An Empirical Study of the Domain Dependence of Supervised Word Sense Disambiguation Systems*

Gerard Escudero, Lluís Màrquez, and **German Rigau**
TALP Research Center. LSI Department. Universitat Politècnica de Catalunya (UPC)
Jordi Girona Salgado 1–3. E-08034 Barcelona. Catalonia
{escudero,lluism,g.rigau}@lsi.upc.es

Abstract

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

Keywords: Cross–corpus evaluation of NLP systems, Word Sense Disambiguation, Supervised Machine Learning

1 Introduction

Word Sense Disambiguation (WSD) is the problem of assigning the appropriate meaning (sense) to a given word in a text or discourse. Resolving the ambiguity of words is a central problem for large scale language understanding applications and their associate tasks (Ide and Véronis, 1998), e.g., machine translation, information retrieval, reference resolution, parsing, etc.

WSD is one of the most important open problems in NLP. Despite the wide range of approaches investigated and the large effort devoted to tackle this problem, to date, no large-scale broad-coverage and highly accurate WSD system has been built —see the main conclusions of the first edition of SensEval (Kilgarriff and Rosenzweig, 2000).

One of the most successful current lines of research is the corpus-based approach in which statistical or Machine Learning (ML) algorithms are applied to learn statistical models or classifiers from corpora in order to perform WSD. Generally, supervised approaches[1] 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, such as: Decision Lists (Yarowsky, 1994; Agirre and Martinez, 2000), Neural Networks (Towell and Voorhees, 1998), Bayesian learning (Bruce and Wiebe, 1999), Exemplar-Based learning (Ng, 1997a; Fujii et al., 1998), Boosting (Escudero et al., 2000a), etc. Unfortunately, there have been very few direct comparisons between alternative methods for WSD.

In general, supervised learning presumes that the training examples are somehow reflective of the task that will be performed by the trainee on other data. Consequently, the performance of such systems is commonly estimated by testing the algorithm on a separate part of the set of training examples (say 10–20% of them), 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 (contrary to the opinion of other authors (Ng, 1997b)): On the

* This research has been partially funded by the Spanish Research Department (CICYT's project TIC98–0423–C06), by the EU Commission (NAMIC IST-1999-12392), and by the Catalan Research Department (CIRIT's consolidated research group 1999SGR-150 and CIRIT's grant 1999FI 00773).

[1] Supervised approaches, also known as *data–driven* or *corpus–driven*, are those that learn from a previously semantically annotated corpus.

one hand, WSD is very dependant to the domain of application (Gale et al., 1992b) —see also (Ng and Lee, 1996; Ng, 1997a), in which quite different accuracy figures are obtained when testing an exemplar–based WSD classifier on two different corpora. On the other hand, it does not seem reasonable to think that the training material is large and representative enough to cover "all" potential types of examples.

To date, a thorough study of the domain dependence of WSD —in the style of other studies devoted to parsing (Sekine, 1997)— has not been carried out. We think that such an study 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 paper explores the portability and tuning of four different ML algorithms (previously applied to WSD) by training and testing them on different corpora.

Additionally, supervised methods suffer from the "knowledge acquisition bottleneck" (Gale et al., 1992a). (Ng, 1997b) estimates that the manual annotation effort necessary to build a broad coverage semantically annotated English corpus is about 16 person–years. This overhead for supervision could be much greater if a costly tuning procedure is required before applying any existing system to each new domain.

Due to this fact, recent works have focused on reducing the acquisition cost as well as the need for supervision in corpus-based methods. It is our belief that the research by (Leacock et al., 1998; Mihalcea and Moldovan, 1999)[2] provide 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.

It is important to note that the focus of this work will be on the empirical cross–corpus evaluation of several ML supervised algorithms. Other important issues, such as: selecting the best attribute set, discussing an appropriate definition of senses for the task, etc., are not addressed in this paper.

[2] In the line of using lexical resources and search engines to automatically collect training examples from large text collections or Internet.

This paper is organized as follows: Section 2 presents the four ML algorithms compared. In section 3 the setting is presented in detail, 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

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.

Model probabilities are estimated during training process using relative frequencies. To avoid the effect of zero counts when estimating probabilities, a very simple smoothing technique has been used, which was proposed in (Ng, 1997a). 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 kept in memory and the classification of a new example is based on a k–NN (Nearest–Neighbours) algorithm using Hamming distance[3] 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,

[3] Although the use of MVDM metric (Cost and Salzberg, 1993) could lead to better results, current implementations have prohivitive computational overheads (Escudero et al., 2000b).

5, 7, 10, 15, 20 and 25) and the results corresponding to the best choice are reported[4].

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 stands for Sparse Network Of Winnows, and it is intended as a representative of on–line learning algorithms.

