On the Portability and Tuning of Supervised Word Sense Disambiguation Systems

Gerard Escudero, Lluís Màrquez and German Rigau

TALP Research Center
Software Department (LSI)
Technical University of Catalonia (UPC)
Jordi Girona Salgado 1–3
E-08034 Barcelona. Catalonia
{escudero,lluism,g.rigau}@lsi.upc.es

Abstract

This report describes a set of experiments carried out to explore the portability of alternative supervised Word Sense Disambiguation algorithms. The aim of the work is threefold: firstly, studying the performance of these algorithms when tested on a different corpus from that they were trained on; secondly, exploring their ability to tune to new domains, and thirdly, demonstrating empirically that the LazyBoosting algorithm outperforms state-of-the-art supervised WSD algorithms in both previous situations.

Keywords: Word Sense Disambiguation, Machine Learning, Natural Language Processing, Portability and Tuning of NLP systems.

1 Introduction

Word Sense Disambiguation (WSD) is the problem of assigning the appropriate meaning (sense) to a given word in a text or discourse where this meaning is distinguishable from other senses potentially attributable to that word. As an example, table 1 shows the definition of two senses of the word *age*[1] and an example sentence[2] for each sense. Thus, a WSD system must be able to assign the correct sense of a given word, for instance *age*, depending on the context in which the word occurs.

Resolving the ambiguity of words is a central problem for Natural Language Processing (NLP) applications and their associated tasks (Ide and Véronis, 1998), including, for instance, natural language understanding, machine translation, information retrieval and hypertext navigation, parsing, acquisition of subcategorization patterns, selectional restrictions, speech synthesis, spelling correction, reference resolution, automatic text summarization, etc.

WSD is one of the most important open problems in NLP. Despite the wide range of approaches investigated (Kilgarriff and Rosenzweig, to appear) and the large effort devoted to tackle this

[1]The senses of the example have been taken from the WordNet 1.5 (Miller et al., 1990).
[2]These examples have been taken from the DSO corpus (Ng and Lee, 1996).

Sense Definitions	
age 1	the length of time something (or someone) has existed; "his age was 71"; "it was replaced because o its age"
age 2	a historic period; "the Victorian age"; "we live in a litigious age"

Corpora Examples	
age 1	He was mad about stars at the **age** of nine .
age 2	About 20,000 years ago the last ice **age** ended .

Table 1: Sense definitions and corpora examples.

problem, it is a fact that to date, no large-scale broad-coverage and highly accurate WSD system has been built.

One of the most successful current lines of research is the corpus-based approach in which statistical or Machine Learning (ML) algorithms have been applied to learn statistical models or classifiers from corpora in order to perform WSD. Generally, supervised approaches (those that learn from previously semantically annotated corpus) have obtained better results than unsupervised methods on small sets of selected ambiguous words, or artificial pseudo-words. Many standard ML algorithms for supervised learning have been applied to WSD, including: Decision Lists (Yarowsky, 1994), Neural Networks (Towell and Voorhees, 1998), Bayesian learning (Bruce and Wiebe, 1999), Exemplar–based learning (Ng, 1997a; Fujii et al., 1998), and Boosting (Escudero et al., 2000a). Further, in (Mooney, 1996) some of the previously cited methods are compared, jointly with Decision Trees and Rule Induction algorithms, on a very restricted domain.

The performance of supervised ML based systems is usually calculated by testing the algorithm on a separate part of the set of annotated examples (say 10–20%), or by N–fold cross–validation, in which the set of examples is partitioned into N disjoint sets (or folds), and the training–test procedure is repeated N times using all combinations of $N-1$ folds for training and 1 fold for testing. In both cases, test examples are different from those used for training, but they belong to the same corpus, and, therefore, they are expected to be quite similar.

Although this methodology could be valid for certain NLP problems, such as English Part-of-Speech tagging, we think that there exists reasonable evidence to say that, in WSD, accuracy results cannot be simply extrapolated to other domains:

- WSD is very dependant on the domain of application. In (Gale et al., 1992), the idea of "one sense per discourse" has been suggested, that is, if a polysemous word appears two or more times in a well-written discourse, it is extremely likely that they will all share the same sense.
- Further, see (Ng and Lee, 1996; Ng, 1997a), where quite different accuracy figures are obtained when testing an Exemplar–based WSD classifier on two different corpora, namely Wall Street Journal and Brown Corpus (both corpora belong to different domains).
- It does not seem reasonable to think that the training material is large and representative enough to cover "all" potential types of examples.

We think that the study of the domain dependence of WSD —in the style of other studies devoted to parsing (Sekine, 1997)— is needed to assess the validity of the supervised approach, and to determine to which extent a tuning process is necessary to make real WSD systems portable. In order to corroborate the previous hypotheses, this work explores the portability and tuning of four different ML algorithms by training and testing them on different corpora.

1.1 Is Supervised WSD Really Viable?

It is well-known that supervised methods suffer from the lack of widely available semantically tagged corpora, from which to construct really broad coverage WSD systems. This is known as the "knowledge acquisition bottleneck" (Gale et al., 1993). In (Ng, 1997b) is estimated that the manual annotation effort necessary to build a broad coverage semantically annotated English corpus would be about 16 man–years. This extremely high overhead for supervision (which could be much greater if a costly tuning procedure is required before applying any existing system to each new domain) and, additionally, the also serious learning overhead when common ML algorithms scale to real size WSD problems, explain why supervised methods have been seriously questioned.

Due to this fact, recent works have focused on reducing the acquisition cost, the need for supervision, and the computational requirements in corpus-based methods for WSD. Consequently, the following four lines of research are being explored:

1. The design of efficient sampling methods (Engelson and Dagan, 1996; Fujii et al., 1998);
2. The use of external lexical resources, such as WordNet (Miller et al., 1990), and Web search engines to automatically obtain from Internet arbitrarily large samples of word senses (Leacock et al., 1998; Mihalcea and Moldovan, 1999);
3. The use of unsupervised EM-like algorithms for estimating the statistical model parameters (Pedersen and Bruce, 1998).
4. The application of efficient and accurate ML algorithms (Escudero et al., 2000a) and attribute representations (Escudero et al., 2000b) in order to deal with real size WSD problems.

Solving the problem of knowledge acquisition is crucial for making the unsupervised WSD approach viable. It is our belief that the referred work, and especially second and fourth lines, provides enough evidence towards the "opening" of the bottleneck in the near future. For that reason, it is worth further investigating the robustness and portability of existing supervised ML methods to better resolve the WSD problem.

1.2 Overview

This report is organized as follows: Section 2 presents the four ML algorithms used. In section 3 the general setting is presented, including the corpora and the experimental methodology used. Section 4 reports the experiments carried out and the results obtained. Finally, section 5 concludes and outlines some lines for further research.

