Proceedings ACL-EACL 1997, Madrid, p. 436-443, 1997

Memory-Based Learning: Using Similarity for Smoothing

Jakub Zavrel and **Walter Daelemans**
Computational Linguistics
Tilburg University
PO Box 90153
5000 LE Tilburg
The Netherlands
{zavrel,walter}@kub.nl

Abstract

This paper analyses the relation between the use of similarity in Memory-Based Learning and the notion of backed-off smoothing in statistical language modeling. We show that the two approaches are closely related, and we argue that feature weighting methods in the Memory-Based paradigm can offer the advantage of automatically specifying a suitable domain-specific hierarchy between most specific and most general conditioning information without the need for a large number of parameters. We report two applications of this approach: PP-attachment and POS-tagging. Our method achieves state-of-the-art performance in both domains, and allows the easy integration of diverse information sources, such as rich lexical representations.

1 Introduction

Statistical approaches to disambiguation offer the advantage of making the most likely decision on the basis of available evidence. For this purpose a large number of probabilities has to be estimated from a training corpus. However, many possible conditioning events are not present in the training data, yielding zero Maximum Likelihood (ML) estimates. This motivates the need for *smoothing* methods, which re-estimate the probabilities of low-count events from more reliable estimates.

Inductive generalization from observed to new data lies at the heart of machine-learning approaches to disambiguation. In Memory-Based Learning[1] (MBL) induction is based on the use of similarity (Stanfill & Waltz, 1986; Aha *et al.*, 1991; Cardie, 1994; Daelemans, 1995). In this paper we describe how the use of similarity between patterns embodies a solution to the sparse data problem, how it relates to backed-off smoothing methods and what advantages it offers when combining diverse and rich information sources.

We illustrate the analysis by applying MBL to two tasks where combination of information sources promises to bring improved performance: PP-attachment disambiguation and Part of Speech tagging.

2 Memory-Based Language Processing

The basic idea in Memory-Based language processing is that processing and learning are fundamentally interwoven. Each language experience leaves a memory trace which can be used to guide later processing. When a new instance of a task is processed, a set of relevant instances is selected from memory, and the output is produced by analogy to that set.

The techniques that are used are variants and extensions of the classic k-nearest neighbor (k-NN) classifier algorithm. The instances of a task are stored in a table as patterns of feature-value pairs, together with the associated "correct" output. When a new pattern is processed, the k nearest neighbors of the pattern are retrieved from memory using some similarity metric. The output is then determined by extrapolation from the k nearest neighbors, i.e. the output is chosen that has the highest relative frequency among the nearest neighbors.

Note that no abstractions, such as grammatical rules, stochastic automata, or decision trees are extracted from the examples. Rule-like behavior results from the linguistic regularities that are present in the patterns of usage in memory in combination with the use of an appropriate similarity metric. It is our experience that even limited forms of abstraction can harm performance on linguistic tasks, which often contain many subregularities and exceptions (Daelemans, 1996).

2.1 Similarity metrics

The most basic metric for patterns with symbolic features is the **Overlap metric** given in equations 1

[1] The Approach is also referred to as Case-based, Instance-based or Exemplar-based.

and 2; where $\Delta(X,Y)$ is the distance between patterns X and Y, represented by n features, w_i is a weight for feature i, and δ is the distance per feature. The k-NN algorithm with this metric, and equal weighting for all features is called IB1 (Aha *et al.*, 1991). Usually k is set to 1.

$$\Delta(X,Y) = \sum_{i=1}^{n} w_i \, \delta(x_i, y_i) \qquad (1)$$

where:

$$\delta(x_i, y_i) = 0 \; if \; x_i = y_i, \; else \; 1 \qquad (2)$$

This metric simply counts the number of (mis)matching feature values in both patterns. If we do not have information about the importance of features, this is a reasonable choice. But if we do have some information about feature relevance one possibility would be to add linguistic bias to weight or select different features (Cardie, 1996). An alternative—more empiricist—approach, is to look at the behavior of features in the set of examples used for training. We can compute statistics about the relevance of features by looking at which features are good predictors of the class labels. Information Theory gives us a useful tool for measuring feature relevance in this way (Quinlan, 1986; Quinlan, 1993).