The basic component is the Winnow algorithm (Littlestone, 1988). It consists of a linear threshold algorithm with multiplicative weight updating for 2-class problems, which learns very fast in the presence of many binary input features.

In the Snow architecture there is a winnow node for each class, which learns to separate that class from all the rest. During training, each example is considered a positive example for winnow node associated to its class and a negative example for all 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. 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.

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), e.g: text categorization, context–sensitive spelling correction, WSD, etc.

In this paper, our approach to WSD using Snow follows that of (Escudero et al., 2000c).

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 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 real AdaBoost.MH algorithm for multi-class multi–label classification (Schapire and Singer, 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 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 (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, thus, the risk of eliminating relevant attributes is avoided. 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 the WSD 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, 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)[5].

[4] In order to construct a real EB–based system for WSD, the k parameter should be estimated by cross–validation using only the training set (Ng, 1997a), however, in our case, this cross–validation inside the cross–validation involved in the testing process would generate a prohibitive overhead.

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

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 them on the WSJ part and testing them 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[6] 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 1 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.

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 $w_{\pm i}$. 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}), 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[7].

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 attributes is not appropriate for NB and EB algorithms. Therefore, the 15 local-context attributes are taken straightforwardly. Regarding the binary topical–context attributes, we have used the variants described in (Escudero et al., 2000b). For EB, the topical information is codified as a single set–valued attribute (containing all words appearing in the sentence) and the calculation of closeness is modified so as to handle this type of attribute. For NB, the topical context is conserved as binary features, but when classifying new examples only the information of words appearing in the example (positive information) is taken into account. In that paper, these variants are called *positive* Exemplar–based (PEB) and *positive* Naive Bayes (PNB), respectively. PNB and PEB algorithms are empirically proven to perform much better in terms of accuracy and efficiency in the WSD task.

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

[6] This is achieved by ramdomly reducing the size of the largest corpus to the size of the smallest.

[7] 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.

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

Table 1: Information about the set of 21 words of reference.

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).

4 Experiments

4.1 First Experiment

Table 2 shows the accuracy figures of the four methods in all combinations of training and test sets[8]. Standard deviation numbers are supplied in all cases involving cross validation. 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.

The following conclusions can be drawn:

- LB outperforms all other methods in all cases. Additionally, this superiority is statistically significant, except when comparing LB to the PEB approach in the cases marked with an asterisk.
- 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 addition of extra examples from another domain does not necessarily contribute to improve the results on the original corpus. 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.
- 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 drop 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 drop from 71.3% (A–A) to 52.0% (B–A), which is lower than the most frequent sense of corpus A, 55.9%.

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 some kind of tuning process is necessary to adapt supervised systems to each new domain.

This experiment explores the effect of a simple 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 1 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,

[8]The second and third column correspond to the train and test sets used by (Ng and Lee, 1996; Ng, 1997a)

		Accuracy (%)						
		A+B–A+B	A+B–A	A+B–B	A–A	B–B	A–B	B–A
MFC	nouns	46.59±1.08	56.68±2.79	36.49±2.41	59.77±1.44	45.28±1.81	33.97	39.46
	verbs	46.49±1.37	48.74±1.98	44.23±2.67	48.85±2.09	45.96±2.60	40.91	37.31
	total	46.55±0.71	53.90±2.01	39.21±1.90	55.94±1.10	45.52±1.27	36.40	38.71
PNB	nouns	62.29±1.25	68.89±0.93	55.69±1.94	66.93±1.44	56.17±1.60	36.62	45.99
	verbs	60.18±1.64	64.21±2.26	56.14±2.79	63.87±1.80	57.97±2.86	50.20	50.75
	total	61.55±1.04	67.25±1.07	55.85±1.81	65.86±1.11	56.80±1.12	41.38	47.66
PEB	nouns	62.66±0.87	69.45±1.51	56.09±1.12	69.38±1.24	56.17±1.80	42.15	50.53
	verbs	63.67±1.94	68.39±3.25	58.58±2.40	68.25±2.84	59.57±2.86	51.19	52.24
	total	63.01±0.93	69.08±1.66	56.97±1.22	68.98±1.06	57.36±1.68	45.32	51.13
Snow	nouns	61.24±1.14	66.36±1.57	56.11±1.45	68.85±1.36	56.55±1.31	42.13	49.96
	verbs	60.35±1.57	64.11±2.76	56.58±2.45	63.91±1.51	55.36±3.27	47.66	49.39
	total	60.92±1.09	65.57±1.33	56.28±1.10	67.12±1.16	56.13±1.23	44.07	49.76
LB	nouns	**66.00**±1.47	**72.09**±1.61	**59.92**±1.93	**71.69**±1.54	**58.33**±2.26	**43.92**	**51.28***
	verbs	**66.91**±2.25	**71.23**±2.99	**62.58**±2.93	**70.45**±2.14*	**60.14**±3.43*	**52.99**	**53.29***
	total	**66.32**±1.34	**71.79**±1.51	**60.85**±1.81	**71.26**±1.15	**58.96**±1.86	**47.10**	**51.99***

Table 2: Accuracy results (± standard deviation) of the methods on all training–test combinations

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.