2 Learning Algorithms Tested

This section provides some references and short descriptions of the main characteristics of the four learning methods compared in this report: Naive Bayes, Exemplar–based, Snow and LazyBoosting.

2.1 Naive Bayes (NB)

Naive Bayes is intended as a simple representative of statistical learning methods. It has been used in its most classical setting (Duda and Hart, 1973). That is, assuming independence of features, it classifies a new example by assigning the class that maximizes the conditional probability of the class given the observed sequence of features of that example.

Let $\{C_1 \ldots C_m\}$ be the set of classes and $\bigcap v_j$ the set of feature values of a test example. The Naive Bayes method tries to find the class that maximizes $P(C_i \mid \cap v_j)$:

$$\arg\max_i P(C_i \mid \cap v_j) \approx \arg\max_i P(C_i) \prod_j P(v_j \mid C_i),$$

where $P(C_i)$ and $P(v_j \mid C_i)$ are estimated during training process using relative frequencies.

To avoid the effects of zero counts when estimating the conditional probabilities of the model, a very simple smoothing technique, proposed in (Ng, 1997a), has been used. It consists in replacing zero counts of $P(v_j \mid C_i)$ with $P(C_i)/N$ where N is the number of training examples.

Despite its simplicity, Naive Bayes is claimed to obtain state–of–the–art accuracy on supervised WSD in many papers (Mooney, 1996; Ng, 1997a; Leacock et al., 1998).

2.2 Exemplar–based Classifier (EB)

In Exemplar–based learning (Aha et al., 1991) no generalization of training examples is performed. Instead, the examples are stored in memory and the classification of new examples is based on the classes of the most similar stored examples. In our implementation, all examples are stored in memory and the classification of a new example is based on a k–NN (Nearest–Neighbours) algorithm using Hamming distance to measure closeness (in doing so, all examples are examined). For k's greater than 1, the resulting sense is the weighted majority sense of the k nearest neighbours —where each example votes its sense with a strength proportional to its closeness to the test example.

In the experiments explained in section 4, the EB algorithm is run several times using different number of nearest neighbours (1, 3, 5, 7, 10, 15, 20 and 25) and the results corresponding to the best choice are reported[3].

Exemplar–based learning is said to be the best option for WSD (Ng, 1997a). Other authors (Daelemans et al., 1999) point out that Exemplar–based methods tend to be superior in language learning problems because they do not forget exceptions.

2.3 Snow: A Winnow–based Classifier

Snow (Golding and Roth, 1999) stands for Sparse Network of Winnows. It is a on-line learning system which has the Winnow algorithm as the basic component. This method has not been applied to WSD before. For this reason it will be described in more detail.

2.3.1 Winnow Algorithm

Winnow (Littlestone, 1988) is a linear threshold algorithm for 2-class problems with binary (i.e., 0/1-valued) input features. It classifies a new example x into positive class if

$$\sum w_f > \theta, \ \ \forall f \in F$$

[3] In order to construct a real EB–based system for WSD, the k parameter would be estimated by cross–validation using only the training set (Ng, 1997a).

and into negative class otherwise. In this formulation, F is the set of *active features* of x (i.e. those that have value 1), w_f is the weight associated to input feature f and θ is the *threshold parameter*. Winnow is an online algorithm; it accepts examples one-at-a-time and updates the weights w_j as necessary.

Winnow initializes its weights w_f to 1. It then accepts a new training example (x,y), where y is the class of x, and applies the threshold rule to compute the predicted class y'. Winnow is a mistake–driven algorithm, that is, it updates its hypotheses only when a mistake is made. If the predicted class is correct ($y = y'$), Winnow does nothing. However, if the predicted class is wrong, Winnow updates its weights as follows. If y' is negative and y positive, then the weights of active features are promoted (informally, they were too low):

$$\forall f \in F,\ w_f \leftarrow \alpha \cdot w_f$$

where α is a number greater than 1 called *promotion parameter*. If y' is positive and y negative, then the weights corresponding active features are demoted (informally, they were too high):

$$\forall f \in F,\ w_f \leftarrow \beta \cdot w_f$$

where β is a positive number lower than 1 called *demotion parameter*.

2.3.2 Snow Architecture

In the Snow architecture there is a Winnow node for each class, which learns to separate that class from all the rest. Figure 1 shows the Snow architecture learned to disambiguate between the two senses of *age* presented in table 1.

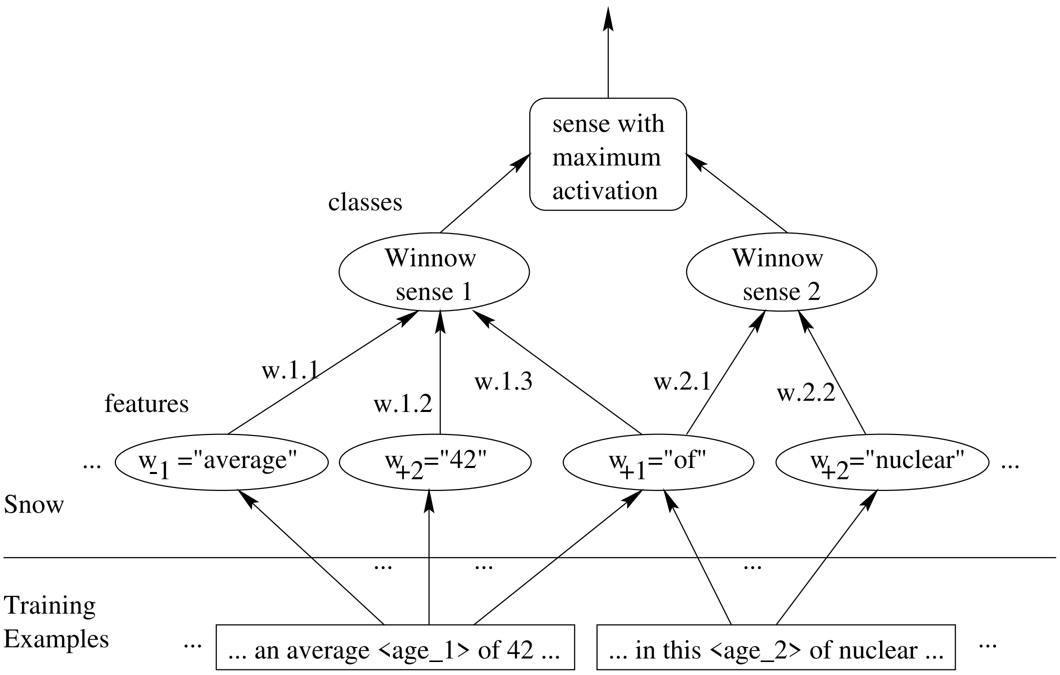

Figure 1: Snow Architecture

Initially, there are no connections between features and classes. During the training step, examples are processed sequentially. Each training example is a positive example for the class it belongs and negative for the rest. A key point that allows a fast learning is that the Winnow nodes are not connected to all features but only to those that are "relevant" for their class (i.e. those that appear in at least one training example of that class). For instance, in figure 1, the features "$w_{+1} = average$" and "$w_{+2} = 42$" are connected only to the first node (and "$w_{+2} = nuclear$" only to the second node), while feature "$w_{+1} = of$" is connected to both winnow nodes because it appears in examples of both classes. These connections between features and classes are created dynamically as the learning examples are processed.

