Probabilistic semantic analysis of speech

Jürgen Haas, Joachim Hornegger, Richard Huber, Heinrich Niemann

Universität Erlangen-Nürnberg,
Lehrstuhl für Mustererkennung (Informatik 5)
Martensstraße 3, D-91058 Erlangen, Germany
E-mail: {haas,hornegger,huber,niemann}@informatik.uni-erlangen.de

Abstract. This paper presents a new probabilistic approach to semantic analysis of speech. The problem of finding the semantic contents of a word chain is modeled as the problem of assigning semantic attributes to words. The discrete assignment function is characterized by random vectors and its probabilities. By computing the best of all possible statistically modeled assignments, we get the semantic contents of a word chain and along with it a semantic segmentation. The introduced general statistical framework has to deal with incomplete data estimation problems. These are solved applying the Expectation Maximization algorithm. We show that the well–known hidden Markov models result from the suggested theory as a specialization. Experiments prove that this approach works quite well in the domain of train–time–table inquiries for German IC/EC–train connections.

1 Introduction: problem, motivation, and related work

The ultimate goal in speech understanding is to extract the meaning and the intention out of the user's utterance and to react in an appropriate way rather than to recognize it exactly word by word. Usually, parsers are used to extract the meaning out of the word chain for the actual application. Looking at the history of speech processing shows that statistical methods and models are a very powerful and promising approach in speech processing and analysis [6].

We propose the use of a probabilistic approach applying Bayesian classification and the decision rule with minimal error for the probabilistic interpretation of word chains. A new statistical modeling scheme for the generation of word chains, under the assumption of special semantic meanings, is introduced. The involved parameters of the stochastic models can be estimated automatically. The learning stage corresponds to incomplete data estimation problems which are solved using the Missing Information Principle and the associated Expectation Maximization algorithm (EM algorithm) [1]. The experimental results prove the efficiency and the practical use of this novel approach to semantic analysis.

The suggested statistical framework is motivated by results discussed in [3] and [5]. In contrast to [5], our approach is not restricted to first order statistical dependencies in the underlying statistical process. We can deal with dependencies of arbitrary order, both in automatic learning and classification. Compared with [3] we estimate parameters for a stochastic assignment function without using the length of word chains or the number of semantic segments.

2 Statistical modeling of understanding

We postulate that speech understanding is done by the assignment of semantic attributes C_k to the words of a word chain $\boldsymbol{w}$. Each word chain is segmented in several parts characterizing one semantic attribute (e.g. "from Bonn" as *departure city*) and its value (e.g. "Bonn"). For the actual application, the considered semantic attributes are represented as a set $\{C_1, \ldots, C_N\}$ describing parameters which influence the system's reaction. These attributes can be quite general (e.g. *time, date*) or application dependent (e.g. *departure city*). It is obvious that a sentence could also contain parts without any information necessary for the current application. Therefore, a semantic attribute C_0 is introduced, which describes these parts by the `NIL` attribute. This attribute defines the so–called *"filler parts"* of the sentence. Our approach of describing the intention of a word chain is quite similar to that described in [3]. Fig. 1 shows an example for the desired assignment of semantic attributes to words.

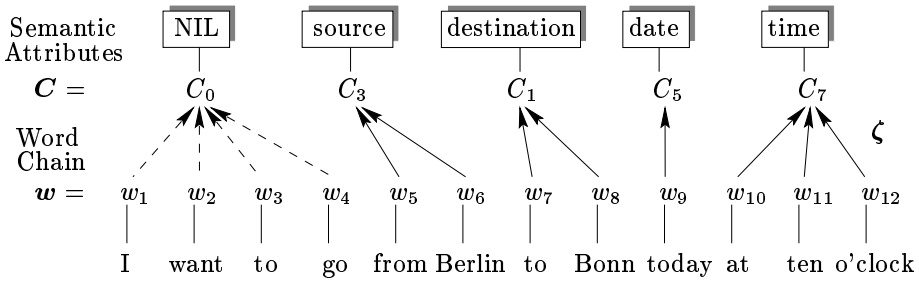

Fig. 1. Optimal assignment of semantic attributes C_k to words w_j for the sentence *"I want to go from Berlin to Bonn today at ten o'clock."*