Information Gain (IG) weighting looks at each feature in isolation, and measures how much information it contributes to our knowledge of the correct class label. The Information Gain of feature f is measured by computing the difference in uncertainty (i.e. entropy) between the situations without and with knowledge of the value of that feature (Equation 3).

$$w_f = H(C) - \frac{\sum_{v \in V_f} P(v) \times H(C|v)}{si(f)} \qquad (3)$$

$$si(f) = -\sum_{v \in V_f} P(v) \log_2 P(v) \qquad (4)$$

Where C is the set of class labels, V_f is the set of values for feature f, and $H(C) = -\sum_{c \in C} P(c) \log_2 P(c)$ is the entropy of the class labels. The probabilities are estimated from relative frequencies in the training set. The normalizing factor $si(f)$ (split info) is included to avoid a bias in favor of features with more values. It represents the amount of information needed to represent all values of the feature (Equation 4). The resulting IG values can then be used as weights in equation 1. The k-NN algorithm with this metric is called IB1-IG (Daelemans & Van den Bosch, 1992).

The possibility of automatically determining the relevance of features implies that many different and possibly irrelevant features can be added to the feature set. This is a very convenient methodology if theory does not constrain the choice enough beforehand, or if we wish to measure the importance of various information sources experimentally.

Finally, it should be mentioned that MB-classifiers, despite their description as table-lookup algorithms here, can be implemented to work fast, using e.g. tree-based indexing into the case-base (Daelemans *et al.*, 1997).

3 Smoothing of Estimates

The commonly used method for probabilistic classification (the Bayesian classifier) chooses a class for a pattern X by picking the class that has the maximum conditional probability $P(class|X)$. This probability is estimated from the data set by looking at the relative joint frequency of occurrence of the classes and pattern X. If pattern X is described by a number of feature-values $x_1, \ldots, x_n$, we can write the conditional probability as $P(class|x_1, \ldots, x_n)$. If a particular pattern $x'_1, \ldots, x'_n$ is not literally present among the examples, all classes have zero ML probability estimates. Smoothing methods are needed to avoid zeroes on events that could occur in the test material.

There are two main approaches to smoothing: count re-estimation smoothing such as the Add-One or Good-Turing methods (Church & Gale, 1991), and Back-off type methods (Bahl *et al.*, 1983; Katz, 1987; Chen & Goodman, 1996; Samuelsson, 1996). We will focus here on a comparison with Back-off type methods, because an experimental comparison in Chen & Goodman (1996) shows the superiority of Back-off based methods over count re-estimation smoothing methods. With the Back-off method the probabilities of complex conditioning events are approximated by (a linear interpolation of) the probabilities of more general events:

$$\begin{aligned} \tilde{p}(class|X) \quad = \quad & \lambda_X \hat{p}(class|X) + \lambda_{X'} \hat{p}(class|X') \\ & + \cdots + \lambda_{X^n} \hat{p}(class|X^n) \end{aligned} \qquad (5)$$

Where $\tilde{p}$ stands for the smoothed estimate, $\hat{p}$ for the relative frequency estimate, λ are interpolation weights, $\sum_{i=0}^{n} \lambda_{X^i} = 1$, and $X \prec X^i$ for all i, where $\prec$ is a (partial) ordering from most specific to most general feature-sets[2] (e.g the probabilities of trigrams (X) can be approximated by bigrams (X') and unigrams (X'')). The weights of the linear interpolation are estimated by maximizing the probability of held-out data (deleted interpolation) with the forward-backward algorithm. An alternative method to determine the interpolation weights without iterative training on held-out data is given in Samuelsson (1996).

[2] $X \prec X'$ can be read as X is more specific than X'.

We can assume for simplicity's sake that the λ_{X^i} do not depend on the value of X^i, but only on i. In this case, if F is the number of features, there are $2^F - 1$ more general terms, and we need to estimate λ_i's for all of these. In most applications the interpolation method is used for tasks with clear orderings of feature-sets (e.g. n-gram language modeling) so that many of the $2^F - 1$ terms can be omitted beforehand. More recently, the integration of information sources, and the modeling of more complex language processing tasks in the statistical framework has increased the interest in smoothing methods (Collins & Brooks, 1995; Ratnaparkhi, 1996; Magerman, 1994; Ng & Lee, 1996; Collins, 1996). For such applications with a diverse set of features it is not necessarily the case that terms can be excluded beforehand.