In (Dagan et al., 1997) the training step is repeated several times so as to better adjust the weights of the network. In order to clarify this, these training step repetitions are equivalent to the Neural Network learning epochs. However, the utility of that retraining step is dubious, given that it is not even mentioned in some papers in which Snow is applied (Roth and Zelenko, 1998; Golding and Roth, 1999). Retraining implies to add a new parameter to the system, "when to stop". In (Dagan et al., 1997) the training process finishes when:

1. no mistakes are made on the training examples; or
2. a maximum number of epochs is achieved (say 50).

When classifying a new example, Snow is similar to a neural network which takes the input features and outputs the class with the highest activation.

Table 2 shows the particular parameter values used in the following experiments[4]. These values have been empirically determined and they are those that lead to better accuracy of the algorithm in the WSD task.

Parameter	Value
Promotion (α)	1.5
Demotion (β)	0.8
Threshold (θ)	1
Initial weights	$\frac{\theta}{\|F\|}$
Number of training epochs	1

Table 2: Snow Parameter Values

Snow is proven to perform very well in high dimensional domains, where both, the training examples and the target function reside very sparsely in the feature space (Roth, 1998), such as text categorization (Dagan et al., 1997) or context–sensitive spelling correction (Golding and Roth, 1999).

2.4 LazyBoosting (LB)

The main idea of boosting algorithms is to combine many simple and moderately accurate hypotheses (called weak classifiers) into a single, highly accurate classifier. The weak classifiers are trained sequentially and, conceptually, each of them is trained on the examples which were the most difficult to classify by the preceding weak classifiers. These weak hypotheses are then linearly combined into a single rule called the combined hypothesis.

More particularly, the Schapire and Singer's AdaBoost.MH algorithm for multiclass multi–label classification (Schapire and Singe, to appear) has been used. As in that paper, very simple weak hypotheses are used. They test the value of a boolean predicate and make a real–valued

[4]When a new connection between a feature and a class is activated its weight is initialized to the value $\frac{\theta}{\|F\|}$, which is the threshold value divided by the number of active features of the current example.

prediction based on that value. The predicates used, which are the binarization of the attributes described in section 3.2, are of the form "$f = v$", where f is a feature and v is a value (e.g: "previous_word = *hospital*"). Each weak rule uses a single feature, and, therefore, they can be seen as simple decision trees with one internal node (testing the value of a binary feature) and two leaves corresponding to the yes/no answers to that test.

LazyBoosting, explained in (Escudero et al., 2000a), is a simple modification of the AdaBoost.MH algorithm, which consists of reducing the feature space that is explored when learning each weak classifier. More specifically, a small proportion p of attributes are randomly selected and the best weak rule is selected only among them. The idea behind this method is that if the proportion p is not too small, probably a sufficiently good rule can be found at each iteration. Besides, the chance for a good rule to appear in the whole learning process is very high. Another important characteristic is that no attribute needs to be discarded and so we avoid the risk of eliminating relevant attributes. The method seems to work quite well since no important degradation is observed in performance for values of p greater or equal to 5% (this may indicate that there are many irrelevant or highly dependant attributes in our domain). Therefore, this modification significantly increases the efficiency of the learning process (empirically, up to 7 times faster) with no loss in accuracy.

3 Setting

3.1 The DSO Corpus

The DSO corpus is a semantically annotated corpus containing 192,800 occurrences of 121 nouns and 70 verbs[5], corresponding to the most frequent and ambiguous English words. This corpus was collected by Ng and colleagues (Ng and Lee, 1996) and it is available from the Linguistic Data Consortium (LDC)[6].

The DSO corpus contains sentences from two different corpora, namely Wall Street Journal (WSJ) and Brown Corpus (BC). Therefore, it is easy to perform experiments about the portability of alternative systems by training on the WSJ part and testing on the BC part, or vice-versa. Hereinafter, the WSJ part of DSO will be referred to as corpus A, and the BC part to as corpus B. At a word level, we force the number of examples of corpus A and B be the same[7] in order to have symmetry and allow the comparison in both directions.

From these corpora, a group of 21 words which frequently appear in the WSD literature has been selected to perform the comparative experiments (each word is treated as a different classification problem). These words are 13 nouns (age, art, body, car, child, cost, head, interest, line, point, state, thing, work) and 8 verbs (become, fall, grow, lose, set, speak, strike, tell). Table 3 contains information about the number of examples, the number of senses, and the percentage of the most frequent sense (MFS) of these reference words, grouped by nouns, verbs, and all 21 words.

3.2 Attributes

Two kinds of information are used to perform disambiguation: local and topical context.

[5]These examples, consisting of the full sentence in which the ambiguous word appears, are tagged with a set of labels corresponding, with minor changes (Ng et al., 1999), to the senses of WordNet 1.5 (Miller et al., 1990).

[6]LDC address: `http://www.ldc.upenn.edu/`

[7]This is achieved by reducing the size of the largest corpus (by performing random elimination of examples) to the size of the smallest.

		examples			senses			MFS (%)		
corpus	PoS	min	max	avg	min	max	avg	min	max	avg
A	nouns	122	714	420	2	24	7.7	37.9	90.7	59.8
	verbs	101	741	369	4	13	8.9	20.8	81.6	49.3
	all	101	741	**401**	2	24	**8.1**	20.8	90.7	**56.1**
B	nouns	122	714	420	3	24	8.8	21.0	87.7	45.3
	verbs	101	741	369	4	14	11.4	28.0	71.7	46.3
	all	101	741	**401**	3	24	**9.8**	21.0	87.7	**45.6**

Table 3: Information about the set of 21 words of reference.

