STRUCTURE AND PERFORMANCE OF A DEPENDENCY LANGUAGE MODEL

Ciprian Chelba, David Engle, Frederick Jelinek, Victor Jimenez, Sanjeev Khudanpur, Lidia Mangu
Harry Printz, Eric Ristad, Ronald Rosenfeld, Andreas Stolcke, Dekai Wu

ABSTRACT

We present a maximum entropy language model that incorporates both syntax and semantics via a dependency grammar. Such a grammar expresses the relations between words by a directed graph. Because the edges of this graph may connect words that are arbitrarily far apart in a sentence, this technique can incorporate the predictive power of words that lie outside of bigram or trigram range. We have built several simple dependency models, as we call them, and tested them in a speech recognition experiment. We report experimental results for these models here, including one that has a small but statistically significant advantage ($p < .02$) over a bigram language model.

1. INTRODUCTION

In this paper, we propose a new language model to remedy two important weaknesses of the well-known Ngram method. We begin by reviewing these problems.

Let S be a sentence consisting of words $w^0 \dots w^n$, each drawn from a fixed vocabulary of size V. By the laws of conditional probability,

$$P(S) = P(w^0)P(w^1 \mid w^0) \dots P(w^n \mid w^0 \dots w^{n-1}). \quad (1)$$

Unfortunately, this decomposition, though exact, does not constitute a usable model. $P(w^i \mid w^0 \dots w^{i-1})$, the general factor in (1), requires the estimation and storage of $V^{i-1}(V-1)$ independent parameters, and since typically $V \approx 25,000$ and $n \approx 20$, this is infeasible.

Ngram models avoid this difficulty by retaining only the $N-1$ most recent words of history, usually with $N = 2$ or 3. But this approach has two significant drawbacks. First, it is frequently linguistically implausible, for it blindly discards relevant words that lie N or more positions in the past, yet retains words of little or no predictive value simply by virtue of their recency. Second, such methods make inefficient use of the training corpus, since the distributions for two histories that differ only by some triviality cannot pool data.

In this paper we present a *maximum entropy dependency language model* to remedy these two fundamental problems. By use of a *dependency grammar*, our model can condition its prediction of word w^i upon related words that lie arbitrarily far in the past, at the same time ignoring intervening linguistic detritus. And since it is a maximum entropy model, it can integrate information from any number of predictors, without fragmenting its training data.

2. STRUCTURE OF THE MODEL

In this section we motivate our model and describe it in detail. First we discuss the entities we manipulate, which are words and disjuncts. Then we exhibit the decomposition of the model into a product of conditional probabilities. We give a method for the grouping of histories into equivalence classes, and argue that it is both plausible and efficient. Since our model is obtained using the maximum entropy formalism, we describe the types of constraints we imposed. Finally, we point out various practical obstacles we encountered in carrying out our plan, and discuss the changes they forced upon us.

2.1. Elements of the Model

Our model is based upon a dependency grammar [2], and the closely related notion of a link grammar [10, 5]. Such grammars express the linguistic structure of a sentence in terms of a planar, directed graph: two related words are connected by a graph edge, which bears a label that encodes the nature of their linguistic relationship. A typical parse or linkage K of a sentence S appears in Figure 1.

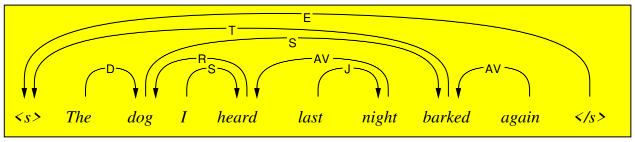

Figure 1. A Sentence S and its Linkage K. <s> and </s> are sentence delimiters.

Our aim is to develop an expression for the joint probability $P(S,K)$. In principle, we can then recover $P(S)$ as the marginal $\sum_K P(S,K)$. In practice, we make the assumption that this sum is dominated by a single term $P(S,K^\star)$, where $K^\star = \operatorname{argmax}_K P(S,K)$, and then approximate $P(S)$ by $P(S,K^\star)$.

For our purposes, every sentence S begins with the "word" <s>, and ends with the "word" </s>. We use shudder quotes because these objects are of course not really words, though mathematically our model treats them as such. They are included for technical reasons: the start marker <s> functions as an anchor for every parse, and the end marker </s> ensures that the function $P(S,K)$ sums to unity over the space of all sentences and parses.

