Tejada-Cárcamo J, Calvo H, Gelbukh A *et al.* Unsupervised WSD by finding the predominant sense using context as a dynamic thesaurus. JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY 25(5): 1030–1039 Sept. 2010. DOI 10.1007/s11390-010-1081-8

Unsupervised WSD by Finding the Predominant Sense Using Context as a Dynamic Thesaurus

Javier Tejada-Cárcamo[1], Hiram Calvo[2,3], Alexander Gelbukh[2], and Kazuo Hara[3]

[1] *San Pablo Catholic University, Arequipa, Peru*

[2] *Center for Computing Research, National Polytechnic Institute, Mexico City, 07738, Mexico*

[3] *Nara Institute of Science and Technology, Takayama, Ikoma, Nara 630-0192, Japan*

E-mail: jawitejada@hotmail.com; hcalvo@cic.ipn.mx; calvo@is.naist.jp; gelbukh@gelbukh.com; kazuo-h@is.naist.jp

Received June 12, 2009; revised June 23, 2010.

Abstract We present and analyze an unsupervised method for Word Sense Disambiguation (WSD). Our work is based on the method presented by McCarthy *et al.* in 2004 for finding the predominant sense of each word in the entire corpus. Their maximization algorithm allows weighted terms (similar words) from a distributional thesaurus to accumulate a score for each ambiguous word sense, i.e., the sense with the highest score is chosen based on votes from a weighted list of terms related to the ambiguous word. This list is obtained using the distributional similarity method proposed by Lin Dekang to obtain a thesaurus. In the method of McCarthy *et al.*, every occurrence of the ambiguous word uses the same thesaurus, regardless of the context where the ambiguous word occurs. Our method accounts for the context of a word when determining the sense of an ambiguous word by building the list of distributed similar words based on the syntactic context of the ambiguous word. We obtain a top precision of 77.54% of accuracy versus 67.10% of the original method tested on SemCor. We also analyze the effect of the number of weighted terms in the tasks of finding the Most Frecuent Sense (MFS) and WSD, and experiment with several corpora for building the Word Space Model.

Keywords word sense disambiguation, word space model, semantic similarity, text corpus, thesaurus

1 Introduction

Word Sense Disambiguation (WSD) consists of determining the sense expressed by an ambiguous word in a specific context. This task has a particular importance in document analysis[1] because the user may be selecting a particular set of documents based on the sense of word being used[2]. When building multilingual querying systems, for example, the right translation of a particular word must be chosen in order to retrieve the right set of documents.

The task of WSD can be addressed mainly in two ways: 1) supervised: applying techniques of machine-learning trained on previously hand-tagged documents and 2) unsupervised: learning directly from raw words grouping automatically clues that lend to a specific sense according to the hypothesis that different words have similar meanings if they are presented in similar contexts[3-4]. To measure the effectiveness of the state-of-the-art methods for WSD, there is a recurrent event called Senseval[5].

For instance, the results of Senseval-2 English all-words task are presented in Table 1. This task consists of 5000 words of running text from three Penn Treebank and Wall Street Journal articles. The total number of words to be disambiguated is 2473. Sense tags are assigned using WordNet 1.7. The last column in Table 1 shows whether a particular system uses manually tagged data for learning or not. The best systems

Table 1. The Top-10 Systems of Senseval-2

Rank	Precision	Recall	Attempted	System
1	0.690	0.690	100.000	SMUaw
2	0.636	0.636	100.000	CNTS-Antwerp
3	0.618	0.618	100.000	Sinequa-LIA - HMM
4	0.605	0.605	100.000	MFS
5	0.575	0.569	98.908	UNED-AW-U2
6	0.556	0.550	98.908	UNED-AW-U
7	0.475	0.454	95.552	UCLA-gchao2
8	0.474	0.453	95.552	UCLA-gchao3
9	0.416	0.451	108.500	CL Research-DIMAP
10	0.451	0.451	100.000	CL Research-DIMAP (R)
11	0.500	0.449	89.729	UCLA-gchao

Regular Paper

Supported by the Mexican Government (SNI, SIP-IPN, COFAA-IPN, and PIFI-IPN), CONACYT and the Japanese Government.

are those which learn from previously manually tagged data, however this resource is not always available for every language, and it can be a costly resource to build. Because of this, we will focus on unsupervised systems, such as UNED-AW-U2 (Rank 4 in Table 1).

Choosing always the most frequent sense for each word yields a precision and recall of 0.605. Comparing this with the results in Table 1 shows that finding the Most Frequent Sense (MFS) can be a good strategy, as the baseline of 60% would be ranked amongst the first 4 systems. We verified this by obtaining the MFS from WordNet. The senses in WordNet are ordered according to the frequency data collected from the manually tagged resource SemCor[6]. Senses that have not occurred in SemCor are ordered arbitrarily.

Diana McCarthy *et al.* propose in [3] an algorithm to find the prevalent (most frequent) sense for each word. They first build a thesaurus using Lin's method[4] and data from a corpus such as the BNC (British National Corpus). Then each ranked term k in the thesaurus votes for a certain sense ws_i of the word. The vote uses the distributional similarity score which is then weighted by the normalized similarity between the sense and sense of k (ks_j) which maximizes the score. For detailed information about sense voting algorithm, see Subsection 2.3.2.

Fig.1 illustrates this concept. For example, in "*I have donated my blow to the Peruvian blow bank*", the top 5 *weighted terms* (similar words) from the Lin's thesaurus for the word *bank* are: *commercial bank, company, financial institution, firm, corporation*. In this example, the *weighted terms* reflect poorly the sense of *blow bank* because they are biased towards the most frequent sense (*financial institution*), thus, effectively finding the MFS for all instances of each word.