Let "$\ldots w_{-3}\ w_{-2}\ w_{-1}\ w\ w_{+1}\ w_{+2}\ w_{+3} \ldots$" be the context of consecutive words around the word w to be disambiguated, and $p_{\pm i}$ $(-3{\leq}i{\leq}3)$ be the part–of–speech tag of word $w_{\pm i}$[8]. Attributes referring to local context are the following 15:

$$p_{-3},\ p_{-2},\ p_{-1},\ p_{+1},\ p_{+2},\ p_{+3},\ w_{-1},\ w_{+1},\ (w_{-2},w_{-1}),\ (w_{-1},w_{+1}),\ (w_{+1},w_{+2}),$$
$$(w_{-3},w_{-2},w_{-1}),\ (w_{-2},w_{-1},w_{+1}),\ (w_{-1},w_{+1},w_{+2}) \text{ and } (w_{+1},w_{+2},w_{+3})$$

where the last seven correspond to collocations of two and three consecutive words.

The topical context is formed by $c_1,\ldots,c_m$, which stand for the unordered set of open class words appearing in the sentence[9].

The four methods tested translate this information into features in different ways. Snow and LB algorithms require binary features. Therefore, local context attributes have to be binarized in a preprocess, while the topical context attributes remain as binary tests about the presence/absence of a concrete word in the sentence. As a result the number of attributes is expanded to several thousands (from 1,764 to 9,900 depending on the particular word).

The binary representation of the topical–context attributes is not appropriate for NB and EB algorithms. Such a representation leads to an extremely sparse vector representation of the examples, since in each example only a few words, among all possible, are observed. Thus, the examples are represented by a vector of thousands of 0's and only a few 1's. In this situation two examples will coincide in the majority of the values of the attributes (roughly speaking in "all" the zeros) and will probably differ in those positions corresponding to 1's. This fact wrongly biases the similarity measure (and thus the classification) used in EB in favour of that stored examples which have less 1's, that is, those corresponding to the shortest sentences.

In order to address this limitation we propose (Escudero et al., 2000b) to reduce the attribute space by collapsing all binary attributes $c_1,\ldots,c_m$ in a single set–valued attribute c that contains, for each example, all closed class words that appear in the sentence. In this setting, the similarity S between two values $V_i = \{w_{i_1}, w_{i_2}, \ldots, w_{i_n}\}$ and $V_j = \{w_{j_1}, w_{j_2}, \ldots, w_{j_m}\}$ is redefined as: $S(V_i,V_j) = \parallel V_i \cap V_j \parallel$, that is, equal to the number of words shared.

This approach implies that a test example is classified taking into account the information about the words it contains (*positive* information), but not the information about the words it does not contain.

[8]In this experiment, the Decision-tree-based tagger by (Màrquez et al., 1999) has been used to annotate training examples with the part–of–speech labels.

[9]The already described set of attributes contains those attributes used in (Ng and Lee, 1996), with the exception of the morphology of the target word and the verb–object syntactic relation.

On the other hand, NB have problems with the efficiency when classifying new examples, since a product of thousands of conditional probabilities (real values) must be done to classify a new example. In order to improve performance, we tested a variant of the Naive Bayes algorithm which does not take into account all attributes but only the conditional probabilities corresponding to the words that appear in the test examples (again positive information). In this way, the efficiency is increased with no loss in accuracy (Escudero et al., 2000b).

These variants of NB and EB algorithms are called PNB and PEB, standing for Positive EB and NB. They are fully tested in (Escudero et al., 2000b) and empirically proven to perform much better than other variants of handling available attributes.

3.3 Experimental Methodology

The comparison of algorithms has been performed in series of controlled experiments using exactly the same training and test sets. There are 7 combinations of training–test sets called: A+B–A+B, A+B–A, A+B–B, A–A, B–B, A–B, and B–A, respectively. In this notation, the training set is placed at the left hand side of symbol "–", while the test set is at the right hand side. For instance, A–B means that the training set is corpus A and the test set is corpus B. The symbol "+" stands for set union, therefore A+B–B means that the training set is A union B and the test set is B.

When comparing the performance of two algorithms, two different statistical tests of significance have been applied depending on the case. A–B and B–A combinations represent a single training–test experiment. In this cases, the McNemar's test of significance is used (with a confidence value of: $\chi^2_{1,0.95} = 3.842$), which is proven to be more robust than a simple test for the difference of two proportions.

In the other combinations, a 10-fold cross-validation was performed in order to prevent testing on the same material used for training. In these cases, accuracy/error rate figures reported in section 4 are averaged over the results of the 10 folds. The associated statistical tests of significance is a paired Student's t-test with a confidence value of: $t_{9,0.975} = 2.262$.

Information about both statistical tests can be found at (Dietterich, 1998).

When classifying test examples, all methods resolve potential ties between senses by choosing the most frequent sense among all those tied.

4 Experiments

4.1 First Experiment

Table 4 shows the accuracy figures of the four methods in all combinations of training and test sets[10]. MFC stands for a Most–Frequent–sense Classifier, that is, a naive classifier that learns the most frequent sense of the training set and uses it to classify all examples of the test set. Averaged results are presented for nouns, verbs, and overall, and the best results for each case are printed in boldface.

It can be observed that LB outperforms all other methods in all cases. Additionally, this superiority is statistically significant, except when comparing LB to the EB approach in the cases marked with an asterisk. Note that, surprisingly, LB in A+B–A (or A+B–B) does not achieve substantial improvement to the results of A–A (or B–B) —in fact, the first variation is not statistically significant and the second is only slightly significant. That is, the knowledge

[10]The second and third column correspond to the same train and test sets presented in (Ng and Lee, 1996; Ng, 1997a)

acquired from a single corpus almost covers the knowledge of combining both corpora. This effect is also observed in the other methods, specially in some cases (e.g. Snow in A+B–A vs. A–A) in which the joining of both training corpora is even counterproductive.

		Accuracy (%)						
Method	PoS	A+B–A+B	A+B–A	A+B–B	A–A	B–B	A–B	B–A
MFC	nouns	46.59	56.68	36.49	59.77	45.28	33.97	39.46
	verbs	46.49	48.74	44.23	48.85	45.96	40.91	37.31
	total	46.55	53.90	39.21	55.94	45.52	36.40	38.71
NB	nouns	62.29	68.89	55.69	66.93	56.17	36.62	45.99
	verbs	60.18	64.21	56.14	63.87	57.97	50.20	50.75
	total	61.55	67.25	55.85	65.86	56.80	41.38	47.66
EB	nouns	62.66	69.45	56.09	69.38	56.17	42.15	50.53
	verbs	63.67	68.39	58.58	68.25	59.57	51.19	52.24
	total	63.01	69.08	56.97	68.98	57.36	45.32	51.13
Snow	nouns	61.24	66.36	56.11	68.85	56.55	42.13	49.96
	verbs	60.35	64.11	56.58	63.91	55.36	47.66	49.39
	total	60.92	65.57	56.28	67.12	56.13	44.07	49.76
LB	nouns	**66.00**	**72.09**	**59.92**	**71.69**	**58.33**	**43.92**	**51.28***
	verbs	**66.91**	**71.23**	**62.58**	**70.45***	**60.14***	**52.99**	**53.29***
	total	**66.32**	**71.79**	**60.85**	**71.26**	**58.96**	**47.10**	**51.99***

Table 4: Accuracy results of the methods on all training–test combinations

Regarding the portability of the systems, very disappointing results are obtained. Restricting to LB results, we observe that the accuracy obtained in A–B is 47.1% while the accuracy in B–B (which can be considered an upper bound for LB in B corpus) is 59.0%, that is, a difference of 12 points. Furthermore, 47.1% is only slightly better than the most frequent sense in corpus B, 45.5%. The comparison in the reverse direction is even worse: a difference of 19% from A–A to B–A, which is lower than the most frequent sense of corpus A, 55.9%.