The problem of understanding a word chain $\boldsymbol{w} = [w_j]_{1 \le j \le n}$, where w_j denotes the j–th word, is now transformed into the problem of extracting the semantic attributes and their values out of the sentence. To provide a suitable mathematical setting, we introduce a statistical modeling of speech understanding based on two tied stochastic processes: one models the appearance of semantic attributes, the other characterizes the assignment of words and attributes. Herein, each possible semantic attribute C_k $(k = 0, 1, \ldots, N)$ is associated with a stochastic process that generates single words. The mass function $p(w_j|C_k)$ is the probability that the word w_j occurs and is associated with the attribute C_k. An assignment function ζ maps each element w_j of the word chain $\boldsymbol{w}$ to the index of the corresponding semantic attribute, i.e., $\zeta(w_j) = k_j \in \{0, 1, \ldots, N\}$. For word chains $\boldsymbol{w}$, the random vector $\boldsymbol{\zeta}$ induced by the assignment function ζ is defined as the vector of all assignments for the words in $\boldsymbol{w}$: $\boldsymbol{\zeta}(\boldsymbol{w}) = (\zeta(w_1), \ldots, \zeta(w_n))^T = (k_1, \ldots, k_n)^T$. This assignment vector $\boldsymbol{\zeta}$ allows the definition of a probability mass function $p(\boldsymbol{\zeta}(\boldsymbol{w}))$ on the set of all correspondences of words and semantic attributes. The

probabilities are subject to the constraint $\sum_{\boldsymbol{\zeta}} p(\boldsymbol{\zeta}(\boldsymbol{w})) = 1$. For the example in Fig. 1 the described assignment vector $\boldsymbol{\zeta}$ is:

$$\boldsymbol{\zeta}(\boldsymbol{w}) = (0,0,0,0,3,3,1,1,5,7,7,7)^T \quad . \tag{1}$$

Using the stochastic modeling of attributes and the probabilities of assignments, we compute the conditional probability $p(\boldsymbol{w} \mid \boldsymbol{C})$ for observing $\boldsymbol{w}$ assuming the set of semantic attributes $\boldsymbol{C}$. This probability measure is rewritten with help of the non–observable assignment function $\boldsymbol{\zeta}$ and its discrete probabilities. Standard probability theory provides the equation

$$p(\boldsymbol{w}, \boldsymbol{\zeta} \mid \boldsymbol{C}) \quad = \quad p(\boldsymbol{\zeta}) p(\boldsymbol{w} \mid \boldsymbol{C}, \boldsymbol{\zeta}) \quad = \quad p(\boldsymbol{\zeta}) p(w_1, \ldots, w_n \mid \boldsymbol{C}, \boldsymbol{\zeta}) \quad . \tag{2}$$

The semantic assignment vector $\boldsymbol{\zeta}$ is not part of the training data. We only have a semantic annotation along with the sentences not an explicit alignment of attributes to words, so to say, for each utterance we know, which attributes can be found, but we do not know the words that are responsible for the occurence of them. For that reason, we make use of the statistical modeling of the assignment, and integrate out all possible assignments $\boldsymbol{\zeta}$ to the attributes C_k $(k = 0, 1, \ldots, N)$. Thus, we get the marginal probability $p(\boldsymbol{w} \mid \boldsymbol{C})$ out of $p(\boldsymbol{w}, \boldsymbol{\zeta} \mid \boldsymbol{C})$ by summation over these assignments, i.e.,

$$p(\boldsymbol{w} \mid \boldsymbol{C}) \quad = \quad \sum_{\boldsymbol{\zeta}} p(\boldsymbol{\zeta}) p(\boldsymbol{w} \mid \boldsymbol{C}, \boldsymbol{\zeta}) \quad . \tag{3}$$

The probability measure for the word chain $\boldsymbol{w}$ can be factorized in measures for observing the current word w_j having seen the predecessor words $w_1, \ldots, w_{j-1}$. Additionally, the assignment function gives the corresponding semantic attribute $C_{\zeta(w_j)}$ for the word and the conditional probability is reduced to the observation of a special word under the assumption of one semantic attribute. That gives us the possibility to factorize over all semantic attributes, pick out the words assigned to a special C_k (all w_j with $\zeta(w_j) = k$) and compute the probability for observing this word with the model for the actual attribute i.e., the model determined by the assignment function.

$$\begin{aligned} p(\boldsymbol{w} \mid \boldsymbol{C}) = \sum_{\boldsymbol{\zeta}} p(\boldsymbol{\zeta}) p(\boldsymbol{w} \mid \boldsymbol{C}, \boldsymbol{\zeta}) &= \sum_{\boldsymbol{\zeta}} p(\boldsymbol{\zeta}) \prod_{k=0}^{N} \left(\prod_{\substack{j=1 \\ \zeta(w_j)=k}}^{n} p(w_j \mid w_1, \ldots, w_{j-1}, C_k) \right) \\ &= \sum_{\boldsymbol{\zeta}} p(\boldsymbol{\zeta}) \prod_{j=1}^{n} p(w_j \mid w_1, \ldots, w_{j-1}, C_{\zeta(w_j)}) \end{aligned} \tag{4}$$

The semantic segmentation of the word chain $\boldsymbol{w}$ in attribute-dependent parts can be found by searching for the best assignment vector $\boldsymbol{\zeta}$ associated with the word sequence.

