The Importance of High Quality Input for WSD: An Application-Oriented Comparison of Part-of-Speech Taggers

Tanja Gaustad
Humanities Computing
University of Groningen
P.O. Box 716
9700 AS Groningen, The Netherlands
Tel +31-(0)50-363 59 77
Fax +31-(0)50-363 68 55
T.Gaustad@let.rug.nl

Abstract

In this paper, we present an application-oriented evaluation of three Part-of-Speech (PoS) taggers in a word sense disambiguation (WSD) system. Following the intuition that high quality input is likely to influence the final results of a complex system, we test whether the more accurate taggers also produce better results when integrated into the WSD system. For this purpose, a stand-alone evaluation of the PoS taggers is used to assess which tagger is the most accurate. The results of the WSD task, computed on the training section of the Dutch Senseval-2 data, including the PoS information from all three taggers show that the most accurate PoS tags do indeed lead to the best results, thereby verifying our hypothesis. A surprising result, however, is the fact that the performance of the complex WSD system with the different PoS tags included does not necessarily reflect the stand-alone accuracy of the PoS taggers.

1 Introduction

Certain NLP tools are typically used as a subcomponent or a pre-processor in a more complex system, rather than as a complete application in their own right. A typical example of such tools are Part-of-Speech (PoS) taggers. What is usually not taken into account is the fact that the quality (in terms of accuracy) of each subpart of a complex system is likely to influence the final results considerably. Lately, standardized evaluation of NLP resources has gained more importance in the field of Computational Linguistics (e.g. CLEF workshops in information retrieval, Parseval, Senseval), but a tendency towards more application-oriented evaluation is only beginning.

In this paper, we will proceed to an application-oriented comparison of three PoS taggers in a word sense disambiguation (WSD) system. We will evaluate to what extent differences in stand-alone PoS accuracy influence the results obtained in the complex WSD system using the acquired PoS information. Since the Dutch data we use is not only ambiguous with regard to meaning but also with regard to PoS, accurate PoS information is very important to achieve high disambiguation accuracy.

The paper is structured as follows: We will start with a detailed description and comparison of the three PoS taggers including a stand-alone evaluation in order to compare their performance independently of the application to the WSD task. Then follows a description of the WSD system in which (the output of) the different PoS taggers will be incorporated and tested. This includes a presentation of the machine learning algorithm employed for classification (maximum entropy) and its application to WSD, as well as a note on the data and the settings used for the reported experiments. Next, the application-dependent results of the three PoS taggers will be presented and discussed. We end the paper with conclusions and some ideas for future work.

2 Comparison of Part-of-Speech Taggers

The PoS taggers we compare in this article are:

- a Hidden Markov Model tagger (section 2.1),
- a Memory-Based tagger (section 2.2),
- a transformation-based tagger (section 2.3).

We chose these three taggers because they were readily available, could easily be trained for Dutch without major changes in the architecture, and represent distinct, widely used types of existing PoS taggers.

All three taggers were trained on the Dutch Eindhoven corpus (uit den Boogaart, 1975) using the WOTAN tag set (Berghmans, 1994). The original WOTAN tag set, consisting of 233 tags, was too detailed for our purpose. Instead, we used the limited WOTAN tag set of 48 PoS tags developed by (Drenth, 1997) for training and testing in the stand-alone comparison.

In the context of our WSD application, however, we are only interested in the main PoS categories. Therefore, we discarded all additional information from the assigned PoS tags in the WSD corpus. This resulted in 12 different tags being kept: Adj (adjective), Adv (adverb), Art (article), Conj (conjunction), Int (interjection), Misc (miscellaneous), N (noun), Num (numeral), Prep (preposition), Pron (pronoun), Punc (punctuation), and V (verb).[1]

For the stand-alone results, 80% of the training data was actually used for training, 10% for tuning (setting of parameters, etc.) and the accuracy was computed on the remaining 10%. Note that the results of the stand-alone comparison solely serve to illustrate the difference in performance observed independently of an application in order to be able to assess the added value of a more accurate PoS tagger in the WSD application.