Apart from accuracy figures, the observation of the predictions made by the four methods on the test sets provides interesting information about the comparison of the algorithms. Figure 2 shows the agreement rates and the Kappa (κ) statistics[11] between all pairs of methods in the A–A, A–B, B–B and B–A cases. 'DSO' stands for the annotation of DSO corpus, which is taken as the correct. Therefore the agreement rate with DSO contain the accuracy results previously reported. Some interesting conclusions can be drawn from those tables:

1. NB obtains the most similar results with regard to MFC in agreement rate and Kappa values in all tables. The average agreement ratio between the four tables is 83% and it achieves 89% in one case. This means that more than 8 out of 10 times it predicts the most frequent sense.
2. LB obtains the most similar results with regard to DSO (accuracy) in agreement rate and Kappa values, and it has the less similar Kappa and agreement values with regard to

[11] The Kappa statistic k (Cohen, 1960) is a better measure of inter–annotator agreement which reduces the effect of chance agreement. It has been used for measuring inter–annotator agreement during the construction of some semantic annotated corpora (Véronis, 1998; Ng et al., 1999).

MFC (furthermore, LB provides the most dissimilar annotation with respect to the rest of algorithms). This indicates that LB is the method that better learns the behaviour of the DSO examples.

3. The Kappa values are very low. But, as it is suggested in (Véronis, 1998), evaluation measures, such as precision and recall, should be computed relative to the agreement between the human annotators of the corpus and not to a theoretical 100%. It seems pointless to expect more agreement between the system and the reference corpus than between the annotators themselves. Contrary to the intuition that the agreement between human annotators should be very high in the WSD task, some papers report surprisingly low figures. For instance, (Ng et al., 1999) reports an accuracy rate of 56.7% and a Kappa value of 0.317 when comparing the annotation of a subset of the DSO corpus performed by two independent research groups[12]. Similarly, (Véronis, 1998) reports values of Kappa near to zero when annotating some special words for the ROMANSEVAL corpus[13]. From this point of view, the Kappa values of 0.44 and 0.32 achieved by LB in A–A and B–B could be considered excellent results. Unfortunately, the subset of the DSO corpus and that used in this report are not the same and, therefore, a direct comparison is not possible. Furthermore, note that the LB Kappa values when moving from one corpus to another (A–B and B–A) are extremely low (0.14 and 0.18, respectively).

4.2 Second Experiment

The previous experiment shows that classifiers trained on the A corpus do not work well on the B corpus, and vice-versa. Therefore, it seems that a tuning process is necessary to make supervised systems portable.

This second experiment explores the effect of a tuning process consisting of adding to the original training set a relatively small sample of manually sense tagged examples of the new domain. The size of this supervised portion varies from 10% to 50% of the available corpus in steps of 10% (the remaining 50% is kept for testing). This set of experiments will be referred to as A+%B–B, or conversely, to B+%A–A.

In order to determine to which extent the original training set contributes to accurately disambiguate in the new domain, we also calculate the results for %A–A (and %B–B), that is, using only the tuning corpus for training.

Figure 3 graphically presents the results obtained by all methods. Each plot contains the X+%Y–Y and %Y–Y curves, and the straight lines corresponding to the lower bound MFC, and to the upper bounds Y–Y and X+Y–Y.

As expected, the accuracy of all methods grows (towards the upper bound) as more tuning corpus is added to the training set. However, the relation between X+%Y–Y and %Y–Y reveals some interesting facts. In plots 2a, 3a, and 1b the contribution of the original training corpus is null. Furthermore, in plots 1a, 2b, and 3b a degradation on the accuracy performance is observed. Summarizing, these six plots show that for Naive Bayes, Exemplar–based, and Snow methods it is not worth keeping the original training examples. Instead, a better (but disappointing) strategy would be simply using the tuning corpus.

[12] A Kappa value of 1 indicates perfect agreement, while 0.8 is considered as indicating good agreement (Carletta, 1996).

[13] ROMANSEVAL is, like SENSEVAL for English, a specific competition between WSD systems for Romance languages. See http://www.lpl.univ-aix.fr/projects/romanseval and http://www.itri.bton.ac.uk/events/senseval.

A–A

	DSO	MFC	NB	EB	Snow	LB
DSO	—	0.57	0.67	0.70	0.69	0.72
MFC	-0.17	—	0.84	0.68	0.73	0.67
NB	0.19	0.02	—	0.80	0.85	0.77
EB	0.37	-0.04	0.38	—	0.79	0.79
Snow	0.32	-0.06	0.51	0.48	—	0.77
LB	0.44	-0.14	0.32	0.48	0.43	—

A–B

	DSO	MFC	NB	EB	Snow	LB
DSO	—	0.37	0.41	0.45	0.44	0.47
MFC	-0.26	—	0.89	0.64	0.71	0.62
NB	-0.12	0.04	—	0.70	0.78	0.68
EB	0.10	-0.11	0.16	—	0.70	0.70
Snow	0.06	-0.08	0.24	0.34	—	0.68
LB	0.14	-0.11	0.14	0.39	0.33	—

B–B

	DSO	MFC	NB	EB	Snow	LB
DSO	—	0.47	0.57	0.58	0.57	0.59
MFC	-0.20	—	0.78	0.65	0.62	0.57
NB	0.15	-0.04	—	0.77	0.77	0.70
EB	0.24	-0.14	0.40	—	0.71	0.71
Snow	0.24	-0.15	0.41	0.37	—	0.69
LB	0.32	-0.17	0.30	0.43	0.39	—

B–A

	DSO	MFC	NB	EB	Snow	LB
DSO	—	0.38	0.46	0.50	0.49	0.51
MFC	-0.32	—	0.81	0.62	0.56	0.52
NB	-0.05	-0.00	—	0.71	0.69	0.63
EB	0.11	-0.18	0.18	—	0.65	0.66
Snow	0.13	-0.21	0.19	0.24	—	0.64
LB	0.18	-0.22	0.11	0.32	0.30	—

Figure 2: Kappa (κ) statistic (below diagonal) and agreement rate (above diagonal) between all methods in A–A, A–B, B–B and B–A experiments

However, this is not the situation of LazyBoosting (plots 4a and 4b), for which a moderate (but consistent) improvement of accuracy is observed when retaining the original training set. Therefore, LazyBoosting shows again a better behaviour than their competitors.

