Architecture and Search Organization for Large Vocabulary Continuous Speech Recognition

Stefan Ortmanns, Lutz Welling, Klaus Beulen, Frank Wessel, Hermann Ney

Lehrstuhl für Informatik VI, RWTH Aachen
D-52056 Aachen, Germany

Abstract. This paper gives an overview of an architecture and search organization for large vocabulary, continuous speech recognition (LVCSR at RWTH). In the first part of the paper, we describe the principle and architecture of a LVCSR system. In particular, the issues of modeling and search for phoneme based recognition are discussed. In the second part, we review the word conditioned lexical tree search algorithm from the viewpoint of how the search space is organized. Further, we extend this method to produce high quality word graphs. Finally, we present some recognition results on the ARPA North American Business (NAB'94) task for a 64 000-word vocabulary (American English, continuous speech, speaker independent).

1 Introduction

During the last decade, the performance of automatic systems for continuous speech recognition has been drastically improved. This progress has been achieved by improving both the statistical modeling techniques and the search strategies so that more complex knowledge sources could be handled. In this paper, we address the search problem in continuous speech recognition. The characteristic features of the presentation given in this paper are:

- The baseline strategy is the time-synchronous one-pass algorithm.
- The time-synchronous concept is extended towards a tree organization of the pronunciation lexicon so that the search effort is significantly reduced.
- By further extension of the one-pass search strategy, it has been possible to construct word graphs.
- We present experimental results on the 64,000-word North American Business (NAB'94) task, which demonstrate the high performance and high efficiency of the word graph method.

2 Architecture of the Speech Recognition System

Every approach to automatic speech recognition faces the problem of taking decisions in the presence of ambiguity and context and of modeling the interdependence of these decisions at various levels. If it were possible to recognize phonemes (or words) with a very high reliability, it would not be necessary to

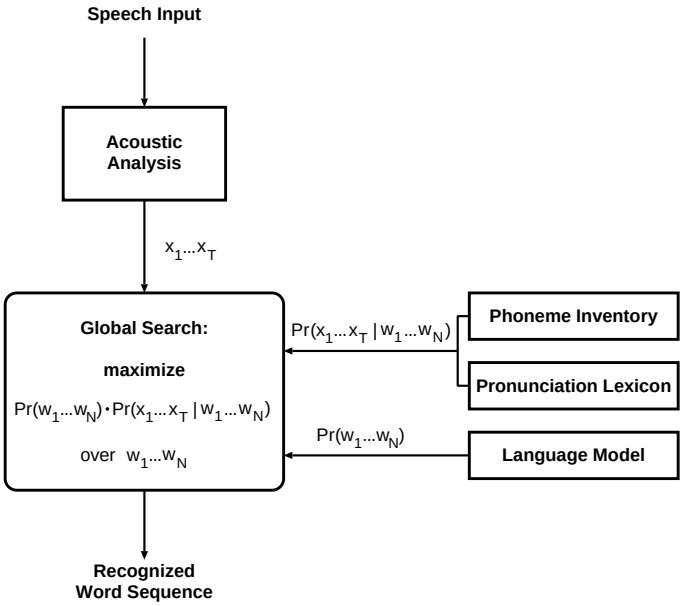

Fig. 1. Application of Bayes decision rule to speech recognition.

rely heavily on delayed decision techniques, error correcting techniques and statistical methods. In the near future, this problem of reliable and virtually error free phoneme or word recognition without using high-level knowledge is unlikely to be solved for large vocabulary continuous speech recognition. As a consequence, the recognition system has to deal with a large number of hypotheses about phonemes, words and sentences, and ideally has to take into account the "high-level constraints" as given by syntax, semantics and pragmatics. Statistical decision theory tells us how to minimize the probability of recognition errors [Bahl et al. 1983]:

> Maximize the posterior probability $Pr(w_1...w_N|x_1...x_T)$, i.e. determine the sequence of words $w_1...w_n...w_N$ (of unknown length N) which has most probably caused the observed sequence of acoustic vectors $x_1...x_t...x_T$ (over time $t = 1...T$) which are derived from the speech signal in the preprocessing step of acoustic analysis.

