Identifying Word Senses in Greek Text: A comparison of machine learning methods

A. Grigoriadis, G. Paliouras, V. Karkaletsis, C. D. Spyropoulos

Software and Knowledge Engineering Laboratory,
Institute of Informatics and Telecommunications, N.C.S.R. "Demokritos",
P.O. BOX 60228, Aghia Paraskevi, 153 10 Athens, Greece.
E-mail: {grigori, paliourg, vangelis, costass}@iit.demokritos.gr

Abstract - In this paper we perform a comparative evaluation of machine learning methods on the task of identifying the correct sense of a word, based on the context in which it appears. This task is known as word sense disambiguation (WSD) and is one of the hardest and most interesting issues in language engineering. Research on the use of machine learning techniques for WSD has so far focused almost exclusively on English words, due to the scarcity of the required linguistic resources for other languages. The work presented here is the first attempt to apply machine learning methods to Greek words. We have constructed a semantically tagged corpus for two Greek words: a noun with clearly distinguishable senses and a verb with overlapping senses. This corpus is used to evaluate four different machine learning methods and three different representations of the context of the ambiguous word. Our results show that the simple naïve Bayesian classifier and a method using Support Vector Machines outperform decision tree induction, even with the use of boosting. Furthermore, the use of a distance-based weighting function for the context of the ambiguous word does not seem to have a substantial effect on the performance of the methods.

Keywords: machine learning, word sense disambiguation, computational linguistics.

1. Introduction

In common dictionaries all words are defined by a list of different senses, which aim to describe a spectrum of possible interpretations for the word. The different senses may be similar in meaning and come from the same root, or be completely separate words that are just spelled the same (*homonymous*). In either case, a human will seldom be confused by the ambiguous nature of the words. An important goal of computational linguistics, is to teach this ability to machines, in other words to train computers so as to distinguish the proper meaning of each word, based on the context. A sufficient solution to this problem would answer several hard issues in language engineering, such as machine translation, information retrieval, content and thematic analysis, grammatical analysis, speech processing and text processing [6].

In this study, our aim is to deal with this difficult problem for the Greek language, by a comparative evaluation of the performance of various machine learning methods. For this purpose, we have constructed a semantically tagged corpus for two Greek words: a noun with clearly distinguishable senses and a verb with overlapping senses. This corpus is used to evaluate four different machine learning methods and three different representations of the context of the ambiguous word. Using ten-fold cross-validation, we estimate the performance of the disambiguators generated by each of the four learning methods for each of the three representations.

In section 2 we present the basic characteristics of various disambiguation approaches that have been used so far. In section 3, we concentrate on the data chosen for our experiments and the procedure of transforming them into proper training input for the learning methods. Section 4 is a quick review of the machine learning algorithms that we used. Finally, we present our results, in section 5, while the last section is devoted to conclusions and further work.

2. Related Work

Early work in computational linguistics focused on constructing computers that would be able to speak and understand the human's natural language. Therefore, there were many efforts to simulate the pattern of human knowledge acquisition with a computer program. However, these attempts achieved limited success, due to the 'knowledge acquisition bottleneck': there is too much information and no way to transform it into proper input for the system.

When the idea of representing accurately human knowledge was abandoned, more empirical methods came along, using a new powerful utility: Machine Readable Dictionaries (MRD). Computers were able to read word definitions in a dictionary and utilize this information in various ways. A typical example is the estimation of overlap between the dictionary definition of a word and its surrounding text, for the purpose of disambiguation [10]. The definition with the highest overlap can be assumed to be the correct one.

The next step, and most typical current approach to the WSD task, was the combination of machine learning methods with information extracted from large corpora. Machine learning is a means of using a large amount of information as

training data, in order to increase a machine's performance on a specific task. There are various machine learning algorithms, which adopt different internal structure and generalization approach but they all aim to train the system based on experience. The recent availability of large corpora provided the necessary material for training machine learning algorithms. Corpora are collections of texts, either general or of a specific field, many of which contain extra information (meta-data) for training purposes. Most useful for WSD are texts with pre-tagged labels for the senses of the words they contain.

