomirsky-Geffet and Dagan (2009) compute similarity of two words, shown as black dots in (a), mainly according to *associated* features (dark-colored circle), or features that has high weights in Equation 5. And the associated features are determined word by word independently.

In contrast, the proposed method relatively reduces features, depending on *location* of input two words. At (b1) in the figure, not only associated (high-colored area) but unassociated features (light-colored area) are used for similarity computation in our method. As Equation 6 shows, regardless of how much a feature is associated to the word, the feature is not reduced when it has similar weight to both w_1 and w_2, located at the middle area of the two words in the figure.

This idea seems to work more effectively, compared with Zhitomirsky-Geffet and Dagan (2009), in case that input two words are not so similar, that is shown at (b2) of the figure. As they define associated features independently, it is likely that the overlapped area is little or none between the two words. In contrast, our method uses features at the *middle* area of two input words, where there is always certain features provided for similarity computation, shown in case (b2). Simplified explanation is that our similarity is computed as the ratio of the associated area to the unassociated area in the figure. We will verify later if the method works better in low similarity calculation.

2.6 Final Similarity

The final similarity of two words are calculated by two shrunk vectors (or feature sets) and Equation 2, that gives a value between 0 and 1.

3 Evaluation

3.1 Evaluation Method

In general it is difficult to answer how similar two given words are. Human have no way to judge correctness if computed similarity of two words is, for instance, 0.7. However, given two word pairs, such as (w, w_1) and (w, w_2), we may answer which of two words, w_1 or w_2, is more similar to w than the other one. That is, degree of similarity is defined relatively hence accuracy of similarity measures is evaluated by way of relative comparisons.

In this paper we employ an automatic evaluation method in order to reduce time, human labor, and individual variations. We first collect four levels of similar word pairs from a thesaurus[1]. Thesaurus is a resource of hierarchical words classification, hence we can collect several levels of similar word pairs according to the depth of common parent nodes that two words have. Accordingly, we constructed four levels of similarity pairs, Level 0, 1, 2, and 3, where the number increases as the similarity increases. Each level includes 800 word pairs that are randomly selected. The following examples are pairs with word *Asia* in each Level.

Example: Four similarity levels for pair of *Asia*.

Level 3(high):	Asia vs. Europe
Level 2:	Asia vs. Brazil
Level 1:	Asia vs. my country
Level 0(low):	Asia vs. system

We then combine word pairs of adjacent similarity Levels, such as Level 0 and 1, that is a test set to see low-level similarity discrimination power. The performance is calculated in terms of how clearly the measure distinguishes the different levels. In a similar fashion, Level 1 and 2, as well as 2 and 3, are combined and tested for middle-level and high-level similarity discrimination, respectively. The number of pairs in each test set is 1,600 as two Levels are combined.

[1]In this experiment we use *Bunrui Goi Hyo* also for evaluation. Therefore, this experimental setting is a kind of closed test. However, we see that the advantage to use the same thesaurus in the evaluation seems to be small.

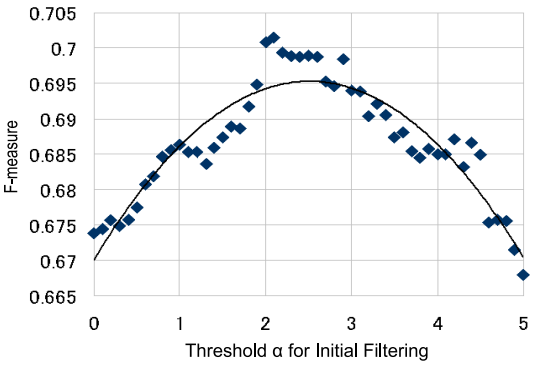

Figure 3: Relation between threshold α and performance in F-measures for Level 3+2 test set.

