EFFICIENCY OF VECTOR SIZE FOR WORD SENSE DISAMBIGUATION

by
Qujiang Peng, Takeshi Ito and Teiji Furugori

This paper analyzes the efficiency of vector size L for word sense disambiguation (WSD). We first obtain a set of words representing a specific topic, called L-vector, for each sense of a polysemous word w; second, we get the L-vector from a portion of the text in which w appears; and third, we measure the topical similarity between each of the L-vectors in the first step and the L-vector in the second step. After this, we select the sense given in an L-vector in the first step that got the highest similarity value with the L-vector in the second step as the meaning of w. Finally, we analyze the efficiency of vector size L for WSD. As our experiments show, the performance is strongly affected by vector size.

1. Introduction

Word sense disambiguation (WSD) has been a concern ever since the beginning days of computer treatment of natural language. The task is not an end in itself, but rather a necessary step at one level or another to have a better or more complete system for information retrieval (IR), natural language processing (NLP), and machine translation (MT) (Ide and Véronis 1998:1). Many systems that are used in word sense disambiguation are based on the notion of *vector*. However, an important research area that has not been given enough attention is a formal analysis of the vector size affecting performance.

In this paper we analyze the efficiency of vector size L, using a WSD method based on vectors. We first obtain an L-vector (a set of words representing a specific topic) for each sense of a polysemous word w. Next, we get the L-vector from a portion of the text in which w appears, then describe a process of resolving lexical ambiguities by measuring topical similarities between the L-vector of each sense of w and the L-vector of the context in which the w appears that has to be disambiguated. Finally, we show that the performance is strongly affected by vector size.

2. Background

Language has ambiguities. They appear in all levels of its phonological, morphological, syntactic, semantic, and pragmatic dimensions. Lexical ambiguity, the feature of potential multiplicity in meanings for any word appearing in sentences, is one of them, and it is a crucial problem to be solved in many IR, NLP, or MT systems.

History tells us that early work in word sense disambiguation took a pure AI approach (Wilks 1968:59; Small and Rieger 1982; Hirst 1987). The early researchers manually set up a knowledge base for each word and described its senses in various linguistic usages. After a period at this stage, some started to use machine-readable dictionaries (MRDs) to select the proper sense of a polysemous word (Lesk 1986:24; Walker 1987; Guthrie, Guthrie, and Aidinejad 1991:146). The majority of research being done nowadays in word sense disambiguation uses occurrence information of words taken from corpora, and employs statistical means to determine the meaning of words in sentences (Yarowsky 1992:454; Dagan and Itai 1994:563; Karov and Edelman 1998:41).

A more comprehensive review of WSD is beyond the scope of this paper, but may be found in Ide and Véronis (1998:1). Many of the methods used in WSD shared a common vector representation. They used different means to create a vector for the document of each sense and test text. The meaning of the polysemous word was determined by calculating the similarity of vectors with the maximum vector size $L_{\max}$ (the total number of elements in the vector). This is a basic procedure for word sense disambiguation. However, an important area of research that has not been given enough attention is a formal analysis of the vector size affecting performance. Intuitively, even if the method is excellent, noise will still exist in the vector. The best sized vector should contain enough disambiguating information and relatively little noise. In this paper, we propose a formal analysis of vector size, using a WSD method based on vectors. Experiments show that our intuition is right.

3. Vector-based Word Sense Disambiguation

In our system, L-vectors are constructed using a corpus-based semantic network (CSN). First, we build our CSN from a corpus and thus it is 'colored' by the domain the corpus deals with.

3.1. Construction of a Corpus-based Semantic Network

We have built our network from the EDR corpus (EDR 2002). To cope with the data sparseness problem, we build the network with all the nouns, verbs and adjectives whose occurrences in the corpus are bigger than a certain number. We use the number 60 in this paper: the inclusion of less frequently used words in a CSN could introduce the data sparseness problem for the polysemous words to be disambiguated. Our CSN contains 1,845 nodes (words). We make each node (word) in CSN to have a hundred links to other nodes (words) labeled with the first to the hundredth higher values of strength of semantic association. Figure 1 shows a portion of the network, where the natural number i indicates the ith word ($1 \le i \le 100$) that has ith highest semantic association to the word *disease*; the real number shows the value of strength of semantic association between the word *disease* and the word in the ith node.