The thesaurus is considering all contexts where the word *star* has appeared previously, when building the thesaurus; however, it is not considering the context of the current sentence which includes key words for finding the correct sense for this instance. Motivated by this, we decided to perform Word Sense Disambiguation

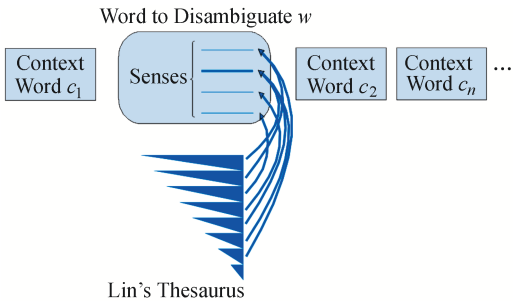


Fig.1. Finding the predominant sense using a static thesaurus as in McCarthy *et al.*

based on improving the finding of the MFS by considering this local context.

In our method, we obtain a list of synonyms or related words (*quasi-synonyms*) for each ambiguous word. These weighted terms will determine the sense for a word using the maximization algorithm presented in [3]. This algorithm allows each quasi-synonym to accumulate a score for each sense of the ambiguous word, so that the sense which has the highest score is chosen.

The main contribution of our method is the algorithm of obtaining quasi-synonyms. For this purpose we collect all the contexts in a corpus where a specific word is present, and then we use this information to build a word space model where it is possible to measure the similarity between words of the training corpus. The weighted terms of an ambiguous word are those which are the closest by their contexts.

Weighted terms of any word change dynamically depending on their local contexts and the corpus. For example, in "*The doctor cured my wounds with a medicine*", the weighted terms for *doctor* would be: *physician, medicine, alcohol, lint*; however, in "*The doctor published his latest research in the conference*", the weighted terms of *doctor* would be *scientific, academic, university, conference*.

The method proposed by McCarthy *et al.* does not consider the local context of the word to be disambiguated. The sense chosen as predominant for the word to disambiguate depends solely on the corpus used to build the thesaurus. We propose considering context words to dynamically build a thesaurus of words related to the word to be disambiguated. This thesaurus is dynamically built based on a Dependency Co-Occurrence

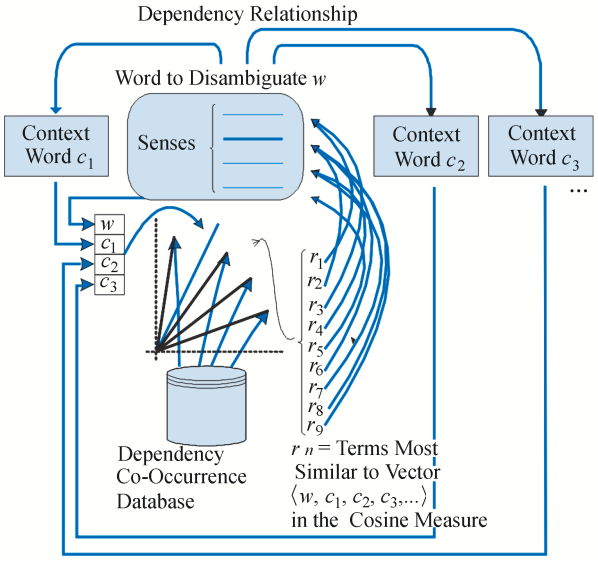


Fig.2. Our proposal: create dynamic thesauri based on the dependency context of the ambiguous word.

Data Base (DCODB) previously collected from a corpus. Each *co-occurrent-with-context* word will vote — as in the original method — for each sense of the ambiguous word, finding the predominant sense for this word, but in a particular context[7]. See Fig.2.

In Subsection 2.1 we explain how the Dependency Co-Occurrence Data Base (DCODB) resource is built; then we explain our way of measuring the relevance of co-occurrences based on Information Theory in Subsection 2.2. In Subsections 2.3 and 2.4, we explain relevant details of our method. In Subsection 3.1, we present a simple comparison with the original method. Then, in Subsection 3.2 we experiment with different corpora for building the DCODB — and therefore the WSM and the dynamic thesaurus. In Subsection 3.3 we evaluate the impact of the *weighting terms* for finding the MFS (Subsection 3.3.1) and for WSD (Subsection 3.3.2), and finally we draw our conclusions.

2 Methodology

2.1 Building the Dependency Co-Occurrence Data Base (DCODB)

We obtain dependency relationships automatically using the MINIPAR parser. MINIPAR has been evaluated with the SUSANNE corpus, a subset of the Brown Corpus, and it is able to recognize 88% of the dependency relationships with an accuracy of 80%[8]. Dependency relationships are asymmetric binary relationships between a *head* word and a *modifier* word. A sentence builds up a tree which connects all words in it. Each word can have several modifiers, but each modifier can modify only one word[9-10].

We apply three simple heuristics for extracting *head-governor* pairs of dependencies:

1) Ignore Prepositions — see Fig.3.

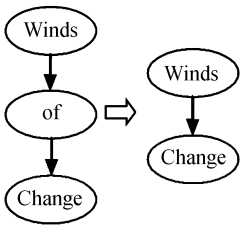


Fig.3. Ignoring prepositions.

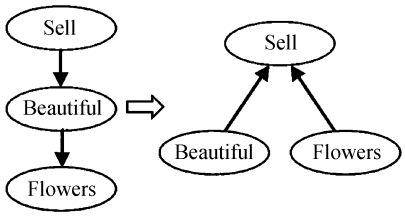


Fig.4. Including sub-modifiers as modifiers of the head.

2) Include sub-modifiers as modifiers of the head — see Fig.4.

3) Separate heads lexically identical, but with different part of speech. This helps to keep context separated.