2.1 Hidden Markov Model PoS Tagger

The first PoS tagger we used is the trigram Hidden Markov Model (HMM) tagger (Prins and van Noord, 2003) developed in the context of 'Alpino', a natural language understanding system for Dutch (Bouma et al., 2001; van der Beek et al., 2002).[2]

In this standard trigram HMM, each state corresponds to the previous two PoS tags and the probabilities are directly estimated from the labeled training corpus (Manning and Schütze, 1999). There are two types of probabilities relevant in this model, the probability of a tag given the preceding two tags $P(t_i|t_{i-2}t_{i-1})$ as well as the probability of a word given its tag $P(w_i|t_i)$.

These probabilities are computed for each tag individually. Training the HMM with the forward-backward algorithm, we can calculate $P(t_i = t)$ for all potential tags:

$$P(t_i = t) = \alpha_i(t)\beta_i(t)$$

where $\alpha_i(t)$ is the total (summed) probability of all paths through the model that end at tag t at position i, and $\beta_i(t)$ is the total probability of all paths starting at tag t in position i continuing to the end. Comparing all the values for $P(t_i = t)$, unlikely tags are removed.

Smoothing of the trigram probabilities is achieved through a variant of linear interpolation (Collins, 1999) where lower order models are also taken into account and weights are assigned to each of the models to capture their relative importance.

Since the tagger's lexicon has been created from the training data, the test data very likely contains unknown words which means that no initial set of possible tags can be assigned to these words. Two different strategies have been incorporated in the HMM tagger used here. First, a heuristic rule for recognizing names has been added which assigns an N tag to all capitalized words.[3] Second, a set of automata (also created on the basis of the training data) is used to find possible tags based on the suffixes of unknown words (Daciuk, 2000).

2.2 Memory-Based PoS Tagger

The second tagger we have used in the experiments reported here is the Memory-Based Tagger (MBT) (Daelemans et al., 2002a).[4] It is a PoS tagger based on Memory-Based Learning, an extension of the k-Nearest-Neighbour approach, which has proved to

[1] See table 2 for a distribution of the main PoS tag categories in the WSD data and the Eindhoven corpus.

[2] See `http://www.let.rug.nl/~vannoord/alp`.

[3] Words in sentence initial position are decapitalized beforehand.

[4] Freely available for research purposes at `http://ilk.uvt.nl/software.html`.

be successful for a number of languages and NLP applications (Zavrel and Daelemans, 1999; Veenstra et al., 2000; Hoste et al., 2002).

MBT consists of two components: a memory-based learning component and a performance component for similarity-based classification. During classification, the similarity between a previously unseen test example and the examples in memory is computed using a similarity metric. The category of the test example is then extrapolated based on the most similar example(s).

Given an annotated corpus, three data structures are automatically extracted: a lexicon, a case base for known words, and a case base for unknown words. During tagging, each word is looked up in the lexicon and, if it is found, its lexical representation is retrieved and its context determined. The resulting pattern is disambiguated using extrapolation from the nearest neighbours in the known words case base. If a word is not present in the lexicon, its lexical representation is computed on the basis of its form, its context is determined, and the resulting pattern is disambiguated using extrapolation from nearest neighbours in the unknown words case base. In both cases, the output is a best guess of the category for the word in its current context.

For the known words, the preceding two tags and words as well as the ambiguous tag and word to the right of the current position have been used to construct the known words case base. Classification was achieved using the IGTREE algorithm with one nearest neighbour. For unknown words, the preceding tag, the ambiguous tag to the right, as well as the first and the last three letters of the ambiguous word itself were taken into account to construct the unknown words case base. For classification, the IB1 algorithm with 9 nearest neighbours was used. In both cases GainRatio feature weighting was applied. For details on the different possible algorithms see (Daelemans et al., 2002b).