If we let the λ_{X^i} depend on the value of X^i, the number of parameters explodes even faster. A practical solution for this is to make a smaller number of *buckets* for the X^i, e.g. by clustering (see e.g. Magerman (1994)).

Note that linear interpolation (equation 5) actually performs two functions. In the first place, if the most specific terms have non-zero frequency, it still interpolates them with the more general terms. Because the more general terms should never overrule the more specific ones, the λ_{X^i} for the more general terms should be quite small. Therefore the interpolation effect is usually small or negligible. The second function is the pure back-off function: if the more specific terms have zero frequency, the probabilities of the more general terms are used instead. Only if terms are of a similar specificity, the λ's truly serve to weight relevance of the interpolation terms.

If we isolate the pure back-off function of the interpolation equation we get an algorithm similar to the one used in Collins & Brooks (1995). It is given in a schematic form in Table 1. Each step consists of a back-off to a lower level of specificity. There are as many steps as features, and there are a total of 2^F terms, divided over all the steps. Because all features are considered of equal importance, we call this the *Naive Back-off* algorithm.

Usually, not all features x are equally important, so that not all back-off terms are equally relevant for the re-estimation. Hence, the problem of fitting the λ_{X^i} parameters is replaced by a term selection task. To optimize the term selection, an evaluation of the up to 2^F terms on held-out data is still necessary. In summary, the Back-off method does not provide a principled and practical domain-independent method to adapt to the structure of a particular domain by determining a suitable ordering $\prec$ between events. In the next section, we will argue that a formal operationalization of similarity between events, as provided by MBL, can be used for this purpose. In MBL the similarity metric and feature weighting scheme automatically determine the implicit back-off ordering using a domain independent heuristic, with only a few parameters, in which there is no need for held-out data.

If $f(x_1, ..., x_n) > 0$:
$\tilde{p}(c|x_1, ..., x_n) = \frac{f(c, x_1, ..., x_n)}{f(x_1, ..., x_n)}$

Else if $f(x_1, ..., x_{n-1}, *) + ... + f(*, x_2, ..., x_n) > 0$:
$\tilde{p}(c|x_1, ..., x_n) = \frac{f(c, x_1, ..., x_{n-1}, *) + ... + f(c, *, x_2, ..., x_n)}{f(x_1, ..., x_{n-1}, *) + ... + f(*, x_2, ..., x_n)}$

Else if ...:
$\tilde{p}(c|x_1, ..., x_n) = \frac{...}{...}$

Else if $f(x_1, *, ..., *) + ... + f(*, ..., *, x_n) > 0$:
$\tilde{p}(c|x_1, ..., x_n) = \frac{f(c, x_1, *, ..., *) + ... + f(c, *, ..., *, x_n)}{f(x_1, *, ..., *) + ... + f(*, ..., *, x_n)}$

Table 1: The Naive Back-off smoothing algorithm. $f(X)$ stands for the frequency of pattern X in the training set. An asterix $(*)$ stands for a wildcard in a pattern. The terms at a higher level in the back-off sequence are more specific $(\prec)$ than the lower levels.

4 A Comparison

If we classify pattern X by looking at its nearest neighbors, we are in fact estimating the probability $P(class|X)$, by looking at the relative frequency of the class in the set defined by $sim_k(X)$, where $sim_k(X)$ is a function from X to the set of most similar patterns present in the training data[3]. Although the name "k-nearest neighbor" might mislead us by suggesting that classification is based on exactly k training patterns, the $sim_k(X)$ function given by the Overlap metric groups varying numbers of patterns into *buckets* of equal similarity. A bucket is defined by a particular number of mismatches with respect to pattern X. Each bucket can further be decomposed into a number of *schemata* characterized by the position of a wildcard (i.e. a mismatch). Thus $sim_k(X)$ specifies a $\prec$ ordering in a Collins & Brooks style back-off sequence, where each bucket is a step in the sequence, and each schema is a term in the estimation formula at that step. In fact, the unweighted overlap metric specifies exactly the same ordering as the Naive Back-off algorithm (table 1). In Figure 1 this is shown for a four-featured pattern. The most specific schema is the schema with zero mismatches, which corresponds to the retrieval of an identical pattern from memory, the most general schema (not shown in the Figure) has a mismatch on every feature, which corresponds to the

[3]Note that MBL is not limited to choosing the best class. It can also return the conditional distribution of all the classes.