2.2 Construction of the Word Space Model (WSM)

From the DCODB we build a word space model. We use TF·IDF (term frequency · inverse document frequency)[7] for weighting. The TF·IDF WSM model is usually used for classification tasks and for measuring document similarity. Each document is represented by a vector whose number of dimensions is equal to the quantity of different words that are in all documents. Words not present in a particular document are filled with zero values. In our method, a *head* is analogous to a document, and its modifiers are dimensions of the vector which represents it. The value for each dimension is a weight that reflects the number of co-occurrences between a modifier and a head. Thus, a vector is represented as:

$$\boldsymbol{Vector}(head_n) = \{(mod_1, w_1), (mod_2, w_2), \ldots, (mod_n, w_n)\}$$

where: $head_n$ is the head word, mod_n is the name of the modifier word, w_n is the weight represented by the normalized number of co-occurrences between mod_n and $head_n$.

The weight of co-occurrence is the dot product of the normalized frequency of a head (TF) and its inverse frequency (IDF). TF shows the importance of a modifier with regard to the modified head, so that the weight of the relationship increases when the modifier appears more frequently with such *head*. TF is calculated with the following formula:

$$f_{i,j} = \frac{freq_{i,j}}{\max(freq_{l,j})}$$

where $freq_{i,j}$ is the frequency of the modifier i with $head_j$, and $\max(freq_{l,j})$ is the highest frequency number of the modifiers of $head_j$.

IDF shows the relevance of a modifier with regard to the remaining heads in the database (DCODB), in a way that the weight of a modifier decreases if it appears more often with every other head in the DCODB; while it increases when it appears with a less number of heads. This means that highly frequent modifiers help little to discriminate when the head is represented by a vector. IDF is calculated with the equation:

$$idf_i = \log \frac{N}{n_i}$$

where N is the total number of heads, and n_i is the total number of heads which co-occur with modifier i.

2.3 Disambiguating Process

Once the database has been built, we are able to begin the disambiguation process for a given word w in a context C (made up of words $C = \{c_1, c_2, \ldots, c_n\}$). The first step of this process consists of obtaining a weighted list of terms related with w. The second step consists of using these terms to choose a sense of w continuing with the original algorithm proposed by McCarthy *et al.*[3] The following subsections explain these steps in detail.

2.3.1 *Obtaining the Weighted List of Terms Related with w*

A word is related with another one when they are used in similar contexts. In our method this context is defined by syntactic dependencies. See Fig.2. Given an ambiguous word, w, its dependencies $c_1, c_2, c_3, \ldots$, form a vector $\boldsymbol{w} = \langle w, c_1, c_2, c_3, \ldots, w_j, \ldots\rangle$, which is compared with all vectors $\boldsymbol{r}_i = \langle r_{i,1}, r_{i,2}, \ldots, r_{i,j}, \ldots\rangle$ from the WSM using the cosine measure function:

$$\cos\_\text{measure}(\boldsymbol{w}, \boldsymbol{r}_i) = \frac{\boldsymbol{w} \cdot \boldsymbol{r}_i}{|\boldsymbol{w}| \times |\boldsymbol{r}_i|} = \frac{\sum_{j=1}^{n} w_j \times r_{i,j}}{\sqrt{\sum_{j=1}^{n}(w_j)^2} \times \sqrt{\sum_{j=1}^{n}(r_{i,j})^2}}.$$

The value obtained is used as a similarity weight for creating the weighted list of related terms. Note that this comparison is subject to the data sparseness problem because the number of modifiers of an ambiguous word is usually between 0 and 5 — considering only one level of the syntactic tree — whereas the dimension of most vectors in the WSM are far higher. To be able to compare both vectors, the remaining dimensions for the ambiguous word with its context are filled with zeroes as in values for $o_1, o_2, o_3, \ldots, o_n$ in Table 2. Also see Table 2 for an example of calculation of the cosine measure. Given the vector $\boldsymbol{w}$ formed by the word w and its context words (based on dependency relationships from the sentence where w is found), the WSM is queried with all the r_n words to compare with each vector $\boldsymbol{r}$. For example, the cosine measure between $\boldsymbol{w}$ and $\boldsymbol{r}_3$ is given by:

$$\cos\_\text{measure}(\boldsymbol{w}, \boldsymbol{r}_3) = \big[(1\cdot 4) + (1\cdot 3) + (1\cdot 1) + (1\cdot 0) + (0\cdot 0) + (0\cdot 0) + (0\cdot 4) + (0\cdot 4)\big] \big/ \big(\sqrt{1^2+1^2+1^2+1^2+0^2+0^2+0^2+0^2} + \sqrt{4^2+3^2+1^2+0^2+0^2+0^2+4^2+4^2}\big).$$

Table 2. Fragment of the WSM Showing Cosine Measure Calculation①

	c_1	c_2	c_3	$\cdots$	c_n	o_1	o_2	o_3	$\cdots$	o_m	cos
$\boldsymbol{w}$	**1**	**1**	**1**	$\cdots$	**1**	**0**	**0**	**0**	$\cdots$	**0**	1.00
r_1	1	6	2	$\cdots$	0	0	0	3	$\cdots$	0	0.99
r_2	0	4	1	$\cdots$	3	4	1	1	$\cdots$	0	0.93
r_3	4	3	1	$\cdots$	0	0	0	4	$\cdots$	4	0.83
$\ldots$				$\ldots$					$\ldots$		
r_{13}	0	0	2	$\cdots$	4	0	0	1	$\cdots$	5	0.68
$\ldots$				$\ldots$					$\ldots$		

2.3.2 *Sense Voting Algorithm*

Here we describe our modifications to the sense voting algorithm proposed by McCarthy *et al.*[3]. This algorithm allows each member of the list of related terms (dynamic — in our proposal, or static — in the original form — thesaurus) to contribute for a particular sense of the ambiguous word w. The weight of the term in the list is multiplied by the semantic similarity — see previous section — between each of the senses of a term $r_i s_j$ and the senses of the ambiguous word ws_k. The highest value of semantic distance determines the sense of w for which the term r_i votes. Once that all terms r_i have voted (or a limit in the number of neighbors has been reached), the sense of w which received more votes is selected.