By applying Bayes' theorem on conditional probabilities, the problem can be written in the following form: Determine the sequence of words $w_1...w_n...w_N$ which maximizes

$$Pr(w_1...w_n...w_N) \cdot Pr(x_1...x_t...x_T|w_1...w_n...w_N). \quad (1)$$

This so-called Bayes decision rule is illustrated in Fig. 1. The first term in the optimization criterion, the a-priori probability of word sequences $Pr(w_1...w_N)$,

is independent of the acoustic observations and is completely specified by the language model. It incorporates restrictions on how to concatenate words of the vocabulary to form whole sentences and thus captures syntactic and semantic restrictions. The acoustic-phonetic modeling is reflected by the second term. $Pr(x_1...x_T|w_1...w_N)$, the acoustic probability, is the conditional probability of observing the acoustic vectors $x_1...x_T$ when the speaker utters the words $w_1...w_N$. These probabilities are estimated during the training phase of the recognition system. For a large vocabulary system, there is typically a set of basic recognition units that are smaller than whole words. Examples of these so-called subword units are phonemes, demisyllables or syllables. The word models are then obtained by concatenating the subword models according to the phonetic transcription of the words in a pronunciation dictionary. The decision on the spoken words must be taken by an optimization procedure which combines information of several knowledge sources: the language model, the acoustic-phonetic models of single phonemes, and the pronunciation dictionary. The optimization procedure is usually referred to as search in a state space defined by the knowledge sources.

3 Knowledge Sources

3.1 Phoneme and Word Units: Hidden Markov Models

As pointed out in the preceding section, the statistical approach requires the conditional probability $Pr(x_1...x_T|w_1...w_N)$ of observing an acoustic vector sequence $x_1...x_T$, given the word sequence $w_1...w_N$. These probabilities are obtained by concatenating the corresponding word models, which again are obtained by concatenating phoneme or other subword unit models according to the pronunciation lexicon. As in many other systems, these subword units are modeled by so-called Hidden Markov Models (HMM). Hidden Markov Models are stochastic finite-state automata (or stochastic regular grammars) which consist of a Markov chain of states, modeling the temporal structure of speech, and a probabilistic function for each of the states, modeling the emission and observation of acoustic vectors [Baker 1975, Bahl et al. 1983, Levinson et al. 1983]. Words are obtained by concatenating the HMM phoneme units according to the nominal phonetic transcription. For the following, it suffices to consider only the product of the emission and transition probabilities:

$$q(x_t, s|\sigma; w) = a(s|\sigma; w)\ b(x_t|\sigma; w), \tag{2}$$

which is the conditional probability that, given state σ in word w, the acoustic vector x_t is observed and the state s is reached.

3.2 Acoustic Search and Time Alignment

For an utterance to be recognized, there is a huge number of possible state sequences, and all combinations of state and time must be systematically considered. An efficient method for computing the probability $Pr(x_\tau...x_t|w)$, i.e.

that, given a word model w, the acoustic vectors $x_\tau ... x_t$ are produced and cover the time interval $\tau, .., t$, is the Baum recursion [Baker 1975, Bahl et al. 1983, Levinson et al. 1983]. We introduce an auxiliary quantity $Q(t, s; w)$:

$Q(t, s; w) :=$ probability that, for a given word w and a fixed start time τ, the acoustic vectors $x_\tau ... x_t$ are produced by state sequences going through state s.

For this quantity $Q(t, s; w)$, we have the recursive equation:

$$Q(t, s; w) = \sum_{\sigma} q(x_t, s|\sigma; w)\ Q(t-1, \sigma; w), \tag{3}$$

where we have explicitly expressed the dependence of all quantities on the word identity w. For $t = \tau$ a suitable initialization must be chosen. The summation is carried out over all states σ from which the state s can be reached. Denoting the terminal state of word w by S_w, we have for the probability that the word w produces the acoustic vectors $x_\tau ... x_t$:

$$Pr(x_\tau ... x_t|w) = Q(t, S_w; w). \tag{4}$$

The experimental results show that for continuous densities the so-called Viterbi approximation results in the same recognition performance. In lieu of summing up the contributions of all transitions, only the transition with the highest contribution is considered. Thus the Viterbi algorithm, which is no more than a dynamic programming (DP) recursion, computes the probability of the single best state sequence rather than the probability of all state sequences:

$$Q(t, s; w) = \max_{\sigma} \{\ q(x_t, s|\sigma; w)\ Q(t-1, \sigma; w)\ \}\ . \tag{5}$$

This equation basically performs a nonlinear time alignment. As will be shown later, the recursive evaluation of the best state sequence within a word will be integrated into the search for the unknown word sequence. The start time τ will then be determined implicitly by reformulating the time alignment problem at the level of word sequences rather than single words.