The most common approach to WSD with the use of machine learning is as a classification task. For each polysemous word, the classes represent the different senses and the classifier must choose among them. Each appearance of the word in a text can form a *case*, whose *attributes* are the words of the context and some of their properties. A set of cases, which can be extracted from a corpus, is used to train the classifier, so as to classify new unseen examples.

This classification approach requires *supervised learning*, since all the senses of the words must be known prior to the training procedure. The sense labels are typically taken from a dictionary. Examples of supervised learning methods for WSD appear in [1], [5], [9], [14], [17]. Although supervised methods have a high success rate, they are usually word-specific, which means that every classifier can disambiguate one word only. Moreover, they require pre-tagged corpora, which are not always available in every language. On the other hand, *unsupervised learning* methods perform a *clustering* of the training instances, according to latent senses of the polysemous word that are not pre-specified. Unsupervised learning methods do not require hand-tagging of the training data, but their performance is difficult to assess [8], [13], [18].

Another important issue in WSD is the selection of features to be used in the construction of disambiguation rules. Usually, we are mostly interested in the words that surround the ambiguous word. A small neighborhood of up to ten words on each side forms a local context group, while a wider window of 100 words or more is a topical context group. Although, most methods focus on one of the two types of context group, some try to combine both local and topical context, such as those in [14], [19]. Apart from the words themselves we can add further information, if it is considered useful for the disambiguation task. This information can be the distance from the polysemous word, phrasal collocations, syntactic relations, semantic categories etc.

Finally, while most methods so far deal with *restricted-vocabulary disambiguation*, that is they concentrate on a handful of polysemous words, what would be ideal for language engineering is a *large-vocabulary disambiguation* system, for all the content words of a text (nouns, verbs, adjectives and adverbs). Exemplary large-vocabulary WSD methods are presented in [11], [15].

The work presented here is primarily an attempt to evaluate the performance of machine learning methods on the disambiguation of Greek words. The fact that similar work has not been done for the Greek language before and the lack of large semantically annotated corpora has led us to adopt the supervised approach to learning restricted-vocabulary sense disambiguation models.

3. The WSD Data

In this study we concentrated on the task of disambiguating two polysemous Greek words: the verb 'διακρίνω' "dia-kri-no" (distinguish, separate) and the noun 'τάξη' "ta-ksi" (class, order). Each word instance was assigned to a sense class, according to the "Triantafyllidis Dictionary of Contemporary Greek".[1] Four different senses were used for the word 'διακρίνω' and three senses for the word 'τάξη'. The English interpretation of the senses is shown in table 1.

'διακρίνω' (dia-kri-no)	**'τάξη' (ta-ksi)**
separate	order
distinguish	class
descry	classroom
make out	

Table 1 - Word Senses

A number of occurrences of these words in text are necessary in order to get the necessary training data for the algorithms. Due to the lack of pre-tagged corpora for the Greek language, we had to collect the texts and tag them manually. The texts come from various sources and have different subjects and writing styles. The main source was newspaper archives, publicly available through the Internet, but also personal pages and commercial presentations.

[1] http://ermis.komvos.edu.gr/Lexika/Triantafyllidis/Triantafyllidis.htm

Altogether 519 examples for the word 'διακρίνω' (dia-kri-no) and 319 for the word 'τάξη' (ta-ksi) were collected, each one in the context of 50 words to the left and 50 words to the right of the polysemous word. The assignment of sense classes was done by a group of four people, with a background in language engineering. Each human annotator assigned a class to each word instance and then their opinions were compared. The four annotators were in complete agreement for the word 'τάξη' (ta-ksi), but the same was not the case for 'διακρίνω' (dia-kri-no). In order to assess agreement, we calculated the kappa coefficient (K) [2], which is an indicator of how different the results are from random and whether or not the data produced are too noisy to use for the purpose for which they were collected.