3 Learning the parameters of semantic assignment

We have to estimate the discrete probabilities $p(\boldsymbol{\zeta})$ attached to assignment vectors and the probabilities $p(w_j|C_l)$ $(1 \leq j \leq n$ and $0 \leq l \leq N)$ for a given semantic attribute C_l to generate words w_j. The training data include a semantic description with the attributes realised within the word chain $\boldsymbol{w}$. An explicit alignment between words and attributes is not available. Thus, the computation of discrete probabilities leads to an incomplete data estimation problem. The probability $p(\boldsymbol{\zeta})$ for the assignment vector and the attribute dependent probability $p(w_j|C_l)$ for observing the word w_j have to be estimated without observing the assignment function ζ in the training set. This incomplete data estimation problem can be solved applying the EM algorithm.
We have to estimate probabilities $p(\boldsymbol{\zeta}) = p(l_1, l_2, \ldots, l_n)$ with $l_j \in \{0, 1, \ldots, N\}$. For simplicity and complexity reasons we assume that the statistical dependency of assignments is of order g. For this purpose, we can decompose the probability for an assignment vector $\boldsymbol{\zeta}$ into the conditional probabilities

$$\begin{aligned} p(\boldsymbol{\zeta}) = p(l_1, l_2, \ldots, l_n) &= p(l_1) \cdot p(l_2 \mid l_1) \cdot p(l_3 \mid l_1 l_2) \cdot \ldots \cdot p(l_n \mid l_1 l_2 \ldots l_{n-1}) \\ &= p(l_1) \cdot p(l_2 \mid l_1) \cdot \ldots \cdot p(l_g \mid l_1 l_2 \ldots l_{g-1}) \prod_{k=g+1}^{n} p(l_k \mid l_{k-g} \ldots l_{k-1}) \quad . \end{aligned} \tag{5}$$

In accordance with hidden Markov models, we introduce the shorter notation $a_{l_{i_1} l_{i_2} \ldots l_{i_g}}$ for the conditional probability $p(l_{i_g} \mid l_{i_1} l_{i_2} \ldots l_{i_{g-1}})$ and get

$$\begin{aligned} p(\boldsymbol{\zeta}) &= p(l_1) \cdot p(l_2 \mid l_1) \cdot \ldots \cdot p(l_g \mid l_1 l_2 \ldots l_{g-1}) \prod_{k=g+1}^{n} p(l_k \mid l_{k-g} \ldots l_{k-1}) \\ &= a_{l_1} \cdot a_{l_1 l_2} \cdot \ldots \cdot a_{l_1 l_2 \ldots l_g} \prod_{k=g+1}^{n} a_{l_{k-g} \ldots l_k} \quad . \end{aligned} \tag{6}$$

Setting the order of statistical dependency for the assignment vector to $g = 1$, the transition probabilities a_{ij} for HMM result from the above formalism.

The second part of the factorization of equation (4) expresses the probability $p(w_j \mid w_1, \ldots, w_{j-1}, C_{\zeta(w_j)})$ for observing the word w_j after having seen the words $w_1 \ldots w_{j-1}$ and assigning w_j to the semantic attribute with index number $\zeta(w_j)$. This probability is simplified by ignoring the dependence for seeing a word w_j of the predecessor words to $p(w_j \mid C_{\zeta(w_j)})$. With these assumptions and notations, equation (4) is:

$$\begin{aligned} p(\boldsymbol{w} \mid \boldsymbol{C}) = &\sum_{\boldsymbol{\zeta}} p(\boldsymbol{\zeta}) \prod_{j=1}^{n} p(w_j \mid w_1, \ldots, w_{j-1}, C_{\zeta(w_j)}) \\ = &\sum_{l_1, l_2, \ldots, l_n} a_{l_1} \cdot a_{l_1 l_2} \cdot \ldots \cdot a_{l_1 l_2 \ldots l_g} \prod_{k=g+1}^{n} a_{l_{k-g} \ldots l_k} \prod_{j=1}^{n} p(w_j \mid C_{l_j}) \end{aligned} \tag{7}$$

The iterative estimation formulas for the required discrete probabilities can be computed applying the EM algorithm. A detailed derivation can be found in [4].