3.2 Experimental Setting

The corpus we use in this experiment is all the articles in *The Nihon Keizai Shimbun Database*, a Japanese business newspaper corpus covering the years 1990 through 2004. As morphological analyzer we use *Chasen* 2.3.3 with IPA morpheme dictionary. The number of collected triples is 2,584,905, that excludes deleted ones due to one time appearance and words including some symbols.

In Subsection 2.4 we use *Bunrui Goi Hyo*, a Japanese thesaurus for synonym collection. The potential target words are all content words, except words that have less than twenty features. The number of words after exclusion is 75,530. Moreover, words that have four or less words in the same category in the thesaurus are regarded as out of target in this paper, due to limitation of $Syn(w)$ in Subsection 2.4. Also, in order to avoid word sense ambiguity, words that have more than two meanings, i.e., those classified in more than two categories in the thesaurus, also remain to be solved.

3.3 Threshold for Initial Filtering

Figure 3 shows relation between threshold α and the performance of similarity distinction that is drawn in F-measures, for Level 3+2 test set. As can be seen, the plots seem to be concave down

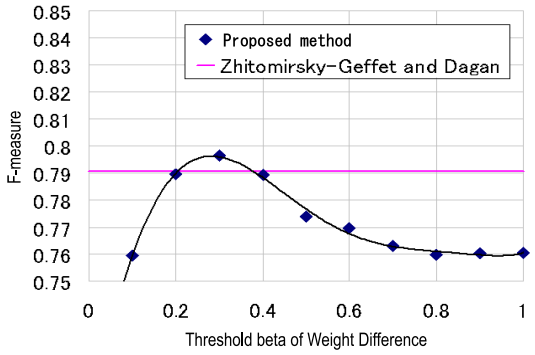

Figure 4: Threshold vs. accuracy in Level 3+2 set.

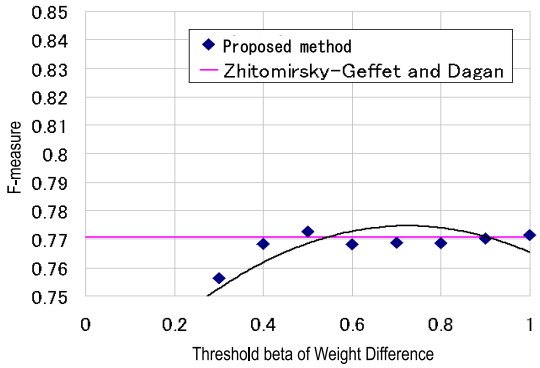

Figure 5: Threshold vs. accuracy in Level 2+1 set.

and there is a clear peak when α is between 2 and 3.

In the following experiments we set α the value where the best performance is given for each test set. We have observed similar phenomena in other test sets. The thresholds we use is 2.1 for Level 3+2, 2.4 for Level 2+1, and 2.4 for Level 1+0.

3.4 Threshold for Weighting Function

Figure 4, 5, and 6 show relation between threshold β and performance in Level 3+2, 2+1, 1+0 test set, respectively. The threshold at the point where highest performance is obtained greatly depends on Levels: 0.3 in Level 3+2, 0.5 in Level 2+1, and 0.9 in Level 1+0. Comparison of these three figures indicates that similarity distinction

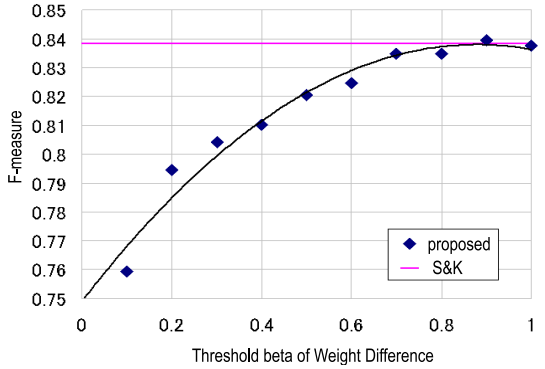

Figure 6: Threshold vs. accuracy in Level 1+0 set.