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 when moving from one domain to another.

4.3 Third Experiment

The bad results about portability 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 algorithms acquire different (and non interchangeable) classification cues from both corpora.

The first hypothesis is confirmed by observing the bar plots of figure 2, which contain the distribution of the four most frequent senses of some sample words in the corpora A and B, respectively. 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. In this way, the first difficulty is artificially overrided and the algorithms should be portable if examples of both parts are quite similar.

Table 3 shows the results obtained by LazyBoosting on these new 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[9]. That is, even when the same distribution of senses is conserved between training and test examples, the portability of the supervised WSD systems is not guaranteed.

These results imply that examples have to be largely different from one corpus to another. By studying the weak rules generated by LazyBoosting in both cases, we could corroborate this fact. On the one hand, 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; On the other hand, the sign of the prediction of many of these common rules was somewhat contradictory between corpora.

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

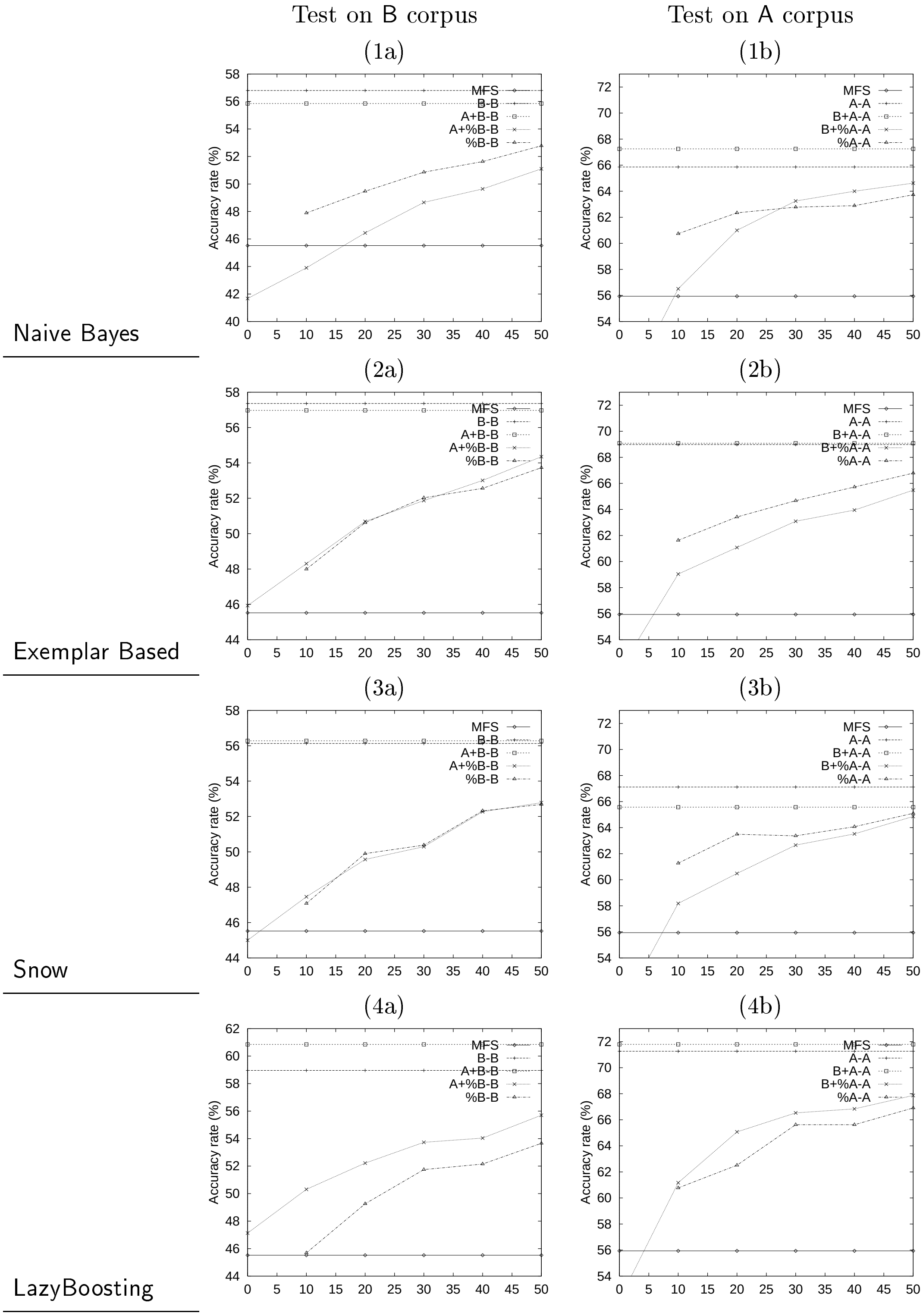

Figure 1: Results of the tuning experiment

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.