2.3 Transformation-Based PoS Tagger

As the third member of the comparison, we used a Brill-style transformation-based tagger (TBL) (Brill, 1995) for Dutch (Drenth, 1997). The main components of a transformation-based tagger are a specification of admissible transformations and a learning algorithm. Interdependencies between words and tags are modeled by starting out with an imperfect tagging which is gradually transformed into one with fewer errors. This is achieved by selecting and sequencing transformation rules using the learning algorithm.

In an initial step, each word is assigned a tag independent of context. A known word is assigned its most likely tag determined by a maximum likelihood estimation from the training corpus. An unknown word, on the other hand, is assigned a tag based on lexical rules learned during training. All unknown words are initially tagged N. The application of lexical rules adapts the tag (where necessary) based on the local properties of the unknown word, such as its suffix.

After each word has received an initial tag, contextual rules are applied changing the initial PoS tag (where necessary) based on the context of the word to be tagged. The best contextual transformation rules and their order of application are selected by the learning algorithm during training.

The present implementation of the TBL PoS tagger for Dutch uses around 250 lexical rules and 300 contextual rules.

PoS Tagger	Accuracy
TBL	94.20
HMM	95.93
MBT	**96.21**

Table 1: Stand-alone results (in %) for the three PoS taggers on 10% of the Eindhoven corpus data

2.4 Stand-Alone Results for the PoS Taggers

As we have mentioned earlier, the stand-alone results for the PoS taggers were computed using 80% of the Eindhoven Corpus (containing a total of 760,000 words) for training and 10% for tuning. The accuracy shown in table 1 was computed on the remaining 10% of the corpus.

We can clearly see that the MBT tagger is performing best, followed by the HMM tagger, the least accurate tagger being the TBL tagger.[5]

If the hypothesis that more accurate input to complex systems will produce more accurate results is

[5] All results differ significantly applying the paired sign test with a confidence level of 95%.

correct, then these stand-alone results raise the expectation that when applying all three taggers in our WSD system—with all other settings being equal—accuracy should be highest when the MBT tagger was used to tag the data. Performance is expected to decrease with the use of the HMM tagger and to be lowest for the TBL tagger.

This expectation might be falsified by the (possible) corpus dependency of the three PoS taggers: the capacity to generalize from the training corpus to the corpus to be tagged might be bigger in one tagger than in another, which means that the results obtained in the complex system can diverge from the expectation raised by the stand-alone results.

Let us now turn to the application in which we will use the three PoS taggers presented and evaluated above.

3 Word Sense Disambiguation for Dutch

Semantic lexical ambiguity remains a major problem in natural language processing (NLP) for which to date no satisfactory solution has been found. Word sense disambiguation (WSD) refers to the resolution of lexical semantic ambiguity and its goal is to attribute the correct sense(s) to words in a certain context. Accurate disambiguation of word senses is important for e.g. machine translation, information retrieval or document extraction.

The WSD system used in these experiments is a supervised corpus-based algorithm combining statistical classification with different kinds of linguistic information. This system explores the intuition that (high quality) linguistic information is beneficial for WSD. PoS is definitely one of the more accessible sources of linguistic knowledge. The hypothesis behind comparing various PoS taggers in this application is that the quality of the PoS tags assigned to the data can significantly influence the accuracy obtained by our WSD system.

In contrast to the English WSD data, the Dutch Senseval-2 WSD data is ambiguous with regard to PoS. This means that accurate PoS information is even more important since the WSD system is supposed to do morpho-syntactic as well as semantic disambiguation.

We will now first explain the statistical classification algorithm used and then proceed to describe the WSD system, its settings as well as the corpus used to generate the comparative results.