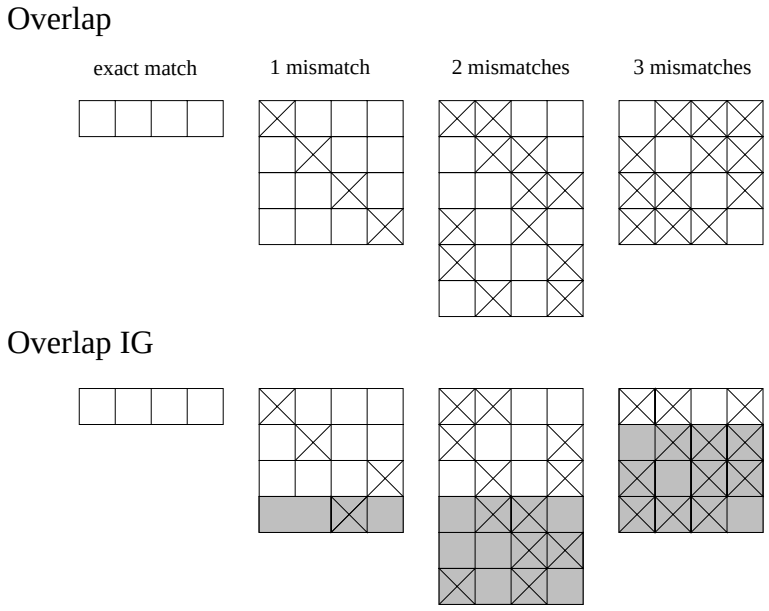

Figure 1: An analysis of nearest neighbor sets into buckets (from left to right) and schemata (stacked). IG weights reorder the schemata. The grey schemata are not used if the third feature has a very high weight (see section 5.1).

entire memory being best neighbor.

If Information Gain weights are used in combination with the Overlap metric, individual schemata instead of buckets become the steps of the back-off sequence[4]. The $\prec$ ordering becomes slightly more complicated now, as it depends on the number of wildcards *and* on the magnitude of the weights attached to those wildcards. Let S be the most specific (zero mismatches) schema. We can then define the $\prec$ ordering between schemata in the following equation, where $\Delta(X, Y)$ is the distance as defined in equation 1.

$$S' \prec S'' \;\Leftrightarrow\; \Delta(S, S') < \Delta(S, S'') \qquad (6)$$

Note that this approach represents a type of implicit parallelism. The importance of the 2^F back-off terms is specified using only F parameters—the IG weights–, where F is the number of features. This advantage is not restricted to the use of IG weights; many other weighting schemes exist in the machine learning literature (see Wettschereck *et al.* (1997) for an overview).

Using the IG weights causes the algorithm to rely on the most specific schema only. Although in most applications this leads to a higher accuracy, because it rejects schemata which do not match the most important features, sometimes this constraint needs to be weakened. This is desirable when: (i) there are a number of schemata which are almost equally relevant, (ii) the top ranked schema selects too few cases to make a reliable estimate, or (iii) the chance that the few items instantiating the schema are mislabeled in the training material is high. In such cases we wish to include some of the lower-ranked schemata. For case (i) this can be done by discretizing the IG weights into bins, so that minor differences will lose their significance, in effect merging some schemata back into buckets. For (ii) and (iii), and for continuous metrics (Stanfill & Waltz, 1986; Cost & Salzberg, 1993) which extrapolate from exactly k neighbors[5], it might be necessary to choose a k parameter larger than 1. This introduces one additional parameter, which has to be tuned on held-out data. We can then use the distance between a pattern and a schema to weight its vote in the nearest neighbor extrapolation. This results in a back-off sequence in which the terms at each step in the sequence are weighted with respect to each other, but without the introduction of any additional weighting parameters. A weighted voting function that was found to work well is due to Dudani (1976): the nearest neighbor schema receives a weight of 1.0, the furthest schema a weight of 0.0, and the other neighbors are scaled linearly to the line between these two points.