Table 1: Performance comparison of three methods in each task (in F-measures).

Level	S&K	ZG&D	proposed
Lvl.3+Lvl.2	0.702	0.791	**0.797**
Lvl.2+Lvl.1	0.747	0.771	**0.773**
Lvl.1+Lvl.0	0.838	0.789	**0.840**

power in higher similarity region requires lower threshold, i.e., fewer features. In contrast, conducting fine distinction in lower similarity level requires higher threshold, i.e., a lot of features most of which may be unassociated ones.

3.5 Performance

Table 1 shows performance of the proposed method, compared with Shibata and Kurohashi (2009) (S&K in the table) and Zhitomirsky-Geffet and Dagan (2009) (ZG&D)[2]. The method of Shibata and Kurohashi (2009) here is the best one among those compared. It uses only initial filtering described in Subsection 2.2. The method of Zhitomirsky-Geffet and Dagan (2009) in addition emphasize associated features as explained in Subsection 2.4. All of the results in the table are the best ones among several threshold settings.

The result shows that the accuracy is 0.797 (+0.006) in Level 3+2, 0.773 (+0.002) in Level

[2]The implementations of providing associated words and the bootstrapping are slightly different to Zhitomirsky-Geffet and Dagan (2009).

2+1, and 0.840 (+0.001) in Level 1+0, where the degree of improvement here are those compared with best ones except our proposed method. This confirms that our method attains equivalent or better performance in all of low, middle, and high similarity levels.

We also see in the table that S&K and ZG&D show different behavior according to the Level. However, it is important to note here that our proposed method performs equivalent or outperforms both methods in all Levels.

4 Discussions

4.1 Behavior at Each Similarity Level

As we have discussed in Subsection 2.5, our method is expected to perform better than Zhitomirsky-Geffet and Dagan (2009) in distinction in lower similarity area. Roughly speaking, we interpret the results as follows. Shibata and Kurohashi (2009) always has many features that degrades the performance in higher similarity level, since the ratio of noisy features may throw into confusion. Zhitomirsky-Geffet and Dagan (2009) reduces such noise that gives better performance in higher similarity level and is stable in all levels. And our proposed method maintains performance of Zhitomirsky-Geffet and Dagan (2009) in higher level while improves performance that is close to Shibata and Kurohashi (2009) in lower level, utilizing fewer features. We think our method can include advantages over the two methods.

4.2 Error Analysis

We overview the result and see that the major errors are NOT due to lack of features. Table 2 illustrates the statistics of words with a few features (less than 50 or 20). This table clearly tells us that, in the low similarity level (Level 1+0) in particular, there are few pairs in which the word has less than 50 or 20, that is, these pairs are considered that the features are erroneously reduced.

4.3 Estimation of Potential Feature Reduction

It is interesting to note that we may reduce 81% of features in Level 3+2 test set while keeping the performance, if we can reduce them properly. In a same way, 87% of features in Level 2+1 set, and 52% of features in Level 1+0 set, may also be reduced. These numbers are given at the situation in which F-measure attains best performance. Here, it is not to say that we are sure to reduce them in future, but to estimate how many features are really effective to distinguish the similarity.

Table 2: Relation of errors and words with a few features. In the table, (h) and (l) shows pairs that are judged higher (lower) by the system. Column of < 50 (< 20) means number of pairs each of which has less than 50 (20) features.

Level	#errs	< 50 fea.	< 20 fea.
Lvl.3+2 (h)	125	76 (61%)	32 (26%)
Lvl.3+2 (l)	220	150 (68%)	60 (27%)
Lvl.2+1 (h)	137	75 (55%)	32 (23%)
Lvl.2+1 (l)	253	135 (53%)	52 (21%)
Lvl.1+0 (h)	149	23 (15%)	4 (3%)
Lvl.1+0 (l)	100	17 (17%)	3 (3%)

Here we have more look at the statistics. The number of initial features on average is 609 in Level 3+2 test set. If we decrease threshold by 0.1, we can reduce 98% of features at the threshold of 0.8, where the performance remains best (0.791). This is a surprising fact for us since only 12 ($\fallingdotseq 609 \times (1 - 0.98)$) features really contribute the performance. Therefore, we estimate that there is a lot to be reduced further in order to purify the features.