Assume, M test sentences ${}^1\boldsymbol{w}, {}^2\boldsymbol{w}, \ldots, {}^M\boldsymbol{w}$ are available for training purposes. The probabilities associated with the assignment function can be iteratively estimated using the following formulas, wherein i and $i+1$ denote the i–th and $(i+1)$–st iteration steps:

$$a_{l_1}^{(i+1)} = \frac{1}{M} \sum_{\varrho=1}^{M} \frac{\sum\limits_{\zeta(\varrho w_1)=l_1} p^{(i)}({}^\varrho\boldsymbol{w}, \boldsymbol{\zeta}|\boldsymbol{C})}{p^{(i)}({}^\varrho\boldsymbol{w}|\boldsymbol{C})} \quad , \tag{8}$$

$$a_{l_1 l_2}^{(i+1)} = \frac{\sum\limits_{\varrho=1}^{M} \sum\limits_{\zeta(\varrho w_1)=l_1, \zeta(\varrho w_2)=l_2} \frac{p^{(i)}({}^\varrho\boldsymbol{w}, \boldsymbol{\zeta}|\boldsymbol{C})}{p^{(i)}({}^\varrho\boldsymbol{w}|\boldsymbol{C})}}{\sum\limits_{\varrho=1}^{M} \sum\limits_{\zeta(\varrho w_1)=l_1} \frac{p^{(i)}({}^\varrho\boldsymbol{O}, \boldsymbol{\zeta}|\boldsymbol{C})}{p^{(i)}({}^\varrho\boldsymbol{w}|\boldsymbol{C})}} \quad , \tag{9}$$

$$a_{l_1, l_2, \ldots, l_g}^{(i+1)} = \frac{\sum\limits_{\varrho=1}^{M} \sum\limits_{\zeta(\varrho w_1)=l_1, \zeta(\varrho w_2)=l_2, \ldots, \zeta(\varrho w_g)=l_g} \frac{p^{(i)}({}^\varrho\boldsymbol{w}, \boldsymbol{\zeta}|\boldsymbol{C})}{p^{(i)}({}^\varrho\boldsymbol{w}|\boldsymbol{C})}}{\sum\limits_{\varrho=1}^{M} \sum\limits_{\zeta(\varrho w_1)=l_1, \zeta(\varrho w_2)=l_2, \ldots, \zeta(\varrho w_{g-1})=l_{g-1}} \frac{p^{(i)}({}^\varrho\boldsymbol{w}, \boldsymbol{\zeta}|\boldsymbol{C})}{p^{(i)}({}^\varrho\boldsymbol{w}|\boldsymbol{C})}} \quad , \tag{10}$$

and

$$a_{l_{k-g}, \ldots, l_k}^{(i+1)} = \frac{\sum\limits_{\varrho=1}^{M} \sum\limits_{k=g+1}^{{}^\varrho n} \sum\limits_{\zeta(\varrho w_{k-g})=l_{k-g}, \ldots, \zeta(\varrho w_k)=l_k} \frac{p^{(i)}({}^\varrho\boldsymbol{w}, \boldsymbol{\zeta}|\boldsymbol{C})}{p^{(i)}({}^\varrho\boldsymbol{w}|\boldsymbol{C})}}{\sum\limits_{\varrho=1}^{M} \sum\limits_{k=g+1}^{{}^\varrho n} \sum\limits_{\zeta(\varrho w_{k-g})=l_{k-g}, \ldots, \zeta(\varrho w_{k-1})=l_{k-1}} \frac{p^{(i)}({}^\varrho\boldsymbol{w}, \boldsymbol{\zeta}|\boldsymbol{C})}{p^{(i)}({}^\varrho\boldsymbol{w}|\boldsymbol{C})}} \quad . \tag{11}$$