3.1 Maximum Entropy Classification

The statistical classifier used in the experiments reported here is a *maximum entropy classifier* (Berger et al., 1996). Maximum entropy is a general technique for estimating probability distributions from data. If nothing about the data is known, it involves selecting the most uniform distribution where all events have equal probability. In other words, it means selecting the distribution which maximises the entropy.

If data is available, labeled training data is seen as a number of features which are used to derive a set of constraints for the model. This set of constraints characterises the class-specific expectations for the distribution. So, while the distribution should maximise the entropy, the model should also satisfy the constraints imposed by the labeled training data. A maximum entropy model is thus the model with maximum entropy of all models that satisfy the set of constraints derived from the training data.

The maximum entropy model is built using the following formula:

$$p(c|x) = \frac{1}{Z} exp\left(\sum_i \lambda_i f_i(x,c)\right)$$

where the property function $f_i(x,c)$ represents the number of times feature i is used to find class c for event x, and the weights λ_i are chosen to maximise the likelihood of the training data and, at the same time, maximise the entropy of p.

This means that during training the weight λ_i for each feature i present in the training data is computed and stored. During testing, the sum of the weights λ_i of all features i found in the test instances is computed for each class c and the class with the highest score is chosen.

The main advantage of maximum entropy modeling is that the property functions, including all the different types of (linguistic) information in the model, take into account any information which might be useful for disambiguation. Thus, dissimilar types of information can be combined into a single model for WSD and no independence assumptions (as in e.g. a Naive Bayes algorithm) are necessary.

3.2 Corpus and System Settings

The corpus used in this evaluation is the Dutch Senseval-2 corpus[6] (see (Hendrickx and van den Bosch, 2001) for a detailed description). In the experiments reported here, we only made use of the *training section* of the Dutch Senseval-2 dataset, containing approximately 120,000 tokens and 9,300 sentences.

In a first step, the corpus is lemmatized and PoS tagged. Then, for each ambiguous wordform/lemma[7] all instances of its occurrence are extracted from the corpus. These instances are then transformed into different feature vectors. So a feature vector of the ambiguous wordform 'aarde' (earth/soil) corresponding to the model which comprises all possible information (incl. PoS) and uses context words would look like this:

```
aarde N gat in de , zodat het aarde_grond
```

where the first slot represents the lemma, the second the PoS, the third to eighth slot are the context words (left before right) and the last slot represents the sense or class.[8] Only context words within the same sentence as the ambiguous wordform/lemma were taken into account. If for instance there was no left context, it was filled with "empty" features. Varying the information included, different feature sets are constructed.

For the *basic classifier* based on ambiguous wordforms, the feature set contains the corresponding lemma as well as a context of three words to the left and to the right of the ambiguous word. For the basic classifier based on ambiguous lemmas, the corresponding wordform and the context are included. The context can either be composed of wordforms or lemmas. For the *classifiers including PoS tags*, we in addition include the PoS tags of the ambiguous wordform/lemma from the various PoS taggers.

On the basis of the different feature sets, separate classifiers are built for every ambiguous wordform or lemma. This implies that the basis for grouping occurrences of particular ambiguous words together is that either their wordform or their lemma is the same. In the experiments presented here, a frequency threshold of 10 was used, which means that classifiers were only built for the wordforms with an amount of training instances equal to or above the threshold. For the remaining wordforms, the baseline count was used, thus assigning the most frequent sense to every instance.

In total, there were 1,364 ambiguous lemmas in the corpus of which 622 presented 10 or more occurrences, and 952 ambiguous wordforms of which 486 had 10 or more occurrences. So 622 lemma classifiers and 486 wordform classifiers were built.

The context was treated as a 'bag of words' which means that the position of a context word relative to the ambiguous wordform was not taken into account. This approach was chosen to help limit the data sparseness problem: if the context features are all treated dependent on their position relative to the ambiguous word in the sentence, the model will have more features to assign weights to. This means that the sparse data problem will be worse. If, on the other hand, context features are "lumped" together independent of their relative position, there are less features to be estimated and there is more data for the particular feature 'context'.