4.3 Third Experiment

The bad results about the portability of systems could be explained by, at least, two reasons: 1) Corpus A and B have a very different distribution of senses, and, therefore, different a–priori biases; 2) Examples of corpus A and B contain different information, and, therefore, the learning

Test on B corpus — Test on A corpus

Naive Bayes — Exemplar–based — Snow — LazyBoosting

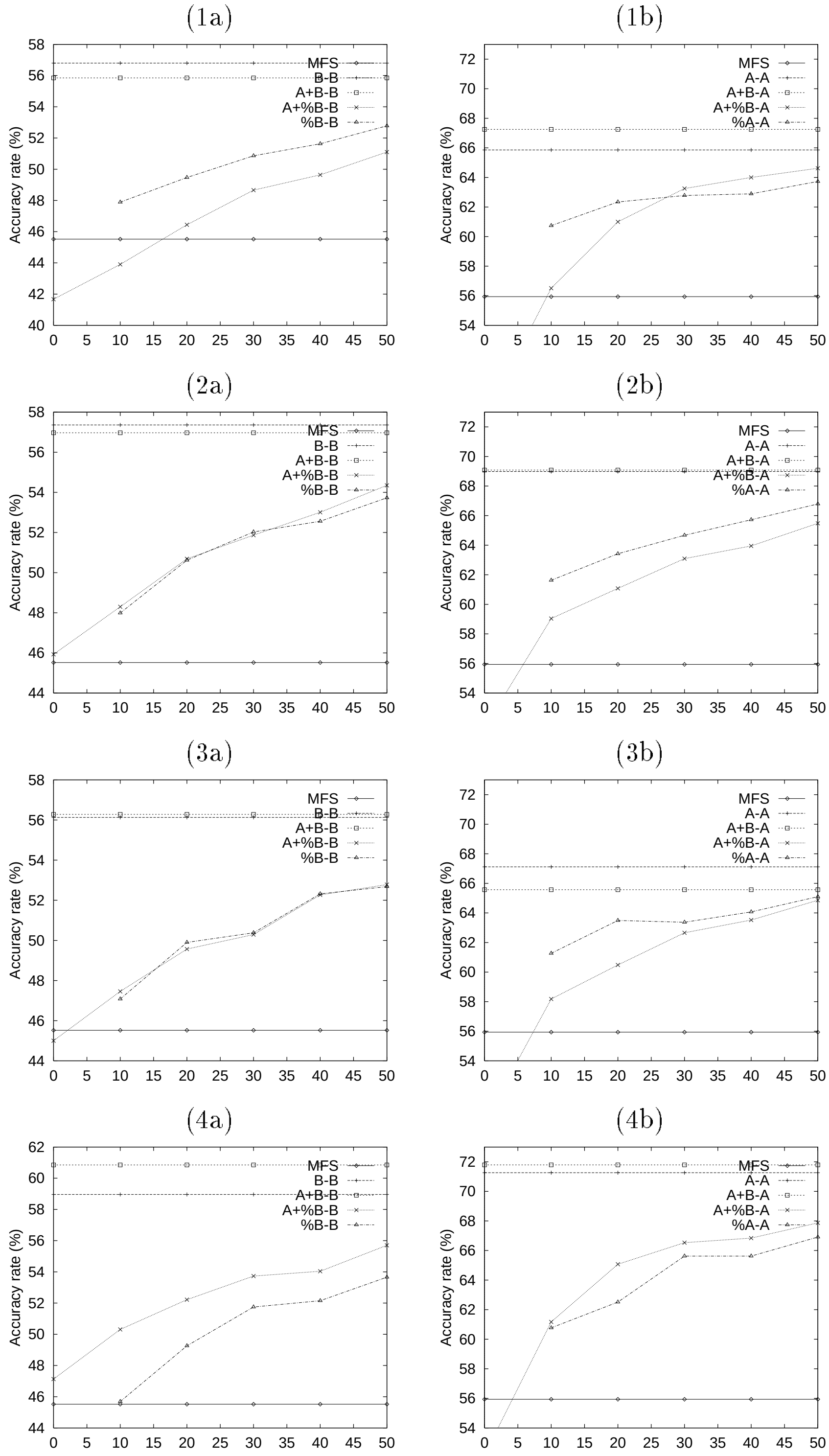

Figure 3: Results of the tuning experiment

algorithms acquire different (and non interchangeable) classification cues from both corpora.

The first hypothesis is confirmed by observing the bar plots of figure 4, which contain the distribution of the four most frequent senses of some sample words in the corpora A and B, respectively. As an example, first and second senses of noun *head* have a completely opposite distribution in both corpora A and B (first plot of figure 4).

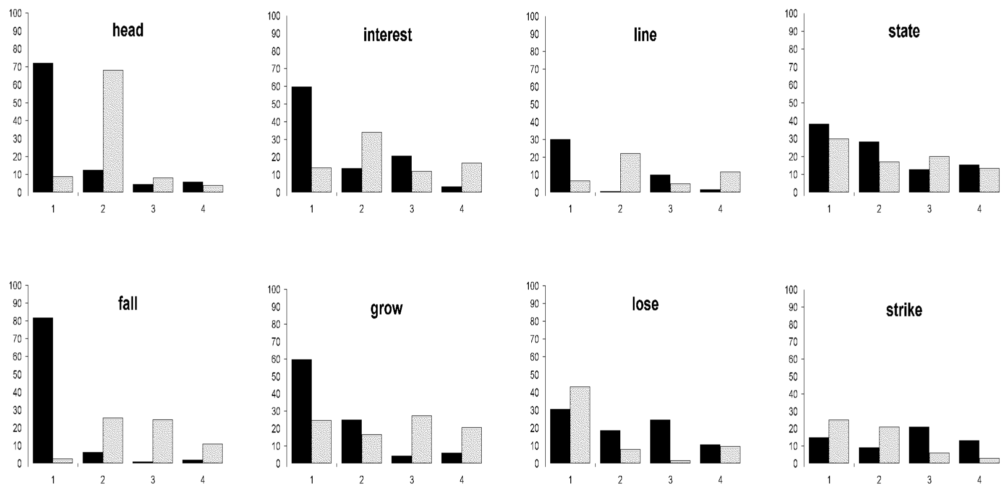

Figure 4: Distribution of the four most frequent senses for four nouns (head, interest, line, state) and four verbs (fall, grow, lose, strike). Black bars = A corpus; Grey bars = B corpus

In order to check the second hypothesis, two new sense–balanced corpora have been generated from the DSO corpus, by equilibrating the number of examples of each sense between A and B parts[14]. In this way, the first difficulty is artificially eliminated and the algorithms should be portable if examples of both parts were quite similar.