Obviously, these equations are generalizations of the well–known Baum–Welch reestimation formulas. They can be used for statistical dependencies of arbitrary order g. The same holds for the discrete probabilities which characterize word productions, if the semantic attribute is known ($1 \leq j \leq n$ and $0 \leq l_j \leq N$):

$$p^{(i+1)}(w_j|C_{l_j}) = \frac{\sum\limits_{\varrho=1}^{M} \sum\limits_{k=1}^{{}^\varrho n} \sum\limits_{\zeta({}^\varrho w_k)=l, {}^\varrho w_k=w_j} \frac{p^{(i)}({}^\varrho\boldsymbol{w}, \boldsymbol{\zeta}|\boldsymbol{C})}{p^{(i)}({}^\varrho\boldsymbol{w}|\boldsymbol{C})}}{\sum\limits_{\varrho=1}^{M} \sum\limits_{k=1}^{{}^\varrho n} \sum\limits_{\zeta({}^\varrho w_k)=l} \frac{p^{(i)}({}^\varrho\boldsymbol{w}, \boldsymbol{\zeta}|\boldsymbol{C})}{p^{(i)}({}^\varrho\boldsymbol{w}|\boldsymbol{C})}} \quad . \tag{12}$$

For an efficient computation of the estimates, generalized versions of the forward–backward–algorithms are required. For the discussion of implementation details and computational aspects we recommend [4].

4 Experiments & Results

For our experiments we use the database described in [2]. These data were collected with the Erlanger Train-Time-Table Dialogue System EVAR. For training purposes, we have 9823 sentences along with their semantic annotation, the number of different sentences is 3873. Some of the sentences appear quite often in the database, e.g. the one word sentence *"Ja"* (Yes) is seen 1660 times and a single *"Nein"* (No) 929 times due to the realized dialog strategy in our information retrieval system. For test purposes, we use 4715 sentences also collected with EVAR. Some of the test sentences were already seen during training, but still 2113 sentences are new. In the training set some very frequent multiple sentences can be found, e.g. the above mentioned *"Ja"* is seen 798 times, *"Nein"* 498 times. The training set consists of 2339 different sentences. The words in the training and test sentences are the observables for the probabilistic processes modeling semantic attributes. We have 1021 different words.

In the experiments reported here we use the semantic annotation to build four respectively twelve semantic attributes to be detected. In the first experiment we want to distinguish between the four attributes `CITY`, `DATE`, `TIME` and `NIL` where the attribute `NIL` should model those parts of sentences that do not belong to any other attribute, e.g. the single word utterances *"Ja"* and *"Nein"* should be assigned to `NIL`. In the second experiment we use the following twelve attributes:

1. `CITY`	5. `POFDAY`	9. `TIME`
2. `SOURCECITY`	6. `RELDAY`	10. `RELTIME`
3. `GOALCITY`	7. `WEEKDAY`	11. `MARKER`
4. `DATE`	8. `SPECIAL`	12. `NIL`

Most attributes are self-explanatory. `POFDAY` is the abbreviation for 'part-of-day' and is used for describing time intervals like "in the morning", "around lunch time", etc. `RELDAY` denotes dates given in a relative manner, e.g. "today" or "tomorrow", `RELTIME` is used for relative time expressions like "earlier" or "as late as possible". The attribute `SPECIAL` is used for dates given as legal holidays like "Easter" or "Christmas", `MARKER` includes all relevant dialog markers, e.g. "Yes", "No", "Thank you" and even swear-words.

For the two sets of semantic attributes we use ergodic models where each state represents one attribute. As initialization for the assignment probabilities we use uniform probabilities. The initialization of the output probabilities for the states is done by counting the words in those sentences in which the corresponding attribute appears. Therefore we take a counter for each word initialized with 1 (for avoiding the value 0 as output probability) and increase this counter every time we see this word in a sentence expressing the interesting attribute.

The initialized models are iteratively trained with the above described training set of approximately 10 000 sentences. Then for each sentence of the test set the best sequence of states is computed. For first order dependencies we use the Viterbi algorithm, for second order dependencies the algorithm described in [4], which is a generalized Viterbi algorithm. The results of these experiments are shown in Table 1.

	4 attributes		12 attributes	
	HMM	G2HMM	HMM	G2HMM
Accurate sentences	3906	4283	2753	2773
Wrong sentences	809	432	1962	1942
Accurate Detections	83 %	91 %	58 %	59 %
	Insertions			
Sentences	343	253	1862	1834
	Deletions			
Sentences	506	253	715	753

Table 1. Accuracy and Error-Rates for models with statistical dependency of order $g = 1$ (HMM) and order $g = 2$ (G2HMM) with four and twelve semantic attributes

We accept only those sentences as correct where all semantic attributes that are annotated in the reference are automatically detected and there is no attribute more than the annotated ones. Insertions are those errors where the model aligns an attribute not to be found in the reference, a deletion occurs if at least one of the reference attributes does not appear in the alignment result. The sum of insertions and deletions must not match the number of wrong sentences as there are sentences with insertions and deletions and those are counted twice.