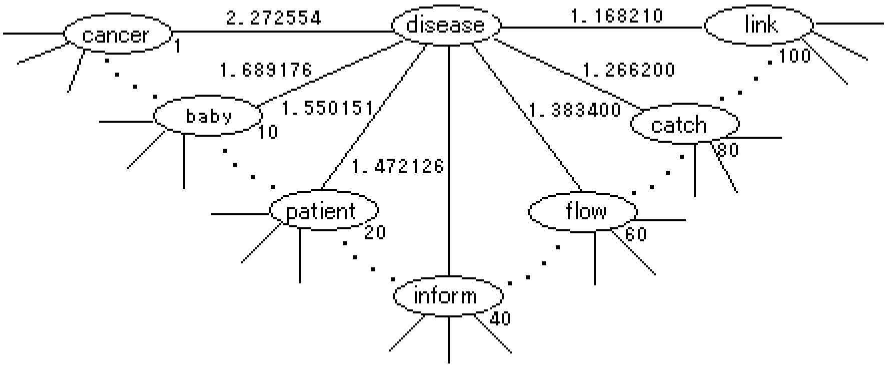

Figure 1. A portion of semantic network

The link from node n_i to n_j in our CSN is labeled with a weighted value given by the mutual information $I(n_i, n_j)$ (Church and Hanks 1990:22) between the words n_i and n_j in the EDR corpus.

3.2. Process of Word Sense Disambiguation

The process of word sense disambiguation starts with collecting from the textual data on the training corpus the sentences that contain the polysemous word to be disambiguated. We get c_m instances for each sense of the polysemous word.

Procedure. Using the CSN, we attempt to arrive at the meaning of the polysemous word in the following procedure:

(a) Activate the CSN using the words in the set of c_m instances and get the L-vector for each sense of a polysemous word w.
(b) Activate the CSN using the words in a portion of the text in which w appears and get the L-vector and its corresponding L-vector value.
(c) Calculate the similarity between each of the L-vectors in (a) and the L-vector in (b).
(d) Select the sense in (a) that obtained the highest similarity value as the meaning of w in the text.

Calculation of the Strength of Association. We use the location l to express the location of the lth word, w_l, of the text. If the text has t_s words, then $1 \le l \le t_s$. We activate the CSN when a word w_l in an instance or in the portion of the text containing the polysemous word w matches the word for node n_k in the network ($w_l = n_k$). We use $a_i(l)$ to express the strength of association between the node n_i $(i = 1, 2, \cdots, 1845)$ and the context $w_1, w_2, \cdots, w_l$. We calculate the strength of association using the following equation. Here, the initial value of $a_i(0)$ is 0.

$$a_i(l) = \begin{cases} a_i(l-1) + I(n_i, n_k) & \textit{if } n_i \textit{ and } n_k \textit{ have a link in the CSN} \\ a_i(l-1) & \textit{otherwise} \end{cases}$$

L-vector and L-vector value. Activate the CSN using the words $w_1, w_2, \cdots, w_l$ and produce an activation state, $S(l)$.

$$S(l) = (a_1(l), a_2(l), \cdots, a_{1845}(l))$$

$S(l)$ gives an influence to $S(l+1)$, and it in turn to $S(l+2)$, and so on. The vector N of the nodes corresponding to $S(l)$ is:

$$N = (n_1, n_2, \cdots, n_{1845})$$

The mutual information estimates the strength of association between two words. We activate the CSN by using the context $w_1, w_2, \cdots, w_l, \cdots, w_{t_s}$ to get the strength of association $a_i(t_s)$ of the node n_i. $a_i(t_s)$ shows the strength of association between the node n_i and the topic represented in the context $w_1, w_2, \cdots, w_l, \cdots, w_{t_s}$. The bigger $a_i(t_s)$ is, the more relevant the corresponding node n_i is to the topic represented in the context $w_1, w_2, \cdots, w_l, \cdots, w_{t_s}$.