A directed, labeled graph edge is called a *link*, and denoted L. (Formally, each L consists of a triple of parse-node tags, plus an indication of the link's direction; we depict them more simply here for clarity.) A link L that connects words y and z is called a *link bigram*, and written yLz.

Each word in the sentence bears a collection of links, emanating from it like so many flowers grasped in a hand. We refer to this collection as a *disjunct*, denoted d. A disjunct is a rule that shows how a word must be connected to other words in a legal parse. In linguist's parlance, a word and a disjunct together constitute a fully specified lexical entry.

For instance, the disjunct atop *dog* in Figure 1 means that it must be preceded by a determiner, and followed by a relative clause and the verb of which it is the subject, in that order. Intuitively, a disjunct functions as a highly specific part-of-speech tag. Note that in different sentences, or in different parses of the same sentence, a given word may bear different disjuncts, just as the word *dog* may function as a subject noun, object noun, or verb. A disjunct d is defined formally by two lists, *left*(d) and *right*(d), respectively its links to the left and right.

2.2. Decomposition of the Model

Just as w^i is the ith word of S, we write d^i for its disjunct in a given linkage. It can be shown that if a sequence $d^0 \ldots d^n$ of disjuncts is obtained from a legal linkage K, then the linkage can be uniquely reconstructed from the sequence. Thus

$$P(S, K) = P(w^0 \ldots w^n d^0 \ldots d^n) = P(w^0 d^0 \ldots w^n d^n)$$

where it is understood that this quantity is 0 if the disjunct sequence does not constitute a legal linkage. Now let us write h^i for the history at position i. That is, h^i lists the constituents of the sentence and its linkage up to but not including $w^i d^i$; explicitly $h^i = w^0 d^0 \ldots w^{i-1} d^{i-1}$. Hence by the laws of conditional probability, we have the exact decomposition

$$P(S, K) = \prod_{i=0}^{n} P(w^i d^i \mid h^i) \qquad (2)$$

A given factor $P(w^i d^i \mid h^i)$ in (2) is the probability that word w^i, playing the grammatical role detailed by d^i, will follow the words and incomplete parse recorded in h^i. Figure 2 depicts this idea.

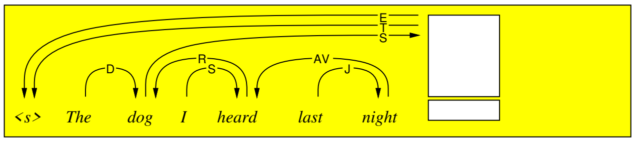

Figure 2. Meaning of $P(w^7 d^7 \mid h^7)$. h^7 is the sequence of all words and disjuncts to the left of position 7. Positions are numbered from the left, starting with 0.

2.3. Equivalence Maps of Histories

The problem is now to determine the individual probabilities in the right hand side of (2). But once again there are too many different histories, hence too many parameters.

We are driven to the solution used by Ngram modelers, which is to divide the space of possible histories into equivalence classes, via some map $\phi : h \mapsto [h]$, and then to estimate the probabilities $P(w\, d \mid [h])$. Approximating each factor $P(w^i d^i \mid h^i)$ by $P(w^i d^i \mid [h^i])$, equation (2) yields

$$P(S, K) = \prod_{i=0}^{n} P(w^i d^i \mid h^i) \approx \prod_{i=0}^{n} P(w^i d^i \mid [h^i]). \qquad (3)$$

This expedient has the advantage of coalescing, into each class $[h]$, evidence that had previously been splintered among many different histories. It has the disadvantage that the map $h \mapsto [h]$ may discard key elements of linguistic information. Indeed, the trigram model, which throws away everything but the two preceding words, leads to the approximation $P($*barked* $\mid$ *The dog I heard last night*$) \approx P($*barked* $\mid$ *last night*$)$. This is precisely what drove us to dependency modeling in the first place.

Our hope in incorporating the incomplete parse into each h^i is that the parser will identify just which words in the history are likely to be of use for prediction. To return to our example, we have a strong intuition that *barked* is better predicted by *dog*, five words in the past, than by the preceding bigram *last night*. Indeed, none of the words of the relative clause—to which *barked* bears no links—would seem to be of much predictive value.