3.3 Language Modeling

The task of a language model is to capture the restrictions on the combinations of words due to the inherent redundancy of the language subset handled by the system. This redundancy results from the syntactic, semantic and pragmatic constraints of the recognition task and may be modeled by probabilistic or non-probabilistic ('yes/no') methods. In large vocabulary recognition tasks, bigram or trigram models have been used primarily. For a trigram model, we have the approximation:

$$Pr(w_n|w_1 ... w_{n-1}) = p(w_n|w_{n-2}, w_{n-1}). \tag{6}$$

4 Search Organization

In this section, we describe the search strategy for a recognition system that is able to handle 64 000 and more words. There are a number of other large vocabulary recognition systems which use concepts like n-best, stack decoding (A^* search) or forward-backward search. The characteristic property of the search strategy to be presented here is that it is still conceptually based on a time-synchronous beam search strategy.

4.1 Basic Concept: Time-Synchronous Beam Search

The decision on the spoken sentence is taken in the search procedure which attempts to determine the word sequence which best explains the input speech signal in terms of the given knowledge sources. The search space can be described as a huge finite-state network [Ney et al. 1992b], which consists of nodes representing a certain state in the language model and suitable types of directed arcs representing acoustic word models. By approximating the 'most likely word sequence' by the 'most likely state sequence' [Baker 1975, Bahl et al. 1983, Levinson et al. 1983], a dynamic programming search procedure allows us to compute the probabilities

$$Pr(w_1...w_N) \cdot Pr(x_1...x_T|w_1...w_N)$$

in a left-to-right fashion and to carry out the optimization over the unknown word sequence at the same time. Within the framework of the Viterbi criterion, the dynamic programming algorithm presents a closed-form solution for handling the interdependence of nonlinear time alignment, word boundary detection and word identification in continuous speech recognition [Ney 1984].

As language model, we will first use a bigram model and extend the search method later to a trigram model. In the word interior, the recursive equation is the same as introduced in the section on acoustic-phonetic modeling:

$$Q(t,s;w) = \max_{\sigma} \{ q(x_t, s|\sigma; w)\, Q(t-1, \sigma; w) \}. \tag{7}$$

To include word boundaries, we introduce a special state $s = 0$ which is used to start up a word. When encountering a potential word boundary, we have to perform the recombination over the predecessor words, which is expressed by the recursion:

$$Q(t,0;w) = \max_{v} \{ p(w|v)\, Q(t, S_v; v) \}, \tag{8}$$

where $p(w|v)$ are the conditional bigram probabilities. This equation assumes that the normal states $s = 1...S_w$ are evaluated for each word w before the start-up states $s = 0$ are evaluated. The same time index t is used intentionally, because the language model does not 'absorb' an acoustic vector. Note that the scores $Q(t,s;w)$ capture both the acoustic observation-dependent probabilities resulting from the HMM and the language model probabilities.

The sequence of acoustic vectors extracted from the input speech signal is processed strictly from left to right. The search procedure works with a time-synchronous breadth-first strategy, i.e. all hypotheses for word sequences are extended in parallel for each incoming acoustic vector. To reduce the storage requirements, it is suitable to introduce backpointers that are propagated from state to state during the dynamic programming recursion and traceback arrays so that the recognized word sequence can be recovered efficiently [Ney 1984].

4.2 Word Conditioned Lexical Tree Search Method

So far, we have considered a straightforward approach to organize the search space, which was based on the use of a bigram language model and a linear-organized pronunciation lexicon, i.e. each word is represented as a linear sequence of phonemes, independently of other words. In a 64 000-word vocabulary, there are typically many words that share the same beginning phonemes. Therefore it seems natural and very desirable for efficiency reasons to organize the pronunciation lexicon in the form of a lexical (prefix) tree [Ney et al. 1992a]. However, for a bigram language model (and other more complicated language models) in combination which such a lexical prefix tree, there is an added complication due to the fact that the identity of the hypothesized word w is known only when a leaf of the tree has been reached. Therefore, the language model probabilities can only be fully incorporated after reaching the terminal state of the second word of the bigram. Therefore, we introduce a separate copy of the lexical prefix tree for each predecessor word v so that during the search process we always know the predecessor word v when a word end w with terminal state S_w is hypothesized. To formulate the dynamic programming approach, we introduce the following two quantities [Ney 1993]:

$Q_v(t,s) :=$ overall score of the best partial path that ends at time t in state s of the lexical tree for predecessor v.