5 Conclusion and Future Work

This paper illustrates improvement of lexical distributional similarity by not only associated features but also utilizing unassociated features. The core idea is simple, and is reasonable when we look at machine learning; in many cases we use training instances of not only something positive but something negative to make the distinction of the two sides clearer. Similarly, in our task we use features of not only associated but unassociated to make computation of similarity (or *distance* in semantic space) clearer. We as-

sert in this work that a feature that has similar weight to two given words also plays important role, regardless of how much it is associated to the given words.

Among several future works we need to further explore reduction of features. It is reported by some literature such as Hagiwara et al. (2006) that we can reduce so many features while preserving the same accuracy in distributional similarity calculation. This implies that, some of them are still harmful and are expected to be reduced further.

List of Tools and Resources

1. Chasen, a morphological analyzer, Ver.2.3.3. Matsumoto Lab., Nara Institute of Science and Technology. http://chasen-legacy.sourceforge.jp/

2. IPADIC, a dictionary for morphological analyzer. Ver.2.7.0. Information-Technology Promotion Agency, Japan. http://sourceforge.jp/projects/ipadic/

3. Bunrui Goihyo, a word list by semantic principles, revised and enlarged edition. The National Institute for Japanese Language. http://www.kokken.go.jp /en/publications/bunrui_goihyo/

4. Nihon Keizai Shimbun Newspaper Corpus, years 1990-2004, Nihon Keizai Shimbun, Inc.

References

Dagan, Ido, Lillian Lee, and Fernando Pereira. 1999. Similarity-based Models of Co-occurrence Probabilities. *Machine Learning*, 34(1-3):43–69.

Grefenstette, Gregory. 1994. Exploration in Automatic Thesaurus Discovery. Kluwer Academic Publishers. Norwell, MA.

Hagiwara, Masato, Yasuhiro Ogawa, Katsuhiko Toyama. 2006. Selection of Effective Contextual Information for Automatic Synonym Acquisition. *In Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pp.353–360.

Harris, Zelig S. 1968. Mathematical Structures of Language. Wiley, New Jersey.

Hindle, Donald. 1990. Noun Classification from Predicate-Argument Structures. *In Proceedings of the 28th Annual Meeting of the Association for Computational Linguistics*, pp.268–275.

Lee, Lillian. 1997. Similarity-Based Approaches to Natural Language Processing. Ph.D. thesis, Harvard University, Cambridge, MA.

Lee, Lillian. 1999. Measures of distributional similarity. *In Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, pp. 25–32, College Park, MD.

Lin, Dekang. 1998. Automatic Retrieval and Clustering of Similar Words. *In Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics*, pp.768–774. Montreal.

Lin, Dekang and and Patrick Pantel. 2002. Concept Discovery from Text. *In Proceedings of 19th International Conference on Computational Linguistics*, pp.577–583. Taipei.

Ruge, Gerda. 1992. Experiments of Linguistically-based Term Associations. *Information Processing & Management*, 28(3):317–332.

Shibata, Tomohide and Sadao Kurohashi. 2009. Distributional similarity calculation using very large scale Web corpus. *In Proceedings of Annual Meeting of Association for Natural Language Processing*. pp. 705–708.

Weeds, Julie and David Weir. 2005. Co-occurrence retrieval: A Flexible Framework for Lexical Distributional Similarity. *Computational Linguistics*. 31(4):439–476.

Zhitomirsky-Geffet, Maayan and Ido Dagan. 2009. Bootstrapping Distributional Feature Vector Quality. *Computational Linguistics*, 35(3):435–461.