This algorithm allows each weighted term to accumulate a score for each sense of the polysemous word. The sense with the highest score is selected. The following equations from McCarthy *et al.* show how the weighted term list accumulates a score for a sense. We use the same equations here, but, as shown in the previous section, the construction of the Term List is different.

$$Weight(ws_i) = \sum_{t_j \in TL_w} sim(w, t_j) \times \frac{pswn(ws_i, t_j)}{\sum_{ws_k \in senses(w)} pswn(ws_k, t_j)}.$$

In these equations, w is the ambiguous word, ws_i is each one of the senses of w, TL_w is the weighted list of terms, and t_j is each term. $P(w, t_j)$ represents the semantic similarity between w and t_j. Note that w and t are words. The similarity value is calculated using the WSM[1]. The second term of the *weight* equation normalizes the weight of w_{si} using all the senses of w and the current t_j.

①We have used simple digits in place of realistic scores to demonstrate the cosine measure.

The function *pswn* returns the sense of a word that has the greatest semantic similarity to a particular sense. For example, $pswn(ws_i, t_j)$ compares all the senses of the quasi-synonym t_j with ws_i and obtains the sense of t_j which has more semantic similarity with regard to ws_i. In the following subsection we describe the measure of similarity used in this algorithm.

2.4 Similarity Measure

To calculate the semantic distance between two senses we use WordNet::Similarity[11]. This package is a set of libraries which implement similarity measures and semantic relationships in WordNet[12-13]. It includes similarity measures proposed by Resnik[14], Lin[4], Jiang-Conrath[15], Leacock-Chodorow[16] among others. Following McCarthy *et al.* approach, we have chosen the Jiang-Conrath similarity measure as well. The Jiang-Conrath measure (jcn) uses exclusively the hyperonym and hyponym relationships in the WordNet hierarchy, and this is consistent with our tests because we are working only on the disambiguation of nouns. The Jiang-Conrath measure obtained the second best result in the experiments presented by Patwardhan *et al.*[6] In that work they evaluate several semantic measures using the WordNet::Similarity package. The best result was obtained with the adapted Lesk measure[6], which uses information of multiple hierarchies and is less efficient.

The Jiang-Conrath measure uses a concept of information content (IC) to measure the specificity of a concept. A concept with a high IC is very specific — for example *dessert_spoon* — while a concept with a lower IC is more general, such as *human_being*. WordNet::Similarity uses SemCor to compute the IC of WordNet concepts. The Jiang-Conrath measure is defined with the following formula:

$$dist_{jcn}(c_1, c_2) = IC(c_1) + IC(c_2) - 2 \times IC(lcs(c_1, c_2))$$

where IC is the information content, lcs (*lowest common subsumer*) is the common lower node of two concepts.

3 Experiments

In this section we describe all the experiments we have made. In Subsection 3.1 we replicate the method proposed in McCarthy *et al.* and compare it with our method. In Subsection 3.2 we describe the corpus used for training and testing the WSM we have built and also give some experimental results about WSD. Finally in Subsection 3.3 we describe deeply the impact of weighted terms in the WSD algorithm we have proposed.

3.1 Comparison with the Original Method

McCarthy *et al.* report 64% using a term list from a static thesaurus calculated using the Lin method. In order to directly compare the performance of our method with the original method of McCarthy *et al.*, we implemented their algorithm and compared it against our method. We varied the maximum number of neighbors. We used the thesaurus built by Lin[4] for their method and ours (for building the DCODB), and we evaluated with the SemCor corpus. The results are shown in Table 2. Traditionally, literature (*v.gr.*, Senseval results) reports performance considering the "disambiguation" of monosemous words together with polysemous words. Monosemous words represent 25% of the nouns of SemCor (97 nouns from 387). McCarthy *et al.* results also include monosemous words, so that, in order to allow a direct comparison, we included results for all the 387 nouns in the experiments of this section — monosemous and polysemous: the *with* column in Table 3. The *without* column shows results for the polysemous words (290 nouns) only. The forthcoming subsections will report results *without* considering monosemous words.

Table 3. Comparison with the Original Method by McCarthy *et al.*

Neighbors	Monosemous			
	Original Method		Our Method	
	Without	With	Without	With
10	53.10	64.86	64.23	73.13
20	**56.06**	**67.10**	69.44	76.94
30	54.45	66.14	67.36	75.66
40	49.47	62.43	66.43	74.87
50	54.45	66.14	67.80	75.93
100	49.82	62.83	**69.86**	**77.54**
Average	52.89	64.92	67.52	75.68

3.2 Role of WSM — Using Different Corpora

We created a WSM using 90% of SemCor corpus (we used it only the raw text part of SemCor for training). We evaluated the model with the remaining 10% of SemCor and Senseval-2 (all words nouns only). We chose these corpora to be able to compare with related work such as McCarthy *et al.*

We created two WSMs from raw text separately:

1) Using 90% of untagged SemCor;

2) Using British National Corpus.

We evaluated separately against:

1) 10% of tagged SemCor;

2) Senseval-2.

When using a corpus for creating a WSM, the semantic tags of word senses were not considered. These tags refer to specific synsets in WordNet.

In these experiments we disambiguated only nouns. For evaluating, we considered the number of weighted terms to choose the right sense. For most of the comparisons, we conducted experiments for the first 10, 20, 30, 40, 50, 60, 70, 100, 200, 500, 1000 and 2000 words from the weighted list of quasi-synonyms.