[4]Unless two schemata are exactly tied in their IG values.

[5]Note that the schema analysis does not apply to these metrics.

Method	% Accuracy
IB1 (=Naive Back-off)	83.7 %
IB1-IG	84.1 %
LexSpace IG	84.4 %
Back-off model (Collins & Brooks)	84.1 %
C4.5 (Ratnaparkhi et al.)	79.9 %
Max Entropy (Ratnaparkhi et al.)	81.6 %
Brill's rules (Collins & Brooks)	81.9 %

Table 2: Accuracy on the PP-attachment test set.

5 Applications

5.1 PP-attachment

In this section we describe experiments with MBL on a data-set of Prepositional Phrase (PP) attachment disambiguation cases. The problem in this data-set is to disambiguate whether a PP attaches to the verb (as in *I ate pizza with a fork*) or to the noun (as in *I ate pizza with cheese*). This is a difficult and important problem, because the semantic knowledge needed to solve the problem is very difficult to model, and the ambiguity can lead to a very large number of interpretations for sentences.

We used a data-set extracted from the Penn Treebank WSJ corpus by Ratnaparkhi *et al.* (1994). It consists of sentences containing the possibly ambiguous sequence *verb noun-phrase PP*. Cases were constructed from these sentences by recording the features: verb, head noun of the first noun phrase, preposition, and head noun of the noun phrase contained in the PP. The cases were labeled with the attachment decision as made by the parse annotator of the corpus. So, for the two example sentences given above we would get the feature vectors `ate,pizza,with,fork,V.` and `ate,pizza,with,cheese,N.` The data-set contains 20801 training cases and 3097 separate test cases, and was also used in Collins & Brooks (1995).

The IG weights for the four features (`V,N,P,N`) were respectively 0.03, 0.03, 0.10, 0.03. This identifies the preposition as the most important feature: its weight is higher than the sum of the other three weights. The composition of the back-off sequence following from this can be seen in the lower part of Figure 1. The grey-colored schemata were effectively left out, because they include a mismatch on the preposition.

Table 2 shows a comparison of accuracy on the test-set of 3097 cases. We can see that IB1, which implicitly uses the same specificity ordering as the Naive Back-off algorithm already performs quite well in relation to other methods used in the literature. Collins & Brooks' (1995) Back-off model is more sophisticated than the naive model, because they performed a number of validation experiments on held-out data to determine which terms to include and, more importantly, which to exclude from the back-off sequence. They excluded all terms which did not match in the preposition! Not surprisingly, the 84.1% accuracy they achieve is matched by the performance of IB1-IG. The two methods exactly mimic each others behavior, in spite of their huge difference in design. It should however be noted that the computation of IG-weights is many orders of magnitude faster than the laborious evaluation of terms on held-out data.

We also experimented with rich lexical representations obtained in an unsupervised way from word co-occurrences in raw WSJ text (Zavrel & Veenstra, 1995; Schütze, 1994). We call these representations Lexical Space vectors. Each word has a numeric 25 dimensional vector representation. Using these vectors, in combination with the IG weights mentioned above and a cosine metric, we got even slightly better results. Because the cosine metric fails to group the patterns into discrete schemata, it is necessary to use a larger number of neighbors ($k = 50$). The result in Table 2 is obtained using Dudani's weighted voting method.

Note that to devise a back-off scheme on the basis of these high-dimensional representations (each pattern has 4×25 features) one would need to consider up to 2^{100} smoothing terms. The MBL framework is a convenient way to further experiment with even more complex conditioning events, e.g. with semantic labels added as features.