Problems arise within this evaluation scheme when we look at the attribute `NIL`. This attribute is not included in the reference annotation as long as there is semantical relevant information in the word chain. Therefore, `NIL` could never be detected as deleted in word chains with other semantic attributes and would always be an insertion. For that reason, it is not necessary to count the insertions on this semantic attribute. If we look at those sentences that are completely irrelevant in the domain, we have an empty semantic annotation for it and identify this empty reference as the `NIL` attribute. In those sentences, this attribute can be deleted.

The results in the Table 1 show that the models with order $g = 2$ have better recognition rates than those of order $g = 1$, because they use a wider context of the assignment function to decide upon the actual word w_j to which attribute it should be aligned. The improvement for 12 attributes is smaller than the one for 4 attributes because of the increase of parameters to be estimated and the small amount of training data.

The number of insertions is clearly higher than the number of deletions. This fact can be explained with the rather bad modeling of filler parts in the `NIL` state. If we look at the initialization procedure we see that we count all the words in a sentence including a special attribute. Therefore, the count of words like *"Ich"* (I) or *"fahren"* (go to) is, e.g. in the experiment with 4 attributes, even for the attribute `CITY` very high because a lot of people call the EVAR system and start with a sentence like *"Ich will nach München fahren."* (I want to go to Munich). As the `NIL` state is only initialized and trained with sentences without meaning in

the application domain at all those words are very rare and therefore their output probability for the `NIL` attribute is lower than the one for the attribute `CITY`. As consequence, words like the mentioned *"Ich"* or *"fahren"* will be aligned to `CITY` even in a sentence expressing anything else, e.g. *"Ich will um neun Uhr fahren."* (I want to go at nine o'clock). Further investigations have to examine whether these insertion errors reduce the capabilities and performance of a complete system using the probabilistic approach to semantic analysis as much as deletions. The deletion of an attribute could never be withdrawn whereas the insertion could be detected in the next step of finding the corresponding value for the attribute and therefore be corrected.

5 Summary and Conclusions

In this paper we describe, how a Bayesian framework and the optimal decision rule can be applied to the problem of extracting the meaning from a word chain.This could be done by assigning semantic attributes to the words and extracting the attribute-dependent parts of the word chain. The problem of finding the semantic contents is statistically modeled with an unknown assignment function. The parameters describing the assignment function can be estimated using the EM algorithm. The semantic analysis is done by finding the best alignment of semantic attributes to the words applyinh a generalized version of the well–known Viterbi algorithm.. As the results with accuracy from 59 % for twelve attributes and 91 % for four attributes show, this probabilistic approach for the semantic analysis of speech works quite well in our domain for German IC/EC-train connections.

References

1. A.P. Dempster, N.M. Laird, and D.B. Rubin. Maximum Likelihood from Incomplete Data via the EM Algorithm. *Journal of the Royal Statistical Society, Series B (Methodological)*, 39(1):1–38, 1977.
2. W. Eckert, E. Nöth, H Niemann, and E.G. Schukat-Talamazzini. Real Users Behave Weird — Experiences made collecting large Human–Machine–Dialog Corpora. In *Proc. of the ESCA Tutorial and Research Workshop on Spoken Dialogue Systems*, pages 193–196, Vigsø, Denmark, June 1995.
3. M. Epstein, K. Papineni, S. Roukos, T. Ward, and S. Della Pietra. Statistical natural language understanding using hidden clumpings. In *Proc. Int. Conf. on Acoustics, Speech, and Signal Processing*, pages 176–179, Atlanta, 1996.
4. J. Hornegger. *Statistische Modellierung, Klassifikation und Lokalisation von Objekten.* Shaker, Aachen, 1996.
5. R. Pieraccini and E. Levin. A learning approach to natural language understanding. In *NATO-ASI, New Advances & Trends in Speech Recognition and Coding*, volume 1, pages 261–279, Bubion (Granada), Spain, 1993.
6. H. Stahl, J. Müller, and M. Lang. An efficient top–down parsing algorithm for understanding speech by using stochastic syntactic and semantic models. In *Proc. Int. Conf. on Acoustics, Speech, and Signal Processing*, pages 397–400, Atlanta, 1996.

This article was processed using the LaTeX macro package with LLNCS style