In both experiments, general results for 10% of the remaining of SemCor corpus were better than for the Senseval-2 corpus. In the first experiment, the best result using SemCor evaluation was 69.86% precision and in the second one 73.07% precision (see Table 4). The results of the second experiment (evaluation with Senseval-2), are better than all the unsupervised methods presented in Senseval-2 (see Table 1). For a direct comparison with the McCarthy *et al.* algorithm, see Subsection 3.1.

Table 4. Precision, Training with SemCor and BNC Evaluation with SemCor and Senseval-2

Tested on (tagged)	10% SemCor		Senseval-2	
Trained on (untagged)	90% SemCor	BNC	90% SemCor	BNC
10	64.23	**73.07**	44.22	51.35
20	69.44	60.00	44.77	52.88
30	67.36	65.27	45.91	53.33
40	66.43	65.16	45.76	53.33
50	67.80	63.80	45.55	53.33
60	68.15	63.41	48.12	55.36
70	**69.86**	63.84	49.84	57.22
100	**69.86**	62.33	48.80	56.02
200	66.75	61.58	49.05	57.57
500	65.89	61.08	49.10	**58.79**
1000	65.06	61.08	44.55	54.27
2000	62.76	61.08	41.05	51.75
Average	61.73	58.38	42.97	54.60

The best results were yielded when building the dynamic thesaurus from the BNC corpus. This might be mainly because BNC is a corpus greater than SemCor and Senseval-2. We expected that using the same corpus for disambiguating itself would yield better results, which in general is the case (see the first two columns under "10% SemCor"), however the top performance on such case is when using 10 neighbors and building the WSM with BNC. A bigger corpus produces richer thesauri which can provide words more adequately for each context.

3.3 Role of Weighted Terms

The purpose of this subsection is to analyze the behaviour of the maximization algorithm, whose role is the same in both our and McCarthy *et al.*'s work: to assign a sense to the ambiguous instance considering each one of the words of the weighted term list (be it dynamic or static). The sense chosen by the maximization algorithm for each word is determined not only by the number of weighted terms which it processes, but also for the quality of the semantic relationship between each one of the members of the list and the ambiguous instance.

McCarthy *et al.* used the first 10, 20, 50 and 70 weighted terms and concluded that the number of weighted terms is not an important feature that influences the performance of their method, thus, they always used the first 50 weighted terms[3] given by the Lin's thesaurus[4]. In our experiments (see Table 4), the best result was obtained using the first 10 weighted terms for 10% SemCor and 500 for Senseval-2. There is an important difference in the results when the number of weighted terms varies.

McCarthy *et al.* tested their algorithm using *static lists* of 10, 30, 50 and 70 weighted terms, given by the Lin's thesaurus, and concluded that the number of words does not have an important influence given the results from the maximization algorithm. If we consider that each list is *ranked*, that is, the top-n words used by the algorithm are those who have more semantic relationship with regard to the ambiguous instance, we can be aware of the fact that the sense chosen by the maximization algorithm for an ambiguous instance does not depend only on the number of words, but also on the semantic relationships that each one of them has with the ambiguous word. This relationship is established by the Lin's thesaurus[4], as this is the lexical resource which provides the weighted list for each ambiguous instance. We use such resource in the same way that it is used in the experiments of McCarthy *et al.*

The maximization algorithm, which tags an ambiguous instance, can be applied to obtain MFS and WSD. Since our goal is to find the impact of the weighted term on such tasks, we performed both experiments as follows:

- impact of the weighted terms for finding the MFS;
- impact of the weighted terms in WSD.

3.3.1 *Impact of the Weighted Terms for Finding the Most Frequent Sense*

In order to determine the impact of this algorithm for detecting the MFS, we use as *test corpus* the MFS of the nouns found in SENSEVAL-2 English all-words, whose ambiguous instance has occurred at least twice, obtaining a total of 34 polysemous nouns, where each one has at least two senses (see Table 5).

In addition, we have compared the MFS that the maximization algorithm chooses for each one of the 34 nouns which the MFS found in WordNet — these were calculated by measuring the frequency of the senses

Table 5. Nouns from SENSEVAL-2

Token	MFS	Senses	Token	MFS	Senses
church	2	4	*individual*	1	2
field	4	13	*child*	2	4
bell	1	10	*risk*	1	4
rope	1	2	*eye*	1	5
band	2	12	*research*	1	2
ringer	1	4	*team*	1	2
tower	1	3	*version*	2	6
group	1	3	*copy*	2	3
year	1	4	*loss*	5	8
vicar	3	3	*colon*	1	5
sort	2	4	*leader*	1	2
country	2	5	*discovery*	1	4
woman	1	4	*education*	1	6
cancer	1	5	*performance*	2	5
cell	2	7	*school*	1	7
type	1	6	*pupil*	1	3
growth	1	7	*student*	1	2

from the SemCor corpus (manually tagged).

For each one of the words listed in Table 4, we applied the maximization algorithm, using 1, 2, 3, 4, 5, 6, 8, 10, 15, 20, 30, 40, 50, 100, 120, 150, 200, 230, 260, 300, 330, 360 and 400 top *weighted terms* provided by the Lin's Thesaurus. We found the following issues:

• The MFS of the words *rope*, *tower*, *vicar*, *woman*, *cancer*, *cell*, *type*, *individual*, *research*, *team*, *copy*, *colon*, *leader* and *discovery* were always correctly found by our algorithm, no matter how many weighted terms were used — see Table 6.

• The MFS of the words *band*, *ringer*, *year*, *sort*, *child*, *version*, *loss*, *performance* and *school* were determined incorrectly, no matter the number of weighted terms used in the process — see Table 6.