This intuition led us to the following design decision: the map $\phi : h \mapsto [h]$ retains (1) a *finite context*, consisting of 0, 1 or 2 preceding words, depending upon the particular model we wish to build, and (2) a *link stack*, consisting of the open (unconnected) links at the current position, and the identities of the words from which they emerge. The action of this map is depicted in Figure 3, where the information retained in $[h^7]$ is rendered in black.

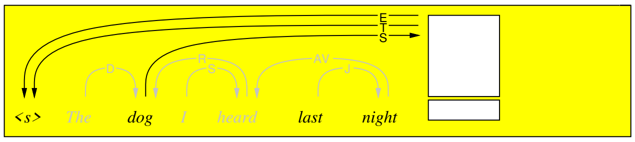

Figure 3. Meaning of $P(w^7 d^7 \mid [h^7])$. $[h^7]$ consists of the elements displayed in black.

We include the finite context in $[h]$ because trigram and bigram models are remarkably effective predictors, despite their linguistic crudeness. We include the link stack because it carries both grammar—it constrains the d that can appear in the next position, since *left*(d) must match some prefix of the stacked links—and semantics—we expect the word in the next position to bear some relation of meaning to any word it links to. Moreover, this choice for ϕ has the advantage of discarding the words and grammatical structure that we believe to be irrelevant (or at least less relevant) to the prediction at hand.

2.4. Maximum Entropy Formulation

Even with the map $h \mapsto [h]$, there are still too many distinct $[h]$ to estimate the probabilities $P(wd \mid [h])$ as ratios of counts. To circumvent this difficulty, we formulated our model using the method of constrained maximum entropy [1]. The maximum entropy formalism allows us to treat each of the numerous elements of $[h]$ as a distinct predictor variable.

By familiar operations with Lagrange multipliers, we know that the model must be of the form

$$P(w\, d \mid [h]) = \frac{e^{\Sigma_i \lambda_i f_i(w,d,[h])}}{Z(\bar{\lambda}, [h])} \qquad (4)$$

Here each $f_i(w, d, [h])$ is a *feature indicator function*, more simply *feature function* or just *feature*, and λ_i is its associated parameter. The constraint consists of the requirement

$$E_{\hat{P}}[f_i] = E_{\tilde{P}}[f_i], \qquad (5)$$

that is, it equates expectations computed with respect to two different probability distributions. On the right hand side, $\tilde{P}$ stands for $\tilde{P}(w, d, [h])$, the joint empirical distribution. On the left hand side, $\hat{P}$ is the composite distribution defined by $\hat{P}(w, d, [h]) = P(w\, d \mid [h]) \cdot \tilde{P}([h])$, where $P(w\, d \mid [h])$ is the model we are building, and $\tilde{P}([h])$ is the empirical distribution on history equivalence classes.

2.5. Model Constraints

Assuming that we retain one word of finite context, denoted h^{-1}, we recognize three different classes of feature. The first two classes are indicator functions for unigrams and bigrams respectively, and are defined as

$$\begin{aligned} f_z(w, d, [h]) &= 1 \qquad && \text{if } w = z \\ f_{yz}(w, d, [h]) &= 1 \qquad && \text{if } w = z \text{ and } h^{-1} = y \end{aligned}$$

attaining 0 otherwise. Typically, there are many such functions, distinguished from one another by the unigram or bigram they constrain. These notions are more fully described in [6, 8, 9].

The novel element of our model is the link bigram constraint. It is here that we condition the probability of the predicted word w upon linguistically related words in the past, possibly out of Ngram range. The link bigram feature function $f_{yLz}(w, d, [h])$ is defined by

$$f_{yLz}(w, d, [h]) = 1 \qquad \text{if } w = z \text{ and } [h] \sim d \text{ via } yLz$$

attaining 0 otherwise. The notation "$[h] \sim d$," read "$[h]$ matches d," means that d is a legal disjunct to occupy the next position in the parse. Specifically, if *left*(d) contains r links, then these must exactly match the links of the first r entries of the link stack of $[h]$, both lists given innermost to outermost. Figure 4 depicts matching and non-matching examples. The additional qualification "via yLz" means that at least one of the links must bear label L, and connect to word y. Returning to Figure 1, we have $f_{dog\mathrm{S}\,barked}(w^7, d^7, [h^7]) = 1$, but $f_{I\mathrm{S}\,barked}(w^7, d^7, [h^7]) = 0$.