Figure 1 shows the distribution of sense labels for the four annotators for the word 'διακρίνω' (dia-kri-no). Only 80.2% of the instances were classified the same by the annotators. The kappa coefficient had a borderline value of 0.8, which barely qualifies the data for further use. The high disagreement of the human annotators illustrates the high overlap of the four senses for the word 'διακρίνω' (dia-kri-no) and as a result the difficulty of disambiguating the word. A common consensus was reached by the four annotators for the cases where there were disagreements, arriving at the final tagged corpus.

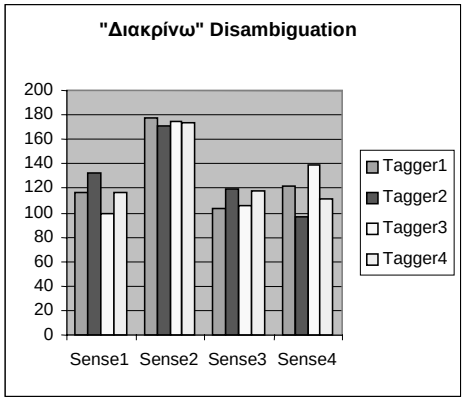


Figure 1 - Tagging Results

Once the training corpus was tagged, a decision had to be made on the representation of the data that would be more suitable for the purpose of disambiguation. Since we consider disambiguation as a classification task, we chose the feature vector format. Based on a set of representative features, each instance (appearance of a polysemous word in text) takes the form of a vector with the right values on the feature fields. The set of vectors make the training set, which will be used to train the classifier.

Choosing the right representation is essential to the disambiguation performance. In the first experiment we chose the most common, 'bag-of-words' representation for the data. Each word surrounding the polysemous word is as important as any other and all 100 words are taken into consideration. Thus, the feature vector consists of a list of the words found in the texts and a binary value is assigned, depending on whether the word appears in an instance or not. Since the total number of words is very large, we apply feature selection choosing the 100 most important words of the vocabulary, according to the information gain metric [12].

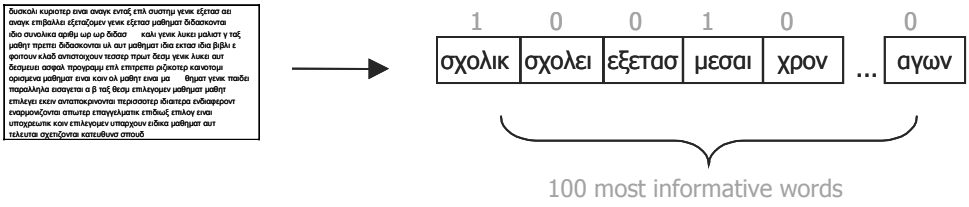


Figure 2 - Forming feature vectors

In Figure 2 we can see how an instance is transformed into a vector, by assigning one bit on the 100 most informative words. This bit denotes the existence or not of the specific word in the context of the instance.

In a second approach, we attempted to discriminate among the surrounding words according to their distance from the polysemous word. So, the nearby words were accredited more weight than the distant ones. The weight distribution can be controlled by a function chosen for this purpose. A square function will leave out the distant words and concentrate on the local context. More interesting are the results of a triangular or a logarithmic function.

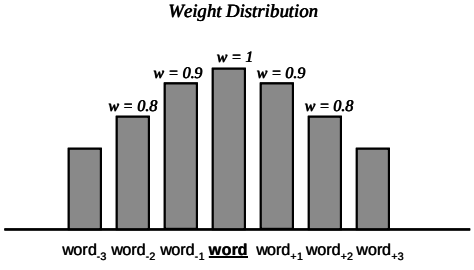


Figure 3 - Weight Distribution

Figure 3 shows how the weights are distributed to the words that surround the polysemous word. In this example, a triangular function is selected.

The same feature vector representation is used again, although this time the feature values are not binary, but real numbers in the range [0..1], and they represent not only the existence of the word but its significance as well, according to the distance-based weighting function.