The average of the number of senses from nouns whose MFS is always found correctly is less than the number of those whose MFS is always found incorrectly — see Table 7.

The remaining MFS of the 11 remaining nouns: *church, field, bell, group, country, growth, risk, eye, education, pupil*, and *student*, were determined correctly and incorrectly, depending on the number of weighted terms, as it can be seen in Table 7 and Fig.3.

In Table 8, $\surd$ means correct and $\times$ incorrect. It shows only up to 200 weighted terms for these 11 words.

Table 6. General Statistics for Automatically Finding of the MFS

	Nouns	(%)
Total number of nouns evaluated	34	100.00
Successful MFS always found, no matter how many weighted terms were processed	14	41.18
Unsuccessful MFS always found, no matter how many weighted terms were processed	9	26.47
Nouns whose result depends of the processed weighted terms	11	32.35

Table 7. SENSEVAL-2 Nouns Predominant Sense

Predominant Sense Always Correct		Predominant Sense Always Incorrect	
Ambiguous Word	Senses	Ambiguous Word	Senses
rope	2	*band*	12
tower	3	*ringer*	4
vicar	3	*year*	4
woman	4	*sort*	4
cancer	5	*child*	2
cell	7	*version*	2
type	6	*loss*	8
individual	1	*performance*	5
research	2	*school*	7
team	2		
copy	3		
colon	5		
leader	2		
discovery	4		
Average	3.50		5.33

Table 8. Per-Word Disambiguation Analysis

	Number of Weighted Terms														
	1	2	3	4	5	6	8	10	15	20	30	40	50	70	100
church	$\surd$	$\times$	$\times$	$\times$	$\times$	$\times$	$\surd$	$\times$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\times$
field	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$
bell	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\times$	$\surd$	$\times$	$\times$
group	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$
country	$\surd$	$\surd$	$\surd$	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$
growth	$\surd$	$\times$	$\times$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$
risk	$\times$	$\times$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$
eye	$\times$	$\times$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$
education	$\surd$	$\surd$	$\surd$	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$	$\surd$	$\surd$	$\surd$
pupil	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$
student	$\surd$	$\times$	$\times$	$\times$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\times$	$\times$	$\times$	$\times$

We could find a slightly better performance when using more than 200 weighted terms only for 7 words. We experimented with 230, 260, 300, 330 and 360 weighted terms — see Fig.4.

From the evaluated 34 nouns, 23 are *invariant* to the number of weighted terms used by the maximization algorithms, that is, 14 of them are always disambiguated properly, and 9 are always not. The remaining 11 are affected by the number weighted terms selected, as shown in Fig.3 and Fig.4. From the first graph, precision varies from 45.45% and 63.34%, whereas for the second values fluctuate between 42.85% and 85.71%.

The irregularity of Figs. 5 and 6, and the results shown in Table 7 suggest that the role of the weighted terms in the maximization algorithm to obtain the MFS is not as important as it might seem from McCarthy *et al.*, as only 32.5% of the nouns have variations when finding their MFS by using the weighted terms from the Lin's thesaurus. Fig.3 also suggests that the best results is obtained with 15, 20 and 30 weighted terms. Results tend to be stable when more than 100 weighted terms are used. It is interesting to note that for these experiments performance with using only 1 weighted term is similar to that of using 100 or more weighted terms. We believe that this should not be a rule, as we expected that using only one word would not give enough evidence for the maximization algorithm to perform consistently.

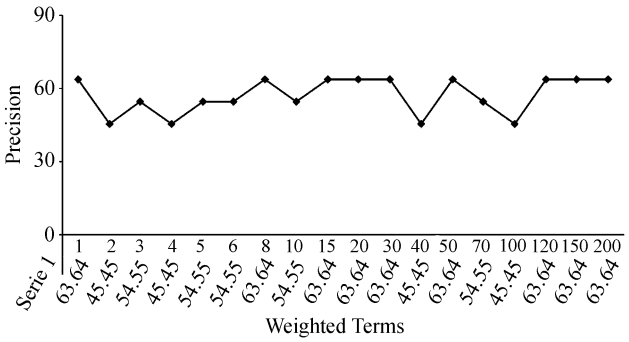


Fig.5. Performance of the maximization algorithm for finding the MFS (11 words).

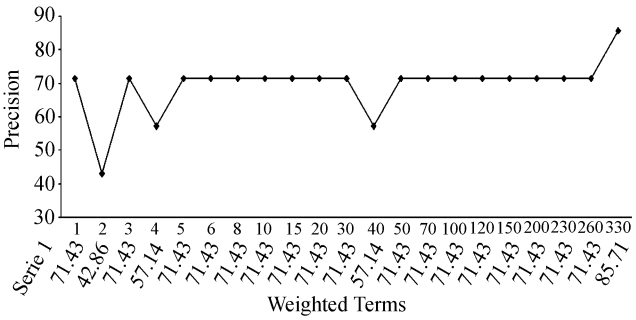


Fig.6. Performance of the maximization algorithm for finding the MFS (7 words).

Fig.6 shows additional information. In this figure, it is possible to see that, while the weighted terms increase, precision tends to be stable. Here we can see that we have the same results for 100, 120, 150 and 200 weighted terms.

3.3.2 Impact of Weighted Terms on WSD

To determine the impact of this algorithm on Word Sense Disambiguation, we used SENSEVAL-2 English all-words as evaluation corpus. This algorithm was originally created to obtain the predominant senses using raw text as source of information; however, it is suitable to be applied to WSD if we use the predominant sense as answer for every ambiguous case. Fig.7 shows the obtained results. We exclude monosemous words for this evaluation, so that results might seem low. For experiments counting the monosemous words, please refer to Section 3.