5.2 POS-tagging

Another NLP problem where combination of different sources of statistical information is an important issue, is POS-tagging, especially for the guessing of the POS-tag of words not present in the lexicon. Relevant information for guessing the tag of an unknown word includes contextual information (the words and tags in the context of the word), and word form information (prefixes and suffixes, first and last letters of the word as an approximation of affix information, presence or absence of capitalization, numbers, special characters etc.). There is a large number of potentially informative features that could play a role in correctly predicting the tag of an unknown word (Ratnaparkhi, 1996; Weischedel *et al.*, 1993; Daelemans *et al.*, 1996). A priori, it is not clear what the relative importance is of these features.

We compared Naive Back-off estimation and MBL with two sets of features:

- PDASS: the first letter of the unknown word (p), the tag of the word to the left of the unknown word (d), a tag representing the set of possible lexical categories of the word to the right of the unknown word (a), and the two last letters (s). The first letter provides information about capitalisation and the prefix, the two last letters

about suffixes.

- PDDDAAASSS: more left and right context features, and more suffix information.

The data set consisted of 100,000 feature value patterns taken from the Wall Street Journal corpus. Only open-class words were used during construction of the training set. For both IB1-IG and Naive Back-off, a 10-fold cross-validation experiment was run using both PDASS and PDDDAAASSS patterns. The results are in Table 3. The IG values for the features are given in Figure 2.

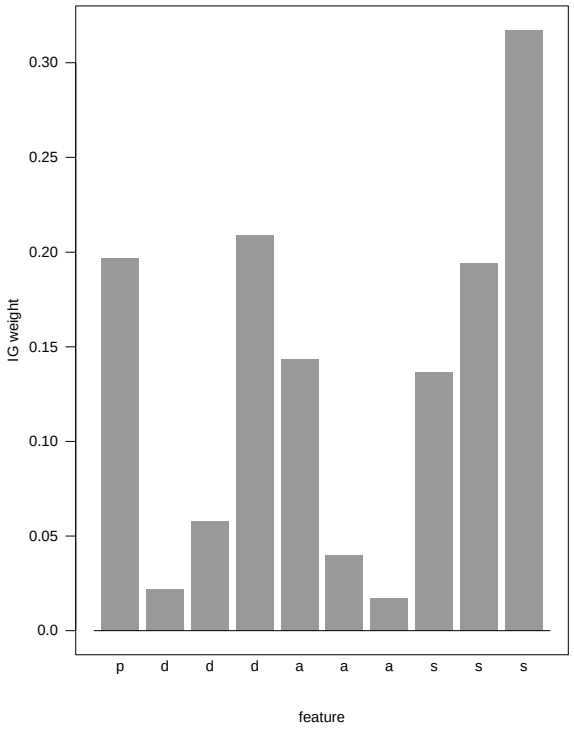

Figure 2: IG values for features used in predicting the tag of unknown words.

	IB1, Naive Back-off	IB1-IG
PDASS	88.5 (0.4)	88.3 (0.4)
PDDDAAASSS	85.9 (0.4)	89.8 (0.4)

Table 3: Comparison of generalization accuracy of Back-off and Memory-Based Learning on prediction of category of unknown words. All differences are statistically significant (two-tailed paired t-test, $p < 0.05$). Standard deviations on the 10 experiments are between brackets.

The results show that for Naive Back-off (and IB1) the addition of more, possibly irrelevant, features quickly becomes detrimental (decrease from 88.5 to 85.9), even if these added features do make a generalisation performance increase possible (witness the increase with IB1-IG from 88.3 to 89.8). Notice that we did not actually compute the 2^{10} terms of Naive Back-off in the PDDDAAASSS condition, as IB1 is guaranteed to provide statistically the same results. Contrary to Naive Back-off and IB1, memory-based learning with feature weighting (IB1-IG) manages to integrate diverse information sources by differentially assigning relevance to the different features. Since noisy features will receive low IG weights, this also implies that it is much more noise-tolerant.

6 Conclusion

We have analysed the relationship between Back-off smoothing and Memory-Based Learning and established a close correspondence between these two frameworks which were hitherto mostly seen as unrelated. An exception is the use of similarity for alleviating the sparse data problem in language modeling (Essen & Steinbiss, 1992; Brown *et al.*, 1992; Dagan *et al.*, 1994). However, these works differ in their focus from our analysis in that the emphasis is put on similarity between *values* of a feature (e.g. words), instead of similarity between patterns that are a (possibly complex) combination of many features.