According to our experiments, it seems that the performance of supervised sense taggers is not guaranteed when moving from one domain to another (e.g. from a balanced corpus, such as BC, to an economic domain, such as WSJ). These results implies that some kind of adaptation is required for cross–corpus application.

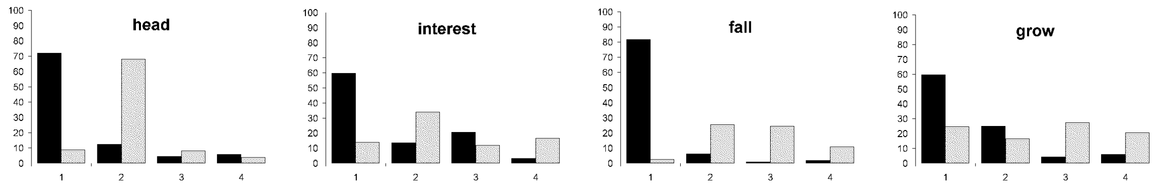

Figure 2: Distribution of the four most frequent senses for two nouns (head, interest) and two verbs (line, state). Black bars = A corpus; Grey bars = B corpus

		Accuracy (%)						
		A+B–A+B	A+B–A	A+B–B	A–A	B–B	A–B	B–A
MFC	nouns	48.75±0.91	48.90±1.69	48.61±0.96	48.87±1.68	48.61±0.96	48.99	48.99
	verbs	48.22±1.68	48.22±1.90	48.22±3.06	48.22±1.90	48.22±3.06	48.22	48.22
	total	48.55±1.16	48.64±1.04	48.46±1.21	48.62±1.09	48.46±1.21	48.70	48.70
LB	nouns	62.82±1.43	64.26±2.07	61.38±2.08	63.19±1.65	60.65±1.01	53.45	55.27
	verbs	66.82±1.53	69.33±2.92	64.32±3.27	68.51±2.45	63.49±2.27	60.44	62.55
	total	64.35±1.16	66.20±2.12	62.50±1.47	65.22±1.50	61.74±1.18	56.12	58.05

Table 3: Accuracy results (± standard deviation) of LazyBoosting on the sense–balanced corpora

Furthermore, these results are 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 be thoroughly 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:

- 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 or large text collections using non-supervised methods (Leacock et al., 1998; Mihalcea and Moldovan, 1999).
- Since most of the knowledge learned from a domain is not useful when changing to a new domain, further investigation is needed on tuning strategies, specially on those using non-supervised algorithms.
- 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.
- Moreover, the inspection of the rules learned by LazyBoosting could provide evidence about similar behaviours of a–priori different senses. This type of knowledge could be useful to perform clustering of too fine-grained or artificial senses.

References

E. Agirre and D. Martinez. 2000. Decision Lists and Automatic Word Sense Disambiguation. In *Proceedings of the COLING Workshop on Semantic Annotation and Intelligent Content*

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.

S. Cost and S. Salzberg. 1993. A weighted nearest neighbor algorithm for learning with symbolic features. *Machine Learning*, **10**(1), 57–78.

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

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.

G. Escudero, L. Màrquez, and G. Rigau. 2000a. Boosting Applied to Word Sense Disambiguation. 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*.

G. Escudero, L. Màrquez, and G. Rigau. 2000c. On the Portability and Tuning of Supervised Word Sense Disambiguation Systems. Research Report LSI-00-30-R, Software Department (LSI). Technical University of Catalonia (UPC).

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. 1992a. A Method for Disambiguating Word Senses in a Large Corpus. *Computers and the Humanities*, 26:415–439.

W. Gale, K. W. Church, and D. Yarowsky. 1992b. Estimating Upper and Lower Bounds on the Performance of Word Sense Disambiguation. In *Proceedings of the 30th Annual Meeting of the Association for Computational Linguistics*. ACL.

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. 2000. 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.

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 Press.

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. 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: Tagging Text with Lexical Semantics: Why, what and how?*, Washington, USA.

D. Roth. 1998. Learning to Resolve Natural Language Ambiguities: A Unified Approach. In *Proceedings of the National Conference on Artificial Intelligence, AAAI '98*, July.

R. E. Schapire and Y. Singer. to appear. Improved Boosting Algorithms Using Confidence-rated Predictions. *Machine Learning*. Also appearing in *Proceedings of the 11th Annual Conference on Computational Learning Theory*, 1998.

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

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

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*, pages 88–95, Las Cruces, NM. ACL.