We get a state $S'(t_s)$ of association values by arranging the elements in $S(t_s)$ in decreasing order.

$$S'(t_s) = (a'_1(t_s), a'_2(t_s), \cdots, a'_L(t_s), \cdots, a'_{1845}(t_s))$$

We see that in $S'(t_s)$ $a'_1(t_s) \geq a'_2(t_s) \geq \cdots \geq a'_L(t_s) \geq \cdots \geq a'_{1845}(t_s)$. The node vector N' of corresponding nodes to $S'(t_s)$ is:

$$N' = (n'_1, n'_2, \cdots, n'_L, \cdots, n'_{1845})$$

The relevant nodes expressing the topic represented in the context of w come in the front part of N' as the number of activations of the

CSN increases. We call this vector of size L the L-vector, N_L, and its corresponding association value the L-vector value, V_L.

$$N_L = (n'_1, n'_2, \cdots, n'_L)$$
$$V_L = \phi(a'_1(t_s), a'_2(t_s), \cdots, a'_L(t_s)) = (b_1, b_2, \cdots, b_L)$$

Here, $1 \le L \le 1845$ and ϕ is the normalization factor that restricts the value of b_i to $[0,\ 1]$.

Determination of the meaning. Let s_m be the mth sense of a polysemous word w. Using the c_m instances from the training corpus, we get the L-vector N_L of s_m:

$$N_L(s_m) = (x_1, x_2, \cdots, x_j, \cdots, x_L)$$

Similarly, we get the L-vector N_L and the L-vector value V_L from the context representation (CR) of a polysemous word w in the test text:

$$N_L(CR) = (y_1, y_2, \cdots, y_j, \cdots, y_L)$$
$$V_L(CR) = (z_1, z_2, \cdots, z_j, \cdots, z_L)$$

Miller and Charles (1991:1) found evidence in several experiments that humans determine the semantic similarity of words from the similarity of the contexts the words are used in. Karov and Edelman (1998:41), in their study of WSD, used the idea that words are considered similar if they appear in similar contexts and contexts are similar if they contain similar words. Extending this finding, Schütze (1998:97) hypothesized that the same holds for ambiguous word senses: a sense is interpreted as a group of similar contexts that are about the same topic.

With this in mind, let us see how topical similarity is measured. If the sense of w in the context representation (CR) is s_1, then CR and the instances of s_m $(m \ne 1)$ from the training corpus are not considered to be similar contexts. The nodes coming to the front part of $N'(CR)$ are relevant to expressing the topic represented in CR. The nodes in the front part of $N'(s_m)$ $(m \neq 1)$ relate closely to the topic in the context of s_m $(m \neq 1)$. CR and the context of s_m $(m \neq 1)$ have different topics since CR and the context of s_m $(m \neq 1)$ are not similar contexts. Node-wise, this is to say that the front parts of $N'(CR)$ and $N'(s_m)$ $(m \neq 1)$ do not have many words in common when they are in different topics, but have many words in common when they are in a similar topic.

From these, we calculate the similarity $Sim(CR, s_m)$ in the equation:

$$Sim(CR, s_m) = \sum_{i=1}^{L} \sum_{j=1}^{L} z_j \times A(x_i, y_j)$$

Here,

$$A(x_i, y_j) = \begin{cases} 1 & if \quad x_i = y_j \\ 0 & otherwise \end{cases}$$

We first get the set of common words, $N_L(s_m) \cap N_L(CR)$, and then calculate its ratio in $N_L(CR)$ by the L-vector value $V_L(CR)$. If we do not care whether the best vector size exists or not, we can directly let L be the maximum vector size $L_{\max}$ ($L_{\max} = 1845$).