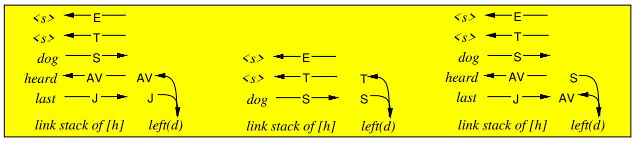

Figure 4. Matching and Non-Matching $[h]$, d Pairs. Left, center: matching. Right: non-matching. Innermost links are at the bottom of the page, outermost at the top.

2.6. Practical Considerations

Unfortunately, the model just described required too much file space to build. Since the heart of the difficulty was the number of potential futures $\{w\} \times \{d\}$, we decided to move the sequence of disjuncts into the model's history. Because the sequence $d^0 \ldots d^n$ is identified with the parse K, this yields a conditional model $P(S \mid K)$.

In this reformulation of the model, the history h^i at each position i consists of the preceding words $w^0 \ldots w^{i-1}$, and *all* disjuncts $d^0 \ldots d^n$. As before, the map $\phi : h^i \mapsto [h^i]$ retains only the finite context and the link stack at position i. By adopting this expedient, we were able to build several small but non-trivial dependency models.

Of course, we are ultimately still interested in obtaining an estimate of $P(S, K)$. This can be recovered via the identity $P(S, K) = P(S \mid K)P(K)$, but we are then faced with the computation of $P(K)$. As it happens though the parsing process generates an estimate of $P(K \mid S)$, and we may use this quantity as an approximation to $P(K)$, yielding

$$P(S, K) \approx P(S \mid K)P(K \mid S).$$

This decomposition is decidedly illegitimate, and renders meaningless any perplexity computation based upon it. However, our aim is to reduce the word error rate, and the performance improvement we realize by incorporating $P(K \mid S)$ this way is for us an adequate justification.

3. EXPERIMENTAL METHOD

In this section we discuss the training and testing of our dependency model. We describe the elements of our experimental design forced upon us by the parser, our methods for training the parser, its underlying tagger, and the dependency model itself, and how we use and evaluate the model.

3.1. Tagging and Parsing

Our model operates on parsed utterances. To obtain the required parse K of an utterance S, we used the dependency parser of Michael Collins [2], chosen because of its speed of operation, accuracy, and trainability. This parser processes a linguistically complete utterance S—what we normally think of as a sentence—that has been labeled with part-of-speech tags. It yields a parse K, and a probability $P(K \mid S)$ of this parse. The parser's need for complete, labeled utterances had three important consequences.

First, we needed some means of dividing the waveforms we decoded into sentences. We adopted the expedient of segmenting all our training and testing data by hand. Second, because the parser does not operate in an incremental, left-to-right fashion, we were forced to adopt an N-best rescoring strategy. Finally, because the parser requires part-of-speech tags on its input, a prior tagging step is required. For this we used the maximum entropy tagger of Adwait Ratnaparkhi [7], again chosen because of its trainability and high accuracy.

All training and testing data were drawn from the Switchboard corpus of spontaneous conversational English speech [4], and from the Treebank corpus, which is a hand-annotated and hand-parsed version of the Switchboard text. We used these corpora as follows. First we trained the tagger, using approximately 1 million words of hand-tagged training data. Next we applied this trained tagger to some 226,000 words of hand-parsed training data, which were disjoint from the tagger's training set; these automatically-tagged, hand-parsed sentences were then used as the the parser's training set. Finally, the trained tagger and parser were applied to some 1.44 million words of linguistically segmented training text, which included the tagger and parser training data just mentioned.

The resulting collection of sentences and their best parses constituted the training data for all our dependency language models, from which we extracted features and their expectations. For all features, we used ratios of counts, or ratios of smoothed counts, to compute empirical expectations.

3.2. Training of the Dependency Model

To find the maximum entropy model subject to a given set of constraints, we used the Maximum Entropy Modeling Toolkit [8]. This program implements the Improved Iterative Scaling algorithm, described in [3]. It proved to be highly efficient: a large trigram model, containing 12,412 unigram features, 36,191 bigram features, and 120,116 trigram features, completed 10 training iterations on a single Sun UltraSparc workstation in under 2 1/2 hours.