Table 5 shows the results obtained by LazyBoosting on these new corpora.

		Accuracy (%)						
Method	PoS	A+B–A+B	A+B–A	A+B–B	A–A	B–B	A–B	B–A
MFC	nouns	48.75	48.90	48.61	48.87	48.61	48.99	48.99
	verbs	48.22	48.22	48.22	48.22	48.22	48.22	48.22
	total	48.55	48.64	48.46	48.62	48.46	48.70	48.70
LB	nouns	62.82	64.26	61.38	63.19	60.65	53.45	55.27
	verbs	66.82	69.33	64.32	68.51	63.49	60.44	62.55
	total	64.35	66.20	62.50	65.22	61.74	56.12	58.05

Table 5: Results of LazyBoosting on the sense–balanced corpora

Regarding portability, we observe a significant accuracy decrease of 7 and 5 points from A–A to B–A, and from B–B to A–B, respectively[15]. That is, even when the same distribution of senses is conserved between training and test examples, the portability of the supervised WSD systems

[14]Again, this is achieved by reducing the size of the largest corpus to the size of the smallest.

[15]This difference in accuracy is not as important as in the first experiment, due to the simplification provided by the balancing of sense distributions.

is not guaranteed. As an extreme example, table 6 shows the performance of LB on the noun *state*. Note that in this case, LB performs even worse than MFC in the A–B and B–A columns.

	Accuracy (%)						
Method	A+B–A+B	A+B–A	A+B–B	A–A	B–B	A–B	B–A
MFC	39.00	39.00	39.00	39.00	39.00	39.00	39.00
LB	51.25	49.00	53.50	45.50	50.50	37.83	38.50

Table 6: Results of LazyBoosting on noun *state* (from the sense–balanced corpora).

These results imply that examples have to be largely different from one corpus to another. After the training process, LB provides as output an ordered set of rules based on the most relevant attributes. By studying the weak rules generated by LB from both corpora, we could observe the difference between them.

- The type of features used in the rules were significantly different between corpora, and, additionally, there were very few rules that apply to both sets. For corpus B, topical rules (i.e. those referring to topical context) seem to be more important than for corpus A. In the 50 firsts rules, corpus B has 30 topical rules and corpus A 20. Restricting to the 20 firsts rules, corpus B has 10 topical rules while corpus A has only 3. Not surprisingly, there are 34 of the 50 firsts rules learned from corpus A that do not appear in those learned from corpus B.

- The sign of the predictions of many of these common rules was somewhat contradictory between corpora. See, for instance, the two rules presented in table 7, acquired from corpus A and B respectively, which account for the collocation *state court*. For each rule and each sense there are two real numbers, which are the rule outputs when the feature holds in the example and when it does not, respectively. The sign of real numbers is interpreted as a prediction as to whether the sense should or should not be assigned to the example. The magnitude of the prediction is interpreted as a measure of confidence in the prediction. Therefore, according to the first rule, the presence of the collocation *state court* gives positive evidence to the sense number 3 and negative to the rest of senses. On the contrary, according to the second rule the presence of such collocation would contribute to sense number 5 and negatively to all the rest (including sense number 3).

			Senses					
		Feature	0	1	2	3	4	5
rule 1	A	*state court*	-0.5064	-1.0970	-0.6992	**1.2580**	-1.0125	-0.4721
			0.0035	0.0118	0.0066	-0.0421	0.0129	0.0070
rule 2	B	*state court*	-0.0696	-0.2988	-0.1476	-1.1580	-0.5095	**1.2326**
			-0.0213	0.0019	-0.0037	0.0102	0.0027	-0.0094

Table 7: Example of WeakRule: it is referred to the collocation *state court*.

It has to be noted that these rules are completely coherent with the senses assigned to the examples containing *state court* in both corpora, and therefore, the contradiction comes from the different information that characterize examples of both corpora.

5 Conclusions and Further Work

This work has pointed out some difficulties regarding the portability of supervised WSD systems, a very important issue that has been paid little attention up to the present. The main conclusion that can be extracted is that to assure the portability of systems, a process of tuning to the new domain is required (at least if the learning–testing corpora differ so as BC and WSJ do). This result is in contradiction with the idea of "robust broad-coverage WSD" introduced by (Ng, 1997b), in which a supervised system trained on a large enough corpora (say a thousand examples per word) should provide accurate disambiguation on any corpora (or, at least significantly better than MFS). Consequently, it is our belief that a number of issues regarding portability, tuning, knowledge acquisition, etc., should now be throrougly studied before stating that the supervised ML paradigm is able to resolve a realistic WSD problem.

Regarding the ML algorithms tested, the contribution of this work consist of empirically demonstrating that the LazyBoosting algorithm outperforms other three state-of-the-art supervised ML methods for WSD. Furthermore, this algorithm is proven to have better properties when is applied to new domains.

Further work is planned to be done in the following directions:

- Studying the problem of obtaining a representative enough training corpus, clarifying what is exactly meant by *enough* and *representative*. Note that in current approaches to WSD each word is treated as a different classification problem. Therefore, the collection of a representative corpus should take into account the particularities of each word in order to decide the number of examples needed to learn, the most useful attributes, and so on.
- Due to the fact that most of the knowledge learned from a domain is not applicable when changing to a new domain, further investigation is needed on tuning strategies, specially on those using non-supervised algorithms.
- Extensively evaluate LazyBoosting on the WSD task. This would include taking into account additional/alternative attributes and testing the algorithm in other corpora —specially on sense-tagged corpora automatically obtained from Internet using non-supervised methods (Leacock et al., 1998; Mihalcea and Moldovan, 1999).
- It is known that mislabelled examples resulting from annotation errors tend to be hard examples to classify correctly, and, therefore, tend to have large weights in the final distribution. This observation allows both to identify the noisy examples and use LazyBoosting as a way to improve data quality. Preliminary experiments have been already carried out in this direction on the DSO corpus.
- The inspection of the rules learned by LazyBoosting could provide evidence about similar behaviours of a–priori different senses. We think that this type of knowledge could be useful to perform clustering of too fine-grained or artificial senses.