4. Learning Methods

In this study we performed a systematic comparison of four different machine learning methods, on the task of disambiguating the selected words: Decision Tree Induction, Boosted Decision Trees, Naive Bayesian Classifier, and Support Vector Machines. For each experiment, we used 10-fold cross-validation, in order to obtain an unbiased estimate of the performance of each method on unseen data. The performance of the algorithms varies significantly, according to the nature and the representation of the problem. Thus, our aim is not to evaluate the quality of the algorithms as such, but to examine which of them are suitable to the WSD task.

Decision tree induction (J48). The decision tree induction algorithm used here is called J48 and it is an improved version of the C4.5 algorithm [12]. Given example cases in the format described in section 3, J48 constructs a decision tree, which can then be used to assign sense tags to unseen data. J48 generates decision trees, the nodes of which evaluate the existence, or significance, of words in the context. Following a path from the root to the leaves of the tree a sequence of such tests is performed, resulting in a decision about the appropriate sense for the word. The decision trees are constructed in a top-down fashion, by choosing the most appropriate feature each time. The features are evaluated according to an information-theoretic measure, which provides an indication of the "classification power" of each feature. Once a feature is chosen, the training data are divided into subsets, corresponding to different values of the selected feature, and the process is repeated for each subset, until a large proportion of the instances in each subset belong in a single class.

Boosted decision trees. Boosting is a technique for improving the performance of machine learning algorithms by constructing various classification or prediction models and combining them through a voting process. We used the AdaBoost method [4], which is designed specifically for classification methods, and we have combined it with decision-tree classifiers. This method attempts to diversify the generated classifiers, by training each new classifier on examples that previous classifiers have found difficult to classify. Each individual classifier is a small decision tree and contributes to the final classification of each instance, according to its performance.

Naive Bayesian classification. The Naïve Bayesian [3] is arguably one of the simplest probabilistic classifiers. The model constructed by this algorithm is a set of probabilities. Each member of this set corresponds to the probability that a specific feature value f_i appears in the instances of class c, i.e., $P(f_i|c)$. These probabilities are estimated by counting the frequency of each feature value in the instances of a class in the training set.[2] Given a new instance the classifier estimates the probability that the instance belongs to a specific class, based on the product of the individual conditional probabilities for the feature values in the instance. The exact calculation uses Bayes theorem and this is the reason why the algorithm is called a Bayesian classifier. The algorithm is also characterized as Naïve, because it makes the assumption that the features are independent given the class, which is rarely the case in real-world problems. Despite this strong assumption, the algorithm performs surprisingly well on a range of tasks.

Support Vector Machines. SVM is an interesting approach to classification, which combines linear regression with instance-based learning. If we consider every instance with n features as a vector in a n-dimensional space and we want to classify all instances into two classes, then we seek for an optimal hyperplane which would divide these instances. Support vectors are the closest vectors on each side of the hyperplane and we want to maximize their distance from it. This is the basic approach, but we have to deal with two problems. Firstly, for some problems, improved classification results can be obtained using a nonlinear classifier. For these cases, we apply a transformation to the input, using a nonlinear mapping. This is done practically by choosing the right kernel, which would substitute the definition of inner product, in all the necessary calculations. Typical kernel functions are polynomial or radial basis. The second problem is that the SVM is a binary classifier, i.e., it can separate the instances into only two classes. In order to deal with multiple classes, we construct a different classifier for each class (or pair of classes) and we combine the results. Every classifier votes for a sense-class and the class that heads the poll is the most likely to be correct. SVM is a flexible method, with good results in problems such as text categorization [7]. It is also resistant to noise and avoids overfitting.

The implementations of the four methods that we used are those found in the WEKA platform (Waikato Environment for Knowledge Analysis), which is publicly available from University of Waikato.[3]

5. Experimental Results

In all of the figures presented in this section, the performance of the system is measured on unseen data. Furthermore, in order to arrive at a robust estimate of performance, we use ten-fold cross-validation at each individual experiment. According to this experimental method, the data set is divided into ten equally-sized pieces and the learning algorithm is

[2] In the case of numeric features, feature values are usually assumed to follow a "normal" distribution.