4. Experiments and Results

We first show the working of the disambiguation process in an example. The process of getting the meaning of w is:

(a) Obtain the $N_L(s_1)$, $N_L(s_2), \cdots, N_L(s_r)$ for the lexical meaning of w.

(b) Get $V_L(CR)$ and $N_L(CR)$ using the words in a portion of the test text in which w appears.

(c) Calculate the similarity $Sim(CR, s_1)$, $Sim(CR, s_2), \cdots, Sim(CR, s_r)$.

(d) Select the s_m with the highest similarity value as the meaning of w.

4.1. Illustrated Example

Suppose the word to be disambiguated is *cell* in the following text:

CR = … *a* convicted rapist-murderer, escaped from the state prison early Monday. The inmates were reported missing from a maximum-security wing *at* the Oklahoma State Penitentiary about 5 a.m. said Jerry Massie, a spokesman for the State Department of Corrections. The *pair's* escape went unnoticed until they were discovered missing from their **cell**s. Massie identified the escapees as James Robert Thomas, 25, *and* Willie Lee Hoffman, 21. Thomas was sentenced to life without parole for first-degree murder in November 1997. He received *an* additional 400 years in prison for rape, Massie said. Hoffman is serving a 20-year sentence for kidnapping and other *charges* … (from a text on the Internet)

The first word of CR is *a*, the second *convict*, and so on. *S(l)* is the state of association values of the *l* th word of CR: *S(1) = S(a)*; *S(20) = S(at)*; *S(40) = S(pair)*; *S(60) = S(and)*; *S(80) = S(an)*; *S(100) = S(charge)*. *S(0)* is the initial state.

Using the method presented earlier, we get the following states of association values:

$$\begin{array}{llcccccccccl}
Nodes & = (& n_1 & n_2 & \cdots & n_{768} & n_{769} & \cdots & n_{1241} & n_{1242} & \cdots & n_{1844} & n_{1845} &) \\
 & = (& abandon & ability & \cdots & guilty & guy & \cdots & prison & prisoner & \cdots & young & youth &) \\
 & & & & & & & & & & & & & \\
S(0) & = (& 0.000000 & 0.000000 & \cdots & 0.000000 & 0.000000 & \cdots & 0.000000 & 0.000000 & \cdots & 0.000000 & 0.000000 &) \\
S(1) & = (& 0.000000 & 0.000000 & \cdots & 0.000000 & 0.000000 & \cdots & 0.000000 & 0.000000 & \cdots & 0.000000 & 0.000000 &) \\
\vdots & & & & & & & & & & & & & \\
S(20) & = (& 2.383491 & 0.000000 & \cdots & 2.845997 & 0.000000 & \cdots & 2.403992 & 4.109032 & \cdots & 0.000000 & 2.565026 &) \\
\vdots & & & & & & & & & & & & & \\
S(40) & = (& 3.035578 & 0.000000 & \cdots & 3.834162 & 0.000000 & \cdots & 5.027419 & 4.776267 & \cdots & 0.000000 & 3.757311 &) \\
\vdots & & & & & & & & & & & & & \\
S(60) & = (& 6.016582 & 0.000000 & \cdots & 3.834162 & 0.740733 & \cdots & 7.512235 & 8.962006 & \cdots & 0.000000 & 7.849979 &) \\
\vdots & & & & & & & & & & & & & \\
S(80) & = (& 7.475369 & 0.000000 & \cdots & 7.286366 & 0.740733 & \cdots & 11.153038 & 10.749029 & \cdots & 0.888497 & 11.621449 &) \\
\vdots & & & & & & & & & & & & & \\
S(100) & = (& 8.127456 & 0.000000 & \cdots & 12.582113 & 0.740733 & \cdots & 17.004826 & 13.203289 & \cdots & 0.888497 & 14.087394 &)
\end{array}$$

Following this calculation, we get $S'(100)$ by arranging the elements in $S(100)$ in decreasing order and N' for $S'(100)$.