The one drawback of this program is the extremely large size of the file that constitutes its input. This file must specify the activating futures of each conditional feature in the model, whether or not this future was observed in the training corpus. In the worst case, for a future space of size F, and a training corpus of E observations, the events file contains $O(FE)$ bytes. For instance, the events file for the trigram model above was over 1 GB.

3.3. Testing Procedure

For testing, we used a set of 11 time-marked telephone conversation transcripts, linguistically segmented by hand, then aligned against the original waveforms to yield utterance boundaries. To implement the N-best rescoring strategy mentioned above, we first used commercially available HTK software, driven by a standard trigram language model, to generate the 100 best hypotheses, $S_1, \ldots, S_{100}$, for each utterance A. We chose this relatively small value for N to allow quick experimental turnaround.

For each hypothesis S, containing words $w^0 \ldots w^n$, we computed the best possible tag sequence $T^\star$ using the tagger, and from S and $T^\star$ together the best possible disjunct sequence $D^\star$ using the parser. Note that n, $T^\star$ and $D^\star$ taken together constitute a linkage K. In fact this threesome was our working definition of $K^\star$, the best possible

model	number of constraints			WER (%)	
	unigram	bigram	linkbg	dm	all
2g24	12,412	36,191		48.3	47.4
1g2c4	12,412		37,007	48.3	48.1
2g24c7	12,412	36,191	10,005	47.5	46.8
2g24c2	12,412	36,191	46,666	48.4	47.6
2g24c5mi	12,412	36,191	12,130	48.6	47.5

Table 1. Experimental Results.

linkage for S. (Note that the maximization of $T^\star$ from S, and then of $D^\star$ from $T^\star$ and S, is not the same as the joint maximization of $D^\star$ and $T^\star$ from S, and hence this is an approximation to $K^\star$.)

With these entities all in hand, we then rescored using the product $P(A \mid S)P(S)$, where $P(A \mid S)$ is the acoustic score, and $P(S)$ is the geometrically averaged quantity

$$P(n)^\alpha P(T \mid n, S)^\beta P(D \mid T, n, S)^\gamma P(S \mid D, T, n)^\delta \quad (6)$$

where α, β, γ and δ are experimentally-determined weights. Here $P(n)$ is an *insertion penalty*, which penalizes the decoding of an utterance as a sequence of short words or particles; $P(T^\star \mid n, S)$ is the *tagger score*; and $P(D^\star \mid T^\star, n, S)$ is the *parser score*. Finally, recalling that $T^\star$, $D^\star$ and n together constitute $K^\star$, we recognize the quantity $P(S \mid D^\star, T^\star, n)$ as precisely $P(S \mid K^\star)$, and it is here that our model finally enters the decoding process. The remaining factors of (6) constitute our estimate of $P(K^\star \mid S)$.

4. RESULTS AND CONCLUSIONS

We built and tested five models with this scheme—a baseline and four dependency models. All of our models were maximum-entropy models, trained as described above. Each of these models retained all unigrams of count ≥ 2 as constraints. They differed only in the number and nature of the additional constraints included during training, and then used as features when computing $P(S \mid K)$.

4.1. Model Details

Model 2g24, a maximum entropy bigram model, was our baseline. For 2g24 we included all unigrams, as well all bigrams of count ≥ 4. Here "bigram" is used in the usual sense of two adjacent words; we will call this an *adjacency bigram* to distinguish it from a link bigram. Thus this model does not use the parse K at all.

For our first two dependency models, 1g2c4 and 2g24c7, we labeled each link bigram with its sense only ($\leftarrow$ or $\rightarrow$), erasing the other linguistic information the link carried. For 1g2c4, we retained unigrams as above but no adjacency bigrams—that is, we included *no* finite context in the history—and instead included all link bigrams, labeled with sense only, of count ≥ 4. Thus 1g2c4 is the link bigram analog of 2g24. Intuitively, its performance relative to the baseline measures the value of link bigrams versus adjacency bigrams.

All the rest of the models we built retained unigrams and adjacency bigrams as in 2g24; they differed only in what constraints beyond these were included. For 2g24c7, we included beyond 2g24 all sense-only link bigrams of count ≥ 7. This was our best model.