The comparison of MBL and Back-off shows that the two approaches perform smoothing in a very similar way, i.e. by using estimates from more general patterns if specific patterns are absent in the training data. The analysis shows that MBL and Back-off use exactly the same type of data and counts, and this implies that MBL can safely be incorporated into a system that is explicitly probabilistic. Since the underlying k-NN classifier is a method that does not necessitate any of the common independence or distribution assumptions, this promises to be a fruitful approach.

A serious advantage of the described approach, is that in MBL the back-off sequence is specified by the used similarity metric, without manual intervention or the estimation of smoothing parameters on held-out data, and requires only one parameter for each feature instead of an exponential number of parameters. With a feature-weighting metric such as Information Gain, MBL is particularly at an advantage for NLP tasks where conditioning events are complex, where they consist of the fusion of different information sources, or when the data is noisy. This was illustrated by the experiments on PP-attachment and POS-tagging data-sets.

Acknowledgements

This research was done in the context of the "Induction of Linguistic Knowledge" research programme, partially supported by the Foundation for Language Speech and Logic (TSL), which is funded by the Netherlands Organization for Scientific Research (NWO). We would like to thank Peter Berck and Anders Green for their help with software for the experiments.

References

D. Aha, D. Kibler, and M. Albert. 1991. Instance-based Learning Algorithms. *Machine Learning*, Vol. 6, pp. 37–66.

L.R. Bahl, F. Jelinek and R.L. Mercer. 1983. A Maximum Likelihood Approach to Continuous Speech Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. PAMI-5 (2), pp. 179-190.

Peter F. Brown, Vincent J. Della Pietra, Peter V. deSouza, Jennifer C. Lai, and Robert L. Mercer. 1992. Class-based N-gram Models of Natural Language. *Computational Linguistics*, Vol. 18(4), pp. 467-479.

Claire Cardie. 1994. *Domain Specific Knowledge Acquisition for Conceptual Sentence Analysis*, PhD Thesis, University of Massachusets, Amherst, MA.

Claire Cardie. 1996. Automatic Feature Set Selection for Case-Based Learning of Linguistic Knowledge. In *Proc. of the Conference on Empirical Methods in Natural Language Processing*, May 17-18, 1996, University of Pennsylvania.

Stanley F.Chen and Joshua Goodman. 1996. An Empirical Study of Smoothing Techniques for Language Modelling. In *Proc. of the 34th Annual Meeting of the ACL*, June 1996, Santa Cruz, CA, ACL.

Kenneth W. Church and William A. Gale. 1991. A comparison of the enhanced Good-Turing and deleted estimation methods for estimating probabilities of English bigrams. *Computer Speech and Language*, Vol 19(5), pp. 19-54.

M. Collins. 1996. A New Statistical Parser Based on Bigram Lexical Dependencies. In *Proc. of the 34th Annual Meeting of the ACL*, June 1996, Santa Cruz, CA, ACL.

M. Collins and J. Brooks. 1995. Prepositional Phrase Attachment through a Backed-Off Model. In *Proceedings of the Third Workshop on Very Large Corpora*, Cambridge, MA.

S. Cost and S. Salzberg. 1993. A weighted nearest neighbour algorithm for learning with symbolic features. *Machine Learning*, Vol. 10, pp. 57–78.

Walter Daelemans and Antal van den Bosch. 1992. Generalisation Performance of Backpropagation Learning on a Syllabification Task. In M. F. J. Drossaers & A. Nijholt (eds.), TWLT*3: Connectionism and Natural Language Processing.* Enschede: Twente University. pp. 27–37.

Walter Daelemans. 1995. Memory-based lexical acquisition and processing. In P. Steffens (ed.), *Machine Translation and the Lexicon.* Springer Lecture Notes in Artificial Intelligence, no. 898. Berlin: Springer Verlag. pp. 85–98

Walter Daelemans. 1996. Abstraction Considered Harmful: Lazy Learning of Language Processing. In J. van den Herik and T. Weijters (eds.), *Benelearn-96. Proceedings of the 6th Belgian-Dutch Conference on Machine Learning.* MATRIKS: Maastricht, The Netherlands, pp. 3-12.