[3] http://ww.cs.waikato.ac.nz/ml.weka

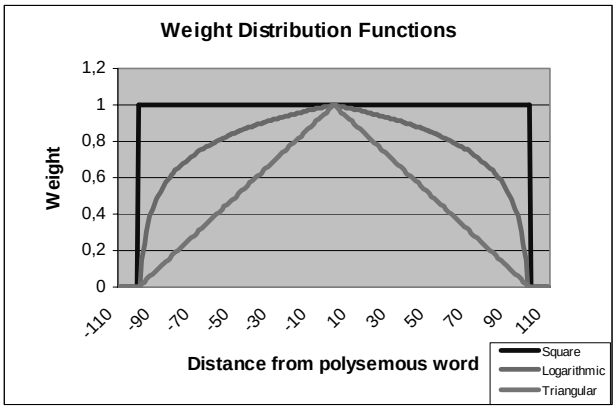
run ten times. Each time, nine of the ten pieces are used for training and the tenth is kept as unseen data for the evaluation of the algorithm. Each of the ten pieces acts as the evaluation set in one of the ten runs. Thus, each success rate presented in the following sections is an average over ten runs, rather than a single train-and-test result, which can often be accidentally high or low. Finally, it should be noted that, unlike MRD-based approaches, the constructed disambiguators make no use of external resources.

Table 2 presents the results of the first experiment, using the bag-of-words representation for all learning methods examined in this study. Support vector machines achieved the best performance for the word 'τάξη' (ta-ksi) and Naive Bayesian for the word 'διακρίνω' (dia-kri-no). An interesting result is the difference in the performance of all methods for the two words. Indeed all the algorithms had a success rate reduced by 20% for the word 'διακρίνω' (dia-kri-no). This implies that the difficulties faced by human annotators when disambiguating instances of 'διακρίνω' (dia-kri-no) carry over to the automated classifiers.

Algorithm	'τάξη' (ta-ksi)	'διακρίνω' (dia-kri-no)
Decision Tree Induction	73.87 %	54.26 %
Boosted Decision Trees	76.45 %	52.91 %
Naive Bayesian Classifier	78.06 %	61.82 %
Support Vector Machines	80 %	59.88 %
Baseline	35.81 %	32.75 %

Table 2 – Results for the bag-of-words representation.

Decision trees did particularly poorly, classifying correctly only 73.78% of the instances for the word 'τάξη' (ta-ksi) and 54.26% for the word 'διακρίνω' (dia-kri-no). By combining boosting with decision trees, the success rate for the word 'τάξη' (ta-ksi) increased to 76.45%, but the same did not happen for 'διακρίνω' (dia-kri-no) (52.91%). Naive Bayesian presented impressive results for both cases, 78.06% for 'τάξη' (ta-ksi) and 61.82% for 'διακρίνω' (dia-kri-no), despite its simplicity. Its high performance suggests that the dependencies in the appearance of some words, i.e., the fact that they are often used together in text, does not cause significant problems, especially, when using information gain to select the most informative features first. Support Vector Machines also performed very well for both cases, (80% and 59.88%) and, although never used for WSD before, may be the most promising method.



[G.P.1]On the second experiment we attributed different weights to the surrounding words, according to a triangular and a logarithmic function. It is assumed that the words near the polysemous word are more important, therefore they have more weight than the distant ones.

Figure 4 shows how the weight is distributed to the context words, taking its maximum value on the polysemous word.