$B_v(t,s) :=$ starting time of the best partial path that ends at time t in state s of the lexical tree for predecessor v.

Both quantities are evaluated using the dynamic programming recursion for $Q_v(t,s)$:

$$\begin{aligned} Q_v(t,s) &= \max_{\sigma} \{ \, q(x_t, s|\sigma) \cdot Q_v(t-1,\sigma) \, \} \\ B_v(t,s) &= B_v(t-1, \sigma_v^{max}(t,s)) \; , \end{aligned} \tag{9}$$

where $\sigma_v^{max}(t,s)$ is the optimum predecessor state for the hypothesis (t,s) and predecessor word v. $q(x_t, s|\sigma)$ is the product of transition and emission probabilities of the Hidden Markov models used for the phonemes. The back pointers $B_v(t,s)$ are propagated according to the dynamic programming decision. Unlike the predecessor word v, the index w for the word under consideration is only needed and known when a path hypothesis reaches an end node of the lexical tree: each end node of the lexical tree is labeled with the corresponding word of

the vocabulary. Using a suitable initialization for $\sigma = 0$, this equation includes the optimization over the unknown word boundaries. At word boundaries, we have to find the best predecessor word v for each word w. To this purpose, we define:

$$H(w;t) := \max_{v} \{ p(w|v) \cdot Q_v(t, S_w) \} \quad , \tag{10}$$

where the state S_w denotes the terminal state of word w in the lexical tree. To propagate the path hypothesis into the lexical tree hypotheses or to start them up if they do not exist yet, we have to pass on the score and the time index *before* processing the hypotheses for time frame t:

$$\begin{aligned} Q_v(t-1, s=0) &= H(v; t-1) \\ B_v(t-1, s=0) &= t-1 \ . \end{aligned} \tag{11}$$

For a trigram language model, the situation is more complicated: two hypotheses about partial word sequences can only be considered to be equivalent when they do not differ in their last *two* words. Therefore the algorithm must keep track of the two non-silence predecessor words for each word. This is achieved by making a separate copy of the lexical prefix tree for each pair of non-silence predecessor words. The full technical details of the integrated search algorithm are given in [Ney 1993, Ortmanns et al. 1996].

Since full search is prohibitive, we use the time-synchronous beam search strategy, where at each time frame only the most promising hypotheses are retained. The pruning approach consists of three steps namely acoustic pruning, language model pruning and histogram pruning. These pruning steps are performed every 10-ms time frame [Steinbiss et al. 1994]. The efficiency of these pruning approach can be improved by using the so-called look-ahead techniques [Ortmanns et al. 1997], e.g. language model look-ahead [Alleva et al. 1996, Steinbiss et al. 1994] and phoneme look-ahead [Ney et al. 1992a]. In this work, we employed only what we call unigram language model look-ahead [Steinbiss et al. 1994].

4.3 Word Graph Method

The basic idea of a word graph is to represent all word sequence hypotheses whose scores are very close to the locally optimal hypothesis in the spirit of beam search. The advantage of a word graph is that a fairly good degree of decoupling between acoustic recognition at the 10-ms level and the final search at the word level using a complicated language model can be achieved. The algorithm for word graph construction is based on the so-called word pair approximation [Schwartz & Austin 1991]: Given a word pair and its ending time, the word boundary between the two words is independent of the further predecessor words. This assumption can be expressed by the word boundary equation:

$$\tau(t; w_1^n) = \tau(t; w_{n-1}^n) \quad .$$

By taking this property into account, we obtain the following algorithm for the word graph construction which fits directly into a word-conditioned search organization [Ney & Aubert 1994]:

- At each time frame t, we consider all word pairs (v, w). Using a beam search strategy, we limit ourselves to the most probable hypotheses $(t; v, w)$, i.e. word pair (v, w) with ending time t.
- For each triple $(t; v, w)$, we have to keep track of:
  - the word boundary $\tau(t; v, w)$
  - the word score $h(w; \tau(t; v, w), t)$
- At the end of the speech signal, the word graph is constructed by tracing back through the bookkeeping lists.