$S'(100) = ($ *17.004826, 15.019628, 14.347972, 14.087394, 13.203289, 12.582113, 12.160870, 11.736558, 11.106226, 11.095078, …*)
$N' =$ (*prison, prosecutor, sentence, youth, prisoner, guilty, wife, commit, murder, governor, …*)

The relevant words in the semantic network to express the topic represented in the context of w come in the front part of N'. For $L = 100$, we get the L-vector:

$N_L(CR) = ($ *prison, prosecutor, sentence, youth, prisoner, guilty, wife, commit, murder, governor, …*)

Using the normalization factor ϕ, we get the L-vector value:

$V_L(CR) = ($*0.020933, 0.018489, 0.017662, 0.017342, 0.016253, 0.015489, 0.014970, 0.014448, 0.013672, 0.013658, …*)

The definition of sense is a notoriously subjective and debatable subject area. Here, we only show that the working principle of our method is based on L-vectors and therefore we will not discuss how the sense distinctions were decided on. *Cell* is given three nominal meanings s_1, s_2, and s_3 in our experiment: *the smallest living unit*, *battery*, and *a room for (a) prisoner*. The L-vectors for them are:

$N_L(s_1) = ($*researcher, cancer, treatment, transplant, virus, connect, cell, …*)
$N_L(s_2) = ($*cell, electric, gas, technology, energy, robot, pursue, requirement, …*)
$N_L(s_3) = ($*sentence, prison, prosecutor, prisoner, commit, criminal, trial, …*)

The similarities $Sim(CR, s_1)$, $Sim(CR, s_2)$, and $Sim(CR, s_3)$ are 0.114585, 0.045904, and 0.612215. So we get the meaning of *cell* to be s_3 (*a room for (a) prisoner*).

4.2. Results

The method proposed here is very simple. We can use it to check the effect of the performance by vector size. We call $Suc(L)$ the success rate when testing a lot of data using the vector size L. $Suc(L) - Suc(L_{\max})$ shows the improvement in that it indicates the difference between two numbers, $Suc(L)$ and $Suc(L_{\max})$. In order to express the precision of the difference, we can define the improvement rate as:

$$IMP(L) = \frac{Suc(L) - Suc(L_{\max})}{Suc(L_{\max})} \times 100\% .$$

It is very difficult to compare the performance of the different WSD methods. Kilgarriff and Rosenzweig (2000) proposed a first evaluation exercise in their SENSEVAL project. We use the first 12 instances ($c_m = 12$) of the mth sense of a polysemous word (noun) in the TRAIN data of SENSEVAL-1 to obtain $N_L(s_m)$. If s_m has training data, but less than 12 instances in TRAIN ($0 < c_m < 12$), we use the c_m instances to obtain $N_L(s_m)$. If s_m has no training data ($c_m = 0$) in TRAIN, we use the definition texts and examples in DICT data of SENSEVAL-1 to obtain $N_L(s_m)$. We use TEST data of SENSEVAL-1 as test data.

The results (success rates) we got from the experiment are 41.7%, 49.3%, 53.1%, 54.1%, 54.0%, 54.3%, 54.0%, 55.7%, 56.1%, 59.0%, 56.1%, 53.1%, and 51.3% for $L = 25$, 50, 100, 200, 300, 400, 600, 800, 1000, 1200, 1400, 1600, and 1845, respectively. The best one is 59.0% for $L = 1200$. The improvement rate $IMP(1200)$ is 15.0%.

Figure 2 shows the results for the various values of L from 25 to 1845.