4 Results and Evaluation of the WSD Application

Before we turn to the actual results of using the different PoS taggers in our WSD system for Dutch, let us first compare the differences regarding the assigned PoS tags. Table 2 shows the distribution of the different PoS tags in the WSD data depending on the PoS tagger used, as well as the distribution of the PoS tags in the training corpus.

A major difference between the distribution of PoS tags is that both the HMM and MBT tagger assign more V tags, whereas the TBL tagger assigns more N tags. The preference for N tags in the TBL tagger can be explained by the fact that all unknown words initially get tagged N. Also, in Dutch verbal infinitives have the same morphological suffix as plural nouns (*-en*). INT and Misc differ with all three taggers, but we could not detect any obvious reason for this. As we can see from table 2, there

[6] For more information on Senseval and for downloads of the data see `http://www.senseval.org/`.

[7] A wordform/lemma is 'ambiguous' if it has two or more different senses in the training data. The sense '=' is seen as marking the basic sense of a word/lemma and is therefore also taken into account.

[8] 'Sense' or 'class' refers to the different labels which disambiguate the ambiguous wordforms/lemmas.

PoS	TBL		HMM		MBT		Train. Corpus
N	22,830	(19.46%)	20,041	(17.08%)	20,384	(17.37%)	20.35%
Punc	19,792	(16.87%)	20,151	(17.17%)	20,142	(17.17%)	12.69%
V	17,645	(15.04%)	19,505	(16.62%)	19,556	(16.66%)	15.13%
Pron	13,880	(11.83%)	13,938	(11.88%)	13,885	(11.83%)	9.82%
Adv	11,250	(9.58%)	11,289	(9.62%)	11,178	(9.53%)	8.19%
Art	9,477	(8.08%)	9,350	(7.96%)	9,328	(7.95%)	9.39%
Prep	8,190	(6.98%)	8,358	(7.26%)	8,229	(7.01%)	10.54%
Conj	6,713	(5.72%)	6,742	(5.74%)	6,770	(5.77%)	5.18%
Adj	6,313	(5.38%)	6,621	(5.63%)	6,626	(5.65%)	6.53%
Num	869	(0.74%)	713	(0.61%)	744	(0.63%)	1.78%
Int	376	(0.32%)	559	(0.47%)	455	(0.39%)	0.18%
Misc	3	(0.003%)	71	(0.04%)	41	(0.04%)	0.22%

Table 2: Frequencies of PoS tags assigned by each PoS tagger in the WSD data and distribution of PoS in the training corpus

are bigger differences between the TBL tagger and the other two, whereas the differences between the HMM and the MBT tagger are less noticeable.

In order to test the real error of the classifiers built, we used a leave-one-out approach (Weiss and Kulikowski, 1991; Manning and Schütze, 1999). This means that every data item in turn is selected once as a test item and the classifier is trained on all remaining items. The accuracy of a single classifier is then the number of data items correctly predicted. The overall accuracy is the total of data items correctly predicted by all classifiers.

The results in table 3 show the average accuracy on our training data using leave-one-out as a test method with respectively wordforms and lemmas as basis.

As the table of results shows, the WSD system performs well. The basic classifiers containing a minimum of information already do significantly better than the frequency baseline.[9] Furthermore, adding PoS as extra linguistic information—next to the lemma/wordform and the context already included in the basic classifiers—does increase results over the accuracy achieved with a basic classifier. This supports the underlying hypothesis behind the WSD system that more linguistic information is beneficial for WSD. Since the WSD data needs to be disambiguated morpho-syntactically as well as with regard to lexical semantic ambiguity, it is not surprising that adding PoS information achieves better results than only using the lemma/wordform and context.