Walter Daelemans, Jakub Zavrel, Peter Berck, and Steven Gillis. 1996. MBT: A Memory-Based Part of Speech Tagger Generator. In E. Ejerhed and I. Dagan (eds.) *Proc. of the Fourth Workshop on Very Large Corpora*, Copenhagen: ACL SIGDAT, pp. 14-27.

Walter Daelemans, Antal van den Bosch, and Ton Weijters. 1997. IGTree: Using Trees for Compression and Classification in Lazy Learning Algorithms. In D. Aha (ed.) *Artificial Intelligence Review*, special issue on Lazy Learning, Vol. 11(1-5).

Ido Dagan, Fernando Pereira, and Lillian Lee. 1994. Similarity-Based Estimation of Word Cooccurrence Probabilities. In *Proc. of the 32nd Annual Meeting of the ACL*, June 1994, Las Cruces, New Mexico, ACL.

S.A. Dudani. 1981 The Distance-Weighted k-Nearest Neighbor Rule *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. SMC-6, pp. 325-327.

Ute Essen, and Volker Steinbiss. 1992. Coocurrence Smoothing for Stochastic Language Modeling. In *Proc. of ICASSP*, Vol. 1, pp. 161-164, IEEE.

Slava M. Katz. 1987. Estimation of Probabilities from Sparse Data for the Language Model Component of a Speech Recognizer *IEEE Transactions on Acoustics, Speech and Signal Processing*, Vol. ASSP-35, pp. 400-401, March 1987.

David M. Magerman. 1994. *Natural Language Parsing as Statistical Pattern Recognition*, PhD Thesis, Department of Computer Science, Stanford University.

Hwee Tou Ng and Hian Beng Lee. 1996. Integrating Multiple Knowledge Sources to Disambiguate Word Sense: An Exemplar-Based Approach. In *Proc. of the 34th Annual Meeting of the ACL*, June 1996, Santa Cruz, CA, ACL.

J. .R. Quinlan. 1986. Induction of Decision Trees. *Machine Learning*, Vol. 1, pp. 81–206.

J. .R. Quinlan. 1993. C4.5*: Programs for Machine Learning*. San Mateo, CA: Morgan Kaufmann.

Adwait Ratnaparkhi. 1996. A Maximum Entropy Part-Of-Speech Tagger. In *Proc. of the Conference on Empirical Methods in Natural Language Processing*, May 17-18, 1996, University of Pennsylvania.

A. Ratnaparkhi, J. Reynar and S. Roukos. 1994. A maximum entropy model for Prepositional Phrase Attachment. In *ARPA Workshop on Human Language Technology*, Plainsboro, NJ.

Christer Samuelsson. 1996. Handling Sparse Data by Successive Abstraction In *Proc. of the International Conference on Computational Linguistics* (COLING'96), August 1996, Copenhagen, Denmark.

Hinrich Schütze. 1994. Distributional Part-of-Speech Tagging. In *Proc. of the 7th Conference of the European Chapter of the Association for Computational Linguistics* (EACL'95), Dublin, Ireland.

C. Stanfill and D. Waltz. 1986. Toward memory-based reasoning. *Communications of the* ACM, Vol. 29, pp. 1213–1228.

Ralph Weischedel, Marie Meteer, Richard Schwartz, Lance Ramshaw, and Jeff Palmucci. 1993. Coping with Ambiguity and Unknown Words through Probabilistic Models. *Computational Linguistics*, Vol. 19(2). pp. 359-382.

D. Wettschereck, D. W. Aha, and T. Mohri. 1997. A Review and Comparative Evaluation of Feature-Weighting Methods for Lazy Learning Algorithms. In D. Aha (ed.) *Artificial Intelligence Review*, special issue on Lazy Learning, Vol. 11(1-5).

Jakub Zavrel and Jorn B. Veenstra. 1995. The Language Environment and Syntactic Word-Class Acquisition. In C.Koster and F.Wijnen (eds.) *Proc. of the Groningen Assembly on Language Acquisition (GALA95).* Center for Language and Cognition, Groningen, pp. 365-374.