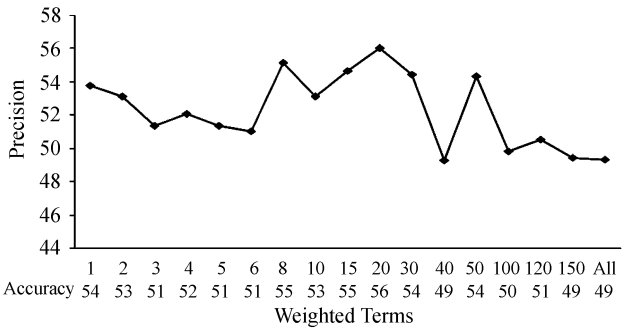


Fig.7. Using the MFS for WSD. Evaluated on SENSEVAL-2.

The best result was 56.74% for 20 weighted terms, whereas the worst result was 49.28% when we used 40 weighted terms, nevertheless with 50 weighted terms the results increment. The second worst result was obtained when all the weighted terms were used — the range of this value is between 80 and 1000, depending on each word.

Similar to MFS, there is not a clear pattern about the number of weighted terms in WSD task. The best results are usually between 8 and 50 weighted terms, and results tend to drop while the number of weighted terms is increased beyond.

4 Conclusions

The method presented disambiguates a corpus more precisely when trained with a richer corpus, as shown by our experiments when training with the BNC corpus used as raw-text, and evaluating with SemCor. We compared against obtaining the most frequent sense from the same set of SemCor and evaluating with the remaining 10%. We obtained a precision of 77.54% against 62.84% of the baseline which uses supervised

information, while our method is unsupervised.

In [17-18], it is shown that the maximization algorithm proposed by McCarthy *et al.*, can be applied to most frequent sense detection and word sense disambiguation. Such algorithm processes a set of weighted terms given by the Lin's thesaurus, and each of them votes for a sense of the ambiguous instance, so that the sense with the highest number of votes is chosen as the most frequent sense. When the maximization algorithm is applied for detecting the most frequent sense, roughly a third part (32.35%) of the nouns in SENSEVAL-2 English all-words task (used for this experiment) are affected by the number of the weighted terms used, i.e., the success in the detection of the most frequent sense depends on the number of weighted terms used by this algorithm.

The most frequent sense in 41% of the processed nouns is always determined correctly, no matter how many weighted terms are used (1, 2, 3, 4, 5, 6, 8, 10, 15, 20, 30, 40, 50, 70, 100, 120, 150, 200, 230, 260, 300, 330, 360 and 400) by the maximization algorithm. In addition, the most frequent sense of 26% of the nouns is always determined incorrectly, without considering the feature of weighted terms.

In conclusion, the number of weighted terms provided by the Lin's thesaurus, which are used by the maximization algorithm, is not the only determining characteristic for finding the most frequent sense. Another characteristic, to be studied further, is the *semantic quality* of the weighted terms, especially, the semantic relationship which exists between each one of them and the ambiguous word. For our experiments, we used the measure provided by the Lin's thesaurus. To explore this characteristic, other sources of information are needed.

Using the MFS for WSD has been proved to be a baseline, which in some cases is not even surpassed by unsupervised systems; however, this can be regarded best as a back-off technique, since no information of context is considered.

As a future work, we plan to use a manually obtained thesaurus as a source for the top-n weighted terms. However, this kind of resource would not include a quantified measure of the semantic relationship between different words — or, put in other words, all measures are binary, there is a relationship, or there is not.

Focusing on the third part of words, which are affected by the number of weighted terms to obtain their most frequent sense by this maximization algorithm, we can particularly conclude that with a greater amount of related words, the results are improved, which agrees with the theoretical background of this algorithm.

On the optimum number of weighted terms we found that this number varies irregularly, so that we cannot conclude globally that the optimum number is always the same. In addition, terms from the weighted list (our dynamic thesaurus) are not always clearly related between them. We expect to build a resource to improve the semantic quality from such terms.

From the WSD experiment that we presented, we can conclude that when using a greater amount of weighted terms, precision decreases. This, of course, does not reflect the real impact of the weighted terms in WSD. For this study we suggest selecting the elements of the list of weighted terms, discarding those that are not related with the context of the ambiguous word. Using this selection on the Lin's thesaurus is part of our plan for future work.

Finally, it is difficult to determine the main factor that has a greater impact in the proposed disambiguation method: the process of obtaining a weighted list of terms (the dynamic thesaurus), or the maximization algorithm. This is because the DCODB sometimes does not provide terms related with a word; additionally, the definitions for each sense of WordNet are sometimes very short. Moreover, as it has been stated previously, for several tasks the senses provided by WordNet are very fine-graded, so that a semantic measure might be not accurate enough.

Acknowledgement The authors wish to thank Rada Mihalcea for her useful comments and discussion.

References

[1] Schütze H. Dimensions of meaning. In *Proc. ACM/IEEE Conference on Supercomputing* (*Supercomputing 1992*), Mannheim, Germany, June, 1992, pp.787-796.

[2] Karlgren J, Sahlgren M. From Words to Understanding. Foundations of Real-World Intelligence, Stanford: CSLI Publications, 2001, pp.294-308.

[3] McCarthy D, Koeling R, Weeds J *et al.* Finding predominant word senses in untagged text. In *Proc. the 42nd Annual Meeting of the Association for Computational Linguistics*, Barcelona, Spain, 2004.

[4] Lin D. Automatic retrieval and clustering of similar words. In *Proc. the 17th Int. Conf. Computational Linguistics*, Montreal, Canada, Aug. 10-14, 1998, pp.768-774.