Comparing the performance among the different PoS taggers, we can see quite clearly that our expectations are (partly) confirmed: the MBT tagger, which did best in the stand-alone evaluation, is also working best in the WSD system. This is the case for all setups: using wordforms or lemmas as basis for the classifiers, as well as for classifiers including context as wordforms or as lemmas.[10]

Surprisingly enough, the hypothesis does not hold for the "ranking" of the HMM and TBL taggers. Despite the fact that the HMM tagger performed second best in the stand-alone evaluation, it does not perform better than the TBL tagger when integrated into the WSD system.

A possible explanation might be that the difference between the training corpus and the WSD data is so big that the HMM tagger is no longer more accurate than the TBL tagger in the WSD application, leading to the conclusion that the HMM tagger is more corpus dependent than the TBL tagger. A possible reason might be that the heuristics for unknown

[9] Assigning the most frequent sense to every occurrence of an ambiguous wordform/lemma.

[10] Applying the paired sign test with a confidence level of 95%, all results using MBT PoS tags were found to be statistically significantly better than results with other PoS tags (and than the basic classifiers). The classifiers including TBL and HMM PoS tags do not differ significantly from each other, but both perform significantly better than the basic classifiers.

Base: Wordforms			
Feature set	Accuracy		
baseline	76.70		
lemma, con. words (basic)	80.81		
lemma, con. lemmas (basic)	80.52		
	TBL	HMM	MBT
lemma, pos, con. words	81.67	81.67	**81.89**
lemma, pos, con. lemmas	81.42	81.36	**81.67**

Base: Lemmas			
Feature set	Accuracy		
baseline	73.41		
word, con. words (basic)	82.52		
word, con. lemmas (basic)	82.25		
	TBL	HMM	MBT
word, pos, con. words	83.32	83.34	**83.46**
word, pos, con. lemmas	83.06	83.05	**83.30**

Table 3: WSD results (in %) comparing the effect of integrating the output of different POS-taggers into a complex system

words in the HMM tagger produces worse results on the WSD data than the heuristics used by the TBL tagger. Since no gold-standard PoS tagged version of the WSD data exists, it is difficult to investigate this puzzle any further.

Nevertheless, our hypothesis that highly accurate input influences the results of a complex system is at least partly verified: the most accurate PoS tags also produce the most accurate results when integrated into our WSD system.

5 Conclusion and Future Work

In this paper, we tested the hypothesis whether high quality input improves the final results of a complex NLP system. We have therefore proceeded to an application-oriented evaluation of three PoS taggers in a WSD system. A transformation-based tagger, a Hidden Markov Model tagger, and a memory-based tagger were compared for this purpose.

After the MBT tagger has been established as the most accurate tagger in a stand-alone evaluation, the PoS information from all three taggers is integrated into our WSD system for Dutch. This supervised system uses maximum entropy classifiers which allow to integrate various sources of information into a single model.

The results computed on the training part of the Dutch Senseval-2 corpus show that the MBT tagger also produces the best results in the WSD system. This clearly indicates that highly accurate input into a WSD system is producing better results than qualitatively lesser input.

A surprising result, however, was the fact that the performance of the complex WSD system with the different PoS tags included does not necessarily reflect the stand-alone accuracy of the PoS taggers. Even though the HMM tagger performed better than the TBL tagger in the stand-alone comparison, there is no significant difference to be observed in the results of the WSD system. A possible explanation might be corpus dependency.

For future work, we would like to include the PoS tags of the context wordforms or lemmas to see whether our hypothesis still holds then. It would also be interesting to see whether the overall results are further improved by this additional information.

Acknowledgments

This research was carried out within the framework of the PIONIER Project *Algorithms for Linguistic Processing*. This PIONIER Project is funded by NWO (Dutch Organization for Scientific Research)

and the University of Groningen. We are grateful to Robbert Prins for his help with the HMM tagger as well as to Gertjan van Noord and Menno van Zaanen for comments and discussions.