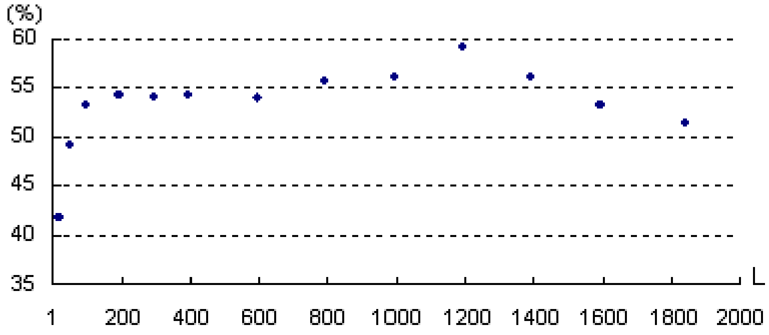

Figure 2. Result of the disambiguation experiment

Figure 3 shows the experimental results for L between 1100 and 1200.

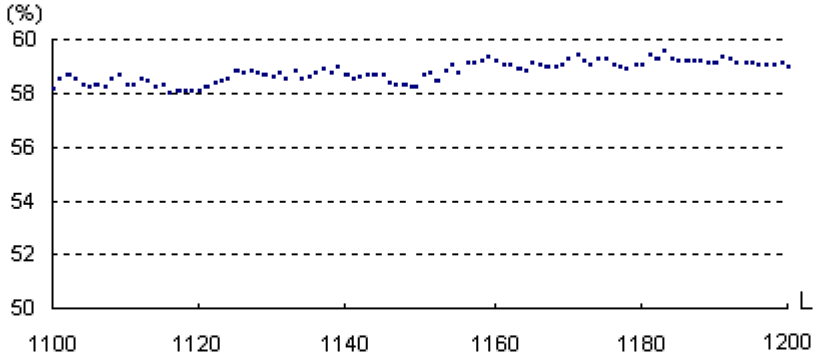

Figure 3. Stability of the disambiguation results

4.3. Evaluation

If vector size is different, then the result is different. If the vector sizes are close to each other then the difference of results is relatively small, because the difference among the words used between $N_{L_1}(s_m) \cap N_{L_1}(CR)$ and $N_{L_2}(s_m) \cap N_{L_2}(CR)$ $(L_1 \approx L_2)$ is small. There is an optimal vector size.

The disambiguation result is not very good when L is either too small or too big. Naturally, the disambiguating information needed to express the topic involved is not sufficient when L is too small, and the L-vectors contain too much noise when L is too big. It is easy to see that $N_L(s_m) \cap N_L(CR)$ contains few words when L is too small. For every sense s_m, $N_L(s_m) \cap N_L(CR)$ would probably be an empty set when $L = 1$. Likewise, $N_L(s_m) \cap N_L(CR)$ would contain too many 'noise words' when L is too big. The difference between $N_L(s_m) \cap N_L(CR)$ and $N_L(s_i) \cap N_L(CR)\,(i \neq m)$ as to words used becomes smaller as L increases.

The results in Figure 3 are relatively stable and they vary within 1.5%. The L-vectors contain enough disambiguating information and relatively little noise information.

In vectors, not all the words are useful for disambiguation. When capturing the disambiguating information, some noise will also be captured. Even if the method is perfected, noise will still exist in the vector. Therefore, the ideas presented in this paper will be useful also for other research on WSD methods that are based on vectors. For example, Lafourcade (2001) proposed a word sense disambiguation method based on conceptual vectors. A conceptual vector was constructed using 873 words (headwords). The similarity of two conceptual vectors was calculated by an arcosine function (more detailed information may be found at his homepage; Lafourcade 2004). Here, using the 1845 nodes of the CSN as headwords, we constructed the conceptual vectors. Using the same SENSEVAL-1 as training and test data, we tested Lafourcade's method. The results we got from the experiment were 18.9%, 19.1%, 19.0%, 18.2%, 18.0%, 18.1%, 18.1%, 17.5%, 17.0%, 19.0%, 20.1%, 18.9%, 17.4%, and 17.1% for $L = 25$, 50, 100, 200, 300, 400, 600, 800, 1000, 1200, 1400, 1600, 1800, and 1845, respectively. The best result was 20.1% for $L = 1400$. For capturing the information about disambiguation, Lafourcade's method may not be ideal. However, the improvement rate $IMP(1400)$ is 17.5%.