[5] Kilgarriff A, Rosenzweig J. English SENSEVAL: Report and results. In *Proc. LREC*, Athens, May-June 2000.

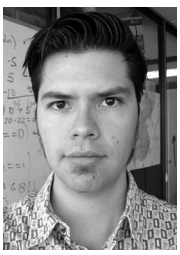
[6] Patwardhan S, Banerjee S, Pedersen T. Using measures of semantic relatedness for word sense disambiguation. In *Proc. the Fourth International Conference on Intelligent Text Processing and Computational Linguistics*, Mexico City, Mexico, 2003, pp.241-257.

[7] Sahlgren M. The Word-Space Model: Using distributional analysis to represent syntagmatic and paradigmatic relations between words in high-dimensional vector spaces [Ph.D. Dissertation]. Department of Linguistics, Stockholm University, 2006.

[8] Lin D. Dependency-based evaluation of MINIPAR. In *Proc. Workshop on the Evaluation of Parsing Systems at LREC*,

Granada, Spain, 1998, pp.317-330.
[9] Hays D. Dependency theory: A formalism and some observations. *Language*, 1964, 40(4): 511-525.
[10] Mel'čuk I A. Dependency Syntax: Theory and Practice. State University of New York Press, Albany, N.Y., 1987.
[11] Pedersen T, Patwardhan S, Michelizzi J. WordNet::Similarity: Measuring the relatedness of concepts. In *Proc. the Nineteenth National Conference on Artificial Intelligence* (*AAAI-2004*), San Jose, CA, 2004, pp.1024-1025.
[12] Miller G. Introduction to WordNet: An On-line Lexical Database. Princeton Univesity, 1993.
[13] Miller G. WordNet: An on-line lexical database. *International Journal of Lexicography*, 1990, 3(4): 235-244.
[14] Resnik P. Using information content to evaluate semantic similarity in a taxonomy. In *Proc. the 14th International Joint Conference on Artificial Intelligence*, Montreal, Canada, Aug. 20-25, 1995, pp.448-453.
[15] Jiang J J, Conrath D W. Semantic similarity based on corpus statistics and lexical taxonomy. In Proc. International Conference on Research in Computational Linguistics, Taiwan, China, Sept. 1997, pp.19-33.
[16] Leacock C, Chodorow M. Combining Local Context and WordNet Similarity for Word Sense Identification. WordNet: An Electronic Lexical Database, Fellbaum C (ed.), 1998, pp.265-283.
[17] Tejada J, Gelbukh A, Calvo H. Unsupervised WSD with a dynamic thesaurus. In *Proc. the 11th International Conference on Text, Speech and Dialogue* (*TSD 2008*), Brno, Czech, Sept. 8-12, 2008, pp.201-210.
[18] Tejada J, Gelbukh A, Calvo H. An innovative two-stage WSD unsupervised method. *SEPLN Journal*, March 2008, 40: 99-105.

Javier Tejada-Cárcamo was born in Perú in 1976. He obtained his Master's degree in computer science (with honors) in 2005 from the Center for Computing Research (CIC) of the National Polytechnic Institute (IPN), Mexico, and his Ph.D. degree in computer science (with honors) in 2009 at the same Center. Since 2010 he is an associated professor and researcher at San Pablo Catholic University in Arequipa. Peru. He works as project leader at Research and Software Development Center of the San Agustin National University in Arequipa, Peru.



Hiram Calvo was born in Mexico in 1978. He obtained his Master's degree in computer science in 2002 from National Autonomous University of Mexico (UNAM), with a thesis on mathematical modeling, and his Ph.D. degree in computer science (with honors) in 2006 from CIC of IPN, Mexico. Since 2006 he is a lecturer at CIC of IPN. He was awarded with the Lázaro Cárdenas Prize in 2006 as the best Ph.D. candidate of IPN in the area of physics and mathematics. This Prize was handed personally by the President of Mexico. Currently he is a visiting researcher at the Nara Institute of Science and Technology, Japan. He is a JSPS fellow.

Alexander Gelbukh holds a honors M.Sc. degree in mathematics from the Moscow State Lomonosov University, Russia, 1990, and Ph.D. degree in computer science from the All-Russian Institute for Scientific and Technical Information, Russia, 1995. He has been a research fellow at the All-Union Center for Scientific and Technical Information, Moscow, Russia; distinguished visiting professor at Chung-Ang University, Seoul, Korea, and is currently research professor and head of the Natural Language Processing Laboratory of the Center for Computing Research of the National Polytechnic Institute, Mexico, and invited professor of the National University, Bogota, Colombia. He is an academician of the Mexican Academy of Sciences, National Researcher of Mexico, and the executive board secretary of the Mexican Society for Artificial Intelligence. His recent awards include the prestigious Research Diploma from the National Polytechnic Institute, Mexico. His main areas of interest are computational linguistics and artificial intelligence. He is author, co-author or editor of more than 400 publications; member of editorial board or reviewer for a number of international journals. He has been program committee member of about 150 international conferences and Chair, Honorary Chair, or Program Committee Chair of more than 20 international conferences, as well as principal investigator of several projects, funded governmentally or internationally, in the field of computational linguistics and information retrieval.

Kazuo Hara was born in Tokyo, Japan, in 1971. He received his Master's degree of engineering from the University of Tokyo, and his Ph.D. degree from Nara Institute of Science and Technology. His research interests include natural language processing aiming for information extraction, such as coordinate structure analysis and word sense disambiguation. Previously he was the team leader in Sankyo Co., LTD, the 2nd largest pharmacy company in Japan, where he composed statistical analysis plans and performed statistical hypothetical testing for new drug candidate compositions in clinical trials. He has experience in bioinformatics and statistics as well. Currently he is a postdoctoral researcher at the Nara Institute of Science and Technology, Japan.