References

Adam Berger, Stephen Della Pietra, and Vincent Della Pietra. 1996. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1):39–71.

Johan Berghmans. 1994. WOTAN—een automatische grammaticale tagger voor het Nederlands. Master's thesis, Nijmegen University, Nijmegen.

Gosse Bouma, Gertjan van Noord, and Robert Malouf. 2001. Alpino: Wide-coverage computational analysis of Dutch. In Walter Daelemans, Khalil Sima'an, Jorn Veenstra, and Jakub Zavrel, editors, *Computational Linguistics in the Netherlands 2000*, pages 45–59, Amsterdam. Rodopi.

Eric Brill. 1995. Transformation-based error-driven learning and natural language processing: A case study in part of speech tagging. *Computational Linguistics*, 21(4):543–565.

Michael Collins. 1999. *Head-Driven Statistical Models for Natural Language Parsing*. Ph.D. thesis, Computer and Information Science Department, University of Pennsylvania, Philadelphia.

Jan Daciuk. 2000. Finite state tools for natural language processing. In *Proceedings of the COLING 2000 Workshop "Using Toolsets and Architectures to Build NLP Systems"*, pages 34–37, Centre Universitaire, Luxembourg.

Walter Daelemans, Jakub Zavrel, Ko van der Sloot, and Antal van den Bosch. 2002a. MBT: Memory-Based tagger, reference guide. Technical Report ILK 02-09, Induction of Linguistic Knowledge, Computational Linguistics, Tilburg University, Tilburg. version 1.0.

Walter Daelemans, Jakub Zavrel, Ko van der Sloot, and Antal van den Bosch. 2002b. TiMBL: Tilburg Memory-Based learner, reference guide. Technical Report ILK 02-10, Induction of Linguistic Knowledge, Computational Linguistics, Tilburg University, Tilburg. version 4.3.

Erwin W. Drenth. 1997. Using a hybrid approach towards Dutch part-of-speech tagging. Master's thesis, Humanities Computing, University of Groningen, Groningen.

Iris Hendrickx and Antal van den Bosch. 2001. Dutch word sense disambiguation: Data and preliminary results. In *Proceedings of Senseval-2, Second International Workshop on Evaluating Word Sense Disambiguation Systems*, pages 13–16, Toulouse.

Véronique Hoste, Walter Daelemans, Iris Hendrickx, and Antal van den Bosch. 2002. Evaluating the results of a Memory-Based word-expert approach to unrestricted word sense disambiguation. In *Proceedings of the ACL-02 Workshop on Word Sense Disambiguation: Recent Successes and Future Directions*, pages 95–101, Philadelphia.

Christopher Manning and Hinrich Schütze. 1999. *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge.

Robbert Prins and Gertjan van Noord. 2003. Reinforcing parser preferences through tagging. *Traitement automatique des langues*. forthcoming.

Pieter uit den Boogaart. 1975. *Woordfrequenties in Geschreven and Gesproken Nederlands*. Oosthoek, Scheltema en Holkema, Utrecht.

Leonoor van der Beek, Gosse Bouma, Rob Malouf, and Gertjan van Noord. 2002. The Alpino dependency treebank. In Mariët Theune, Anton Nijholt, and Hendri Hondorp, editors, *Computational Linguistics in the Netherlands 2001*, pages 8–22, Amsterdam. Rodopi.

Jorn Veenstra, Antal van den Bosch, Sabine Buchholz, Walter Daelemans, and Jakub Zavrel. 2000. Memory-Based word sense disambiguation. *Computers and the humanities*, 34(1-2):171–177.

Sholom Weiss and Casimir Kulikowski. 1991. *Computer Systems that Learn*. Morgan Kaufman, San Mateo.

Jakub Zavrel and Walter Daelemans. 1999. Recent advances in Memory-Based part-of-speech tagging. In *VI Simposio Internacional de Communicación Social*, pages 590–597, Santiago de Cuba.