5. Conclusion

We have analyzed the efficiency of vector size L, using a word sense disambiguation method based on vectors. Even though the L-vectors contain enough disambiguating information and relatively little noise information, some noise will still be captured, when we try to capture the information that is essential for disambiguating purposes.

Intuitively, even if the method is excellent, it is difficult to eliminate the noise completely. The best vector size should contain enough disambiguating information and relatively little noise. Experiments show that our intuition is right. The optimal vector size is not the maximum vector size $L_{\max}$. Therefore, the efficiency of vector size will be useful for other research on WSD methods that are based on vectors.

The purpose of this paper was to analyze the efficiency of vector size for WSD. But our method may be combined with other sense disambiguation techniques to improve the performance of sense disambiguation. For example, we may use syntactic clues, such as syntactic relations between two words (e.g. subject-verb, verb-object) as done by Dagan and Itai (1994:563) as supplement indicators for WSD. This will be especially effective in verb sense disambiguation.

Department of Computer Science
The University of Electro-Communications
1-5-1, Chofugaoka
Chofu-shi
Tokyo 182-8585
Japan

References

Church, Kenneth W. & Patrick Hanks. 1990. Word association norms, mutual information and lexicography. Computational Linguistics 16.22-29.

Dagan, Ido & Alon Itai. 1994. Word sense disambiguation using a second language monolingual corpus. Computational Linguistics 20 (4).563-595.

EDR. 2002. http://www2.crl.go.jp/kk/e416/EDR/index.html

Guthrie, Joe A. et al. 1991. Subject-dependent co-occurrence and word sense disambiguation. Proceedings of the 29th Meeting of ACL, Berkeley: Morgan Kaufmann, 146-152.

Hirst, Graeme. 1987. Semantic interpretation and the resolution of ambiguity. Cambridge: Cambridge University Press.

Ide, Nancy & Jean Véronis. 1998. Introduction to the special issue on word sense disambiguation: The state of the art. Computational Linguistics 24(1).1-40.

Karov, Yael & Shimon Edelman. 1998. Similarity-based word sense disambiguation. Computational Linguistics 24 (1).41-59.

Kilgarriff, Adam & Joseph Rosenzweig. 2000. English SENSEVAL: report and results. Proceedings of 2nd International Conference on Language Resources & Evaluation (LREC 2000), Athens.

Lafourcade, Mathieu. 2001. Lexical sorting and lexical transfer by conceptual vectors. Proceedings of the First International Workshop on MultiMedia Annotation (MMA'2001), Tokyo, Japan, 2001.

Lafourcade, Mathieu. 2004. http://www.lirmm.fr/~lafourcade/ML-biblio/lafourcade-publications.html

Lesk, Michael. 1986. Automatic sense disambiguation: how to tell a pine cone from an ice cream cone. Proceedings of the 1986 SIGDOC Conference, New York: Association for Computing Machinery, 24-26.

Miller, George A. & Walter G. Charles. 1991. Contextual correlates of semantic similarity. Language and Cognitive Processes 6 (1).1-28.

Schütze, Hinrich. 1998. Automatic word sense discrimination. Computational Linguistics 24 (1).97-123.

Small, Steve & Chuck Rieger. 1982. Parsing and comprehending with word experts (a theory and its realization). In: W. Lehnert & M. Ringle (eds.), Strategies for natural language processing. Hillsdale, N.J.: Lawrence Erlbaum.

Walker, Donald E. 1987. Knowledge resource tools for accessing large text files. In: S. Nirembe (ed.), Machine Translation: Theoretical and Methodological issues. Cambridge: Cambridge University Press.

Wilks, Yorick. 1968. On-line semantic analysis of English texts. Mechanical Translation 11(3-4).59-72.

Yarowsky, David. 1992. Word-sense disambiguation using statistical models of Roget's categories trained on large corpora. Proceedings of COLING-92. Nantes: ICCL, 454-460.