Acknowledgments

This research has been partially funded by the Spanish Research Department (CICYT's BASURDE project TIC98–0423–C06) and by the Catalan Research Department (CIRIT's consolidated research group 1999SGR-150, CREL's Catalan WordNet project and CIRIT's grant 1999FI 00773).

References

D. Aha, D. Kibler, and M. Albert. 1991. Instance–based Learning Algorithms. *Machine Learning*, 7:37–66.

R. F. Bruce and J. M. Wiebe. 1999. Decomposable Modeling in Natural Language Processing. *Computational Linguistics*, 25(2):195–207.

J. Carletta. 1996. Assessing agreement on classification tasks: The kappa statistic. *Computational Linguistics*, 22(2):249–254.

J. Cohen. 1960. A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 22(2):249–254.

W. Daelemans, A. van den Bosch, and J. Zavrel. 1999. Forgetting Exceptions is Harmful in Language Learning. *Machine Learning*, 34:11–41.

I. Dagan, Y. Karov, and D. Roth. 1997. Mistake-Driven Learning in Text Categorization. In *Proceedings of the 2nd Conference on Empirical Methods in Natural Language Processing, EMNLP*, Brown University, Providence, RI, USA.

T. G. Dietterich. 1998. Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithms. *Neural Computation*, 10(7).

R. O. Duda and P. E. Hart. 1973. *Pattern Classification and Scene Analysis*. Wiley.

S. P. Engelson and I. Dagan. 1996. Minimizing Manual Annotation Cost in Supervised Training from Corpora. In E. Riloff S. Wermter and G. Scheler, editors, *Connectionist, Statistical an Symbolic Approaches to Learning for Natural Language Processing*, Lecture Notes in Artificial Intelligence, 1040. Springer.

G. Escudero, L. Màrquez, and G. Rigau. 2000a. Boosting Applied to Word Sense Disambiguation. In *To appear in Proceedings of the 12th European Conference on Machine Learning, ECML*, Barcelona, Spain.

G. Escudero, L. Màrquez, and G. Rigau. 2000b. Naive Bayes and Exemplar-Based Approaches to Word Sense Disambiguation Revisited. In *To appear in Proceedings of the 14th European Conference on Artificial Intelligence, ECAI*, Berlin, Germany.

A. Fujii, K. Inui, T. Tokunaga, and H. Tanaka. 1998. Selective Sampling for Example–based Word Sense Disambiguation. *Computational Linguistics*, 24(4):573–598.

W. Gale, K. W. Church, and D. Yarowsky. 1992. One Sense Per Discourse. In *Proceedings of the Speech and Natural Language Workshop*, pp. 233–237.

W. Gale, K. W. Church, and D. Yarowsky. 1993. A Method for Disambiguating Word Senses in a Large Corpus. *Computers and the Humanities*, 26:415–439.

A. R. Golding and D. Roth. 1999. A Winnow-Based Approach to Context-Sensitive Spelling Correction. *Machine Learning*, 34:107–130.

N. Ide and J. Véronis. 1998. Introduction to the Special Issue on Word Sense Disambiguation: The State of the Art. *Computational Linguistics*, 24(1):1–40.

A. Kilgarriff and J. Rosenzweig. to appear. English SENSEVAL: Report and Results. In *Proceedings of the 2nd International Conference on Language Resources and Evaluation, LREC*, Athens, Greece.

C. Leacock, M. Chodorow, and G. A. Miller. 1998. Using Corpus Statistics and WordNet Relations for Sense Identification. *Computational Linguistics*, 24(1):147–166.

N. Littlestone. 1988. Learning Quickly when Irrelevant Attributes Abound. *Machine Learning*, 2:285–318.

L. Màrquez, H. Rodríguez, J. Carmona, and J. Montolio. 1999. Improving POS Tagging Using Machine Learning Techniques. In *Proceedings of the joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora, EMNLP/VLC*.

R. Mihalcea and I. Moldovan. 1999. An Automatic Method for Generating Sense Tagged Corpora. In *Proceedings of the 16th National Conference on Artificial Intelligence, AAAI*.

G. A. Miller, R. Beckwith, C. Fellbaum, D. Gross, and K. Miller. 1990. Five Papers on WordNet. *Special Issue of International Journal of Lexicography*, 3(4).

R. J. Mooney. 1996. Comparative Experiments on Disambiguating Word Senses: An Illustration of the Role of Bias in Machine Learning. In *Proceedings of the 1st Conference on Empirical Methods in Natural Language Processing, EMNLP*.

H. T. Ng and H. B. Lee. 1996. Integrating Multiple Knowledge Sources to Disambiguate Word Sense: An Exemplar-based Approach. In *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics*. ACL.

H. T. Ng, C. Y. Lim, and S. K. Foo. 1999. A Case Study on Inter-Annotator Agreement for Word Sense Disambiguation. In *Proceedings of the ACL SIGLEX Workshop: Standardizing Lexical Resources*.

H. T. Ng. 1997a. Exemplar-Base Word Sense Disambiguation: Some Recent Improvements. In *Proceedings of the 2nd Conference on Empirical Methods in Natural Language Processing, EMNLP*.

H. T. Ng. 1997b. Getting Serious about Word Sense Disambiguation. In *Proceedings of the ACL SIGLEX Workshop: Standardizing Lexical Resources*.

T. Pedersen and R. Bruce. 1998. Knowledge Lean Word-Sense Disambiguation. In *Proceedings of the 15th National Conference on Artificial Intelligence, AAAI*.

D. Roth and D. Zelenko. 1998. Part of Speech Tagging Using a Network of Linear Separators. In *Proceedings of the joint 17th International Conference on Computational Linguistics and 36th Annual Meeting of the Association for Computational Linguistics, COLING-ACL*, pp. 1136–1142, Montréal, Canada.

D. Roth. 1998. Learning to Resolve Natural Language Ambiguities: A Unified Approach. In *Proceedings of the National Conference on Artificial Intelligence, AAAI*.

R. E. Schapire and Y. Singe. to appear. Improved Boosting Algorithms Using Confidence-rated Predictions. *Machine Learning*.

S. Sekine. 1997. The Domain Dependence of Parsing. In *Proceedings of the 5th ACL Conference on Applied Natural Language Processing, ANLP*, Washington DC, USA.

G. Towell and E. M. Voorhees. 1998. Disambiguating Highly Ambiguous Words. *Computational Linguistics*, 24(1):125–146.

J. Véronis. 1998. A study of polysemy judgements and inter–annotator agreement. In *Programme and advanced papers of the Senseval workshop*, Herstmonceux Castle, England.

D. Yarowsky. 1994. Decision Lists for Lexical Ambiguity Resolution: Application to Accent Restoration in Spanish and French. In *Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics, ACL*, pp. 88–95, Las Cruces, NM, USA.