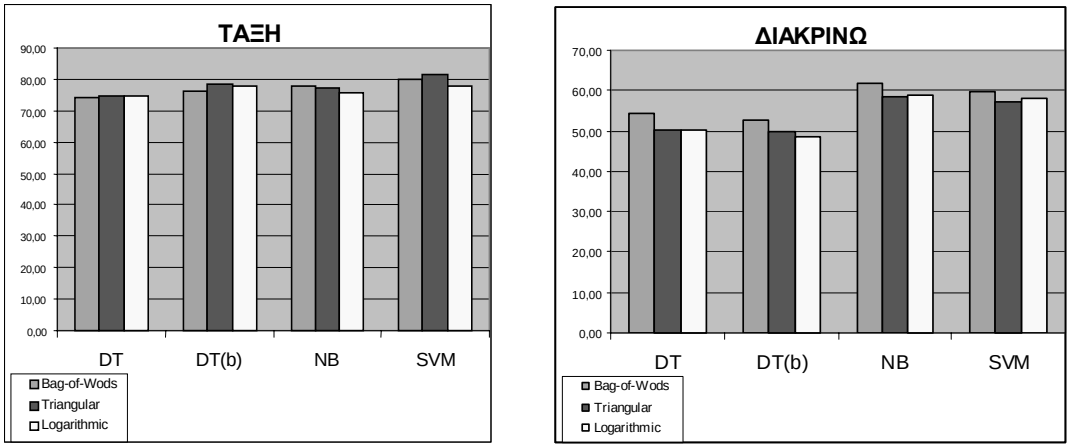


Figure 5 - Comparative Charts. DT: decision tree, DT(b): decision tree with boosting, NB: Naive Bayesian, SVM: Support Vector Machine.

Figure 5 presents the results for the two words, of all three representations that we assessed. The first observation is that the effect of using the distance-based feature weighting is not substantial. Triangular representation benefited disambiguation for the word 'τάξη' (ta-ksi), reaching an 81.65% success rate for the SVM algorithm. The same was not the case for the logarithmic representation, which presented poor results. Naive Bayesian seems to work better with 'bag-of-words' representation, since its success rate decreased from 78.06% to 77.53% and 75.71%, using the two weighting functions.

In the experiments for the word 'διακρίνω' (dia-kri-no), 'bag-of-words' comes out as the best representation. Indeed, for all the algorithms there was a slightly reduced performance, except for decision trees where the success rate fell even more drastically from 54.26% to 50.19% and 50.38%.

Overall, Support Vector Machines achieved the best performance for the word 'τάξη' (ta-ksi), with a rate of 81.65%, using the triangular weight-distribution function. For the word 'διακρίνω' (dia-kri-no), the Naive Bayesian Classifier presented the best results, with a rate of 61.82%, using the bag-of-words representation.

6. Concluding Remarks and Further Work

The study presented here is a first approach to the difficult task of Word Sense Disambiguation for the Greek language, using various supervised learning methods. Our purpose was mostly to gain an intuition for the problem and the special characteristics of the Greek language, rather than to achieve a very high success rate. For this purpose, we chose an "easy" case, the noun 'τάξη' (ta-ksi), and a "hard" one, the verb 'διακρίνω' (ta-ksi). Using these two words, we evaluated four different machine learning methods and compared their performance. Furthermore, we tested each of the methods, using three different feature-weighting schemes, which vary the importance of contextual information, according to the distance from the ambiguous word.

As expected, all four methods did very well in the easy case and less so in the harder case. The difference in performance was approximately 20 percentage points. Given the difficulty that the human annotators had with the "hard" case, the results that the learning methods achieved are very encouraging. Overall, Support Vector Machines and the Naive Bayesian classifier outperformed decision trees, even with the use of boosting. To some extent, this phenomenon can be attributed to the lack of higher-level features in the representation of the training instances, i.e., the use of low-level lexical information. Classifiers that combine the features in a multiplicative fashion seem more appropriate for this type of data. Finally, the use of distance-based feature weighting did not seem to affect the performance of the methods substantially. This is an unexpected result and we are currently examining variants that might lead to improvement. One of these is the use of a larger feature set, i.e., allowing more context words to pass the "information gain" test. Additionally, we are examining the selection of features, using the "weighted" vectors, rather than the original binary ones.