Given a word graph and an m-gram language model, the second-pass of the word graph method can be carried out at the sentence level using a left-to-right dynamic programming algorithm as described in [Ney & Aubert 1994]. Because the word graph generated by the acoustic recognition process can be very large, pruning methods can be applied to reduce the size of the word graph without affecting the word error rate. A detailed description of the word graph method is given in [Ortmanns, Ney & Aubert 1997].

5 Experimental Results

The experimental tests were carried out on the ARPA North American Business (NAB'94) H1 development corpus including 310 sentences with 7387 words by 10 male and 10 female speakers. 39 of the spoken words were out-of-vocabulary words. The emission probability distributions of the underlying Hidden Markov models were trained on the so-called WSJ 0 and WSJ 1 training data as described in [Dugast et al. 1995]. In all the experiments, we used a 64 000-word lexicon and a language model as described in [Wessel et al. 1997]. To study the quality and the efficiency of the word graph method, a conservatively large word graph was constructed using a bigram language model with a test set perplexity (PP) of 237. The acoustic search space (when computing the initial word graph) consisted of 64 691 active states, 18 087 active arcs and 193

Table 1. Recognition results for the word graph method on the NAB'94 H1 development data (64 000-word task, trigram language model with $PP = 172.0$; OOV rate: 0.5%)

	Graph density			Graph word error rate		Recognition word error rate	
f_{lat}	WGD	NGD	BGD	DEL / INS	GER[%]	DEL / INS	WER[%]
200	1571.5	763.6	18.3	27 / 18	2.6	146 / 134	12.3
100	517.3	224.3	12.4	31 / 17	2.7	146 / 134	12.3
70	116.6	48.1	7.1	38 / 19	3.1	146 / 134	12.3
40	12.6	6.8	2.9	69 / 35	4.6	149 / 130	12.3
20	2.7	2.1	1.6	125 / 72	8.6	161 / 116	12.5
10	1.7	1.5	1.4	168 / 98	11.5	175 / 113	13.1
1	1.3	1.3	1.2	196 / 124	14.2	199 / 127	14.3

Table 2. Recognition results for the integrated search method (64 000-word task, trigram language model with $PP = 172.0$) as a function of the search space.

Average number of active			Recognition word error rate	
States	Arcs	Trees	DEL / INS	WER[%]
3950	1164	30	151 / 161	14.2
8259	2378	50	137 / 149	13.0
16068	4593	76	136 / 144	12.4
43381	10754	129	132 / 142	12.2
60177	16904	150	132 / 145	12.2

active trees per time frame during the first pass of the two-pass search strategy and results in a word error rate of 14.0%. Then the size of the word graph was reduced by applying a pruning operation using a pruning threshold f_{lat}. For this resulting word graph, Table 1 reports the size of the word graph in terms of the word graph density (WGD), the graph word error rate (GER) and the recognition word error rate (WER), for both of which the number of word deletions (DEL) and insertions (INS) is also given. For the recognition test, a full search through the word graph was performed using a trigram language model (perplexity of 172). To verify the viability and the quality of the word graph method, we use the speech recognition results obtained for the integrated search method in combination with a trigram language model (Table 2). Table 2 shows the search space, which is given in terms of the average number (per time frame) of active states, of active arcs, active trees and the recognition word error rate. Comparing the results of the integrated method with the results of the word graph method, we can see that the integrated method leads to a slight improvement of the recognition accuracy. Nevertheless, we have to keep in mind that the integrated method does not offer the flexibility of the word graph method.

6 Conclusion

This paper has given a description of the search problem in large vocabulary continuous speech recognition. Starting with the one-pass beam search, we have presented the word conditioned search algorithm using a tree-organized pronunciation lexicon. In addition, we have used the so-called word pair approximation in the construction of very-high quality word graphs and studied its viability in recognition experiments on the ARPA 64 000-word NAB'94 task.

References

[Baker 1975] J. K. Baker: "Stochastic Modeling for Automatic Speech Understanding", in D. R. Reddy (ed.): 'Speech Recognition', Academic Press, New York, pp. 512-542, 1975.