For the last two models, 2g24c2 and 2g24c5mi, we did *not* erase the link label information, which comprise part-of-speech tags and parser labels, in addition to sense. Model 2g24c2 included beyond 2g24 all fully-labeled link bigrams of count ≥ 2. Finally, for model 2g24c5mi, we applied an information-theoretic measure to link selection: we included beyond 2g24 all link bigrams of count ≥ 5, for which the average link gain [11] exceeded 1 bit.

4.2. Model Performance

Table 1 above lists word error rate scores for these models. Column *dm* reports results with $\beta, \gamma = 0$, in expression (6), and α and δ fixed at nominal values. The superior performance of 2g24c7 over the baseline in this column, though small, is statistically significant according to a sign test, $p < .02$. Column *all* reports results with all exponents of (6) allowed to float to optimal values on the test suite, independently for each model.

We interpret the identical *dm* performance of 2g24 and 1g2c4 to mean that the link bigrams captured essentially the same information as regular bigrams. Moreover, the superior figures of column *all* versus *dm* confirm our intuition that the tag and parse scores are useful.

Finally, the slim but statistically significant superiority of 2g24c7 convinces us that dependency modeling is a promising if unproven idea. We intend to pursue it, constructing more elaborate models, training them on larger corpora, and testing them more thoroughly. A more thorough discussion of the methods and results presented here may be found in reference [11].

ACKNOWLEDGEMENTS

We gratefully acknowledge the contributions and assistance of Michael Collins and Adwait Ratnaparkhi, both of the University of Pennsylvania, who generously donated their software and energy to this effort. We also thank the Center for Language and Speech Processing, at Johns Hopkins University, for hosting the summer workshop that was the site of much of this work.

REFERENCES

[1] A. Berger, S. Della Pietra, V. Della Pietra, "A Maximum Entropy Approach to Natural Language Processing," *Computational Linguistics,* 1996.

[2] M. J. Collins, "A New Statistical Parser Based on Bigram Lexical Dependencies," *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics,* pages 184–191, May, 1996.

[3] S. Della Pietra, V. Della Pietra, and J. Lafferty, *Inducing Features of Random Fields,* Technical Report CMU-CS-95-144, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, May 1995.

[4] J. J. Godfrey, E. C. Holliman, and J. McDaniel, "SWITCHBOARD: Telephone Speech Corpus for Research and Development," *Proceedings of the IEEE Conference on Acoustics, Speech and Signal Processing*, Volume I, pages 517–520, San Francisco, March 1992.

[5] John Lafferty, Daniel Sleator, Davy Temperley, "Grammatical Trigrams: A Probabilistic Model of Link Grammar," *Proceedings of the 1992 AAAI Fall Symposium on Probabilistic Approaches to Natural Language,* Cambridge, MA, 1992.

[6] Raymond Lau, Ronald Rosenfeld, Salim Roukos, "Trigger-Based Language Models: A Maximum Entropy Approach," *Proceedings of the International Conference on Acoustics, Speech and Signal Processing,* II:45–48, Minneapolis, MN, April 1993.

[7] A. Ratnaparkhi, "A Maximum Entropy Model for Part-of-Speech Tagging," *Proceedings of the Conference on Empirical Methods in Natural Language Processing,* pages 133–142, May 1996.

[8] E. Ristad, "Maximum Entropy Modeling Toolkit," Technical Report, Department of Computer Science, Princeton University, 1996.

[9] R. Rosenfeld, "A Maximum Entropy Approach to Adaptive Statistical Language Modeling," *Computer Speech and Language,* pages 187–228, October 1996.

[10] Daniel D. K. Sleator and Davy Temperley, *Parsing English with a Link Grammar,* Technical Report CMU-CS-91-196, Carnegie Mellon University, School of Computer Science, Pittsburgh, PA, 1991.

[11] A. Stolcke, C. Chelba, D. Engle, V. Jimenez, L. Mangu, H. Printz, E. Ristad, R. Rosenfeld, D. Wu, F. Jelinek and S. Khudanpur, "Dependency Language Modeling," *1996 Large Vocabulary Continuous Speech Recognition Summer Research Workshop Technical Reports*, Research Note 24, Center for Language and Speech Processing, Johns Hopkins University, Baltimore, MD, April 1997. `http://www.speech.sri.com/people/stolcke/papers/ws96-report.ps.Z`