At a more abstract level, one issue that has arisen is the suitability of standard dictionary sense definitions for Word Sense Disambiguation systems. When reading the dictionary, all definitions seem quite reasonable and accordant to our linguistic perception. But when we attempted to label the word senses in real examples for the verb 'διακρίνω' (dia-kri-no), we discovered how confusing and unsuitable the definitions were for the specific problem. The agreement rate was not satisfactory and there was even a case where each of the four annotators suggested a different sense label! This problem has been noted also by other researchers [16] and we are therefore encouraged to examine the construction of specific sense definitions for different applications. For example, if we want to perform machine translation we might

need different sense definitions from those that would apply to an information retrieval application. Taking this approach one step further, we would like to examine unsupervised learning methods, such as in [13], for the construction of application-specific, or even domain-specific, sense definitions.

Another issue that we are examining is the use of higher-level features for the context words, such as syntactic information, and their effect to the disambiguation performance. Moreover, in order to draw more valuable conclusions about the disambiguation of Greek words, we are planning to extend our corpora with more words from different parts of speech, such as adjectives and adverbs. Eventually, we would like to construct a semantically annotated corpus for Greek that will be a publicly available resource. Finally, assuming the collection and tagging of sufficient data, our purpose is to attempt a more realistic 'large-vocabulary' approach to the problem.

References

[1] Black E.: An experiment in computational discrimination of English word senses. IBM Journal of Research and Development, v.32, n.2, (1988) 185-194

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

[3] Duda, R. O. and Hart, P. E.: Pattern Classification and Scene Analysis, John Wiley, (1973)

[4] Freund, Y. and Schapire, R. E.: Experiments with a new boosting algorithm. In Proceedings of the International Conference on Machine Learning, 1996.

[5] Gale, W. A., Church, K. W. and Yarowsky, D.: A method for disambiguating word senses in a large corpus. Computers and the Humanities, v.26, (1993) 415-439

[6] Ide N. and Véronis J. Introduction to the special issue on word sense disambiguation: the state of the art. Comp. Linguistics (1998), 24(1):1-40.

[7] Joachims T. Text Categorization with Support Vector Machines: Learning with Many Relevant Features. Proceedings of the European Conference on Machine Learning, Springer, (1998).

[8] Leacock, C., Chodrow, M. and Miller, G. A.: Using corpus statistics and WordNet relations for sense identification. Computational Linguistics, v.24, n.1, (1998) 147-165

[9] Leacock, C., Towell, G. and Voorhees, E. M.: Corpus-based statistical sense resolution. In Proceedings of the ARPA Human Languages Technology Workshop (1993)

[10] Lesk M. Automatic Sense Disambiguation Using Machine Readable Dictionaries: How to Tell a Pine Cone from an Ice Cream Cone, in Proc. SIGDOC, Toronto.

[11] Paliouras G., Karkaletsis V., Androutsopoulos I., Spyropoulos C. D. Learning Rules for Large-Vocabulary Word Sense Disambiguation: A Comparison of Various Classifiers. Natural Language Processing 2000: 383-394

[12] Quinlan, J. R.: C4.5: Programs for machine learning, Morgan - Kaufmann (1993).

[13] Schütze, H.: Automatic word sense discrimination. Computational Linguistics, v.24, m.1, (1998) 97-124

[14] Towell G. and Voorhees E.M. Disambiguating Highly Ambiguous Words. Computational Linguistics (1998), 24(1):125-145,.

[15] Veronis J. and Nancy I. Word sense disambiguation with very large neural networks extracted from machine readable dictionaries (1990). In COLING 90, volume 2, pages 389--394, Helsinki.

[16] Véronis J. Sense tagging: Don't look for the meaning but for the use, Computational Lexicography and Multimedia Dictionaries (COMLEX'2000) (pp. 1-9). Kato Achaia Greece

[17] Yarowsky D.: 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 (1994), pp. 88--95 Las Cruces, NM

[18] Yarowsky, D.: Word-sense disambiguation using statistical models of Roget's categories trained on large corpora. In Proceedings of the International Conference in Computational Linguistics, (1992) 454-460

[19] Yarowsky D. *Unsupervised word sense disambiguation rivaling supervised methods*. In Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics (1995), pages 189--196, Cambridge, MA.