[Alleva et al. 1996] F. Alleva, X. Huang, M.-Y Hwang: Improvements on the Pronunciation Prefix Tree Search Organization. Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing, Atlanta, GA, pp. 133-136, May 1996.

[Bahl et al. 1983] L. R. Bahl, F. Jelinek, R. L. Mercer: A Maximum Likelihood Approach to Continuous Speech Recognition. IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. 5, pp. 179-190, March 1983.

[Dugast et al. 1995] C. Dugast, R. Kneser, X. Aubert, S. Ortmanns, K. Beulen, H. Ney: Continuous Speech Recognition Tests and Results for the NAB'94 Corpus. Proc. ARPA Spoken Language Technology Workshop, Austin, TX, pp. 156-161, January 1995.

[Levinson et al. 1983] S. E. Levinson, L. R. Rabiner, M. M. Sondhi: An Introduction to the Application of the Theory of Probabilistic Functions of a Markov Process to Automatic Speech Recognition. The Bell System Technical Journal, Vol. 62, No. 4, pp. 1035- 1074, April 1983.

[Ney 1984] H. Ney: The Use of a One-Stage Dynamic Programming Algorithm for Connected Word Recognition. IEEE Trans. on Acoustics, Speech, and Signal Processing, Vol. ASSP-32, No. 2, pp. 263-271, April 1984.

[Ney et al. 1992a] Ney, H., Haeb-Umbach, R., Tran, B.-H. & Oerder, M.: Improvements in Beam Search for 10000-Word Continuous Speech Recognition. 1992 IEEE Int. Conf. on Acoustics, Speech and Signal Processing, San Francisco, CA, pp. 13-16, March 1992.

[Ney et al. 1992b] H. Ney, D. Mergel, A. Noll, A. Paeseler: Data Driven Organization of the Dynamic Programming Beam Search for Continuous Speech Recognition. IEEE Trans. on Signal Processing, Vol. SP-40, No. 2, pp. 272-281, February 1992.

[Ney 1993] H. Ney: Search Strategies for Large-Vocabulary Continuous-Speech Recognition. NATO Advanced Studies Institute, Bubion, Spain, June-July 1993, pp. 210–225, in A.J. Rubio Ayuso, J.M. Lopez Soler (eds.): 'Speech Recognition and Coding – New Advances and Trends', Springer, Berlin, 1995.

[Ney & Aubert 1994] H. Ney, X. Aubert: A Word Graph Algorithm for Large Vocabulary Continuous Speech Recognition. Proc. Int. Conf. on Spoken Language Processing, Yokohama, Japan, pp. 1355-1358, September 1994.

[Ortmanns et al. 1996] S. Ortmanns, H. Ney, F. Seide, I. Lindam: A Comparison of Time Conditioned and Word Conditioned Search Techniques for Large Vocabulary Speech Recognition. Proc. Int. Conf. on Spoken Language Processing, Philadelphia, PA, pp. 2091-2094, October 1996.

[Ortmanns et al. 1997] S. Ortmanns, A. Eiden, H. Ney, N. Coenen: Look-Ahead Techniques for Fast Beam Search. Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing, Munich, Germany, Vol. 3, pp. 1783-1786, April 1997.

[Ortmanns, Ney & Aubert 1997] S. Ortmanns, H. Ney, X. Aubert: A Word Graph Algorithm for Large Vocabulary Continuous Speech Recognition. Computer, Speech and Language, Vol. 11, No. 1, pp. 43-72, January 1997.

[Schwartz & Austin 1991] R. Schwartz, S. Austin: A Comparison of Several Approximate Algorithms for Finding Multiple (N-Best) Sentence Hypotheses. Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing, Toronto, pp. 701-704, May 1991.

[Steinbiss et al. 1994] V. Steinbiss, B.-H. Tran, H. Ney: Improvements in Beam Search. Proc. Int. Conf. on Spoken Language Processing, Yokohama, Japan, pp. 2143-2146, September 1994.

[Wessel et al. 1997] F. Wessel, S. Ortmanns, H. Ney: Implementation of Word Based Statistical Language Models. Proc. SQEL Workshop on Multi-Lingual Information Retrieval Dialogs, Pilsen, Czech Republic, pp. 55-59, April 1997.

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