PERMUGRAM LANGUAGE MODELS

E.G. Schukat–Talamazzini[1,2] *R. Hendrych*[1] *R. Kompe*[1] *H. Niemann*[1]

[1]Univ. Erlangen-Nürnberg, Lehrstuhl für Mustererkennung (Informatik 5), Erlangen, F.R. of Germany

[2]Friedrich-Schiller-Universität, Institut für Informatik, Jena, F.R. of Germany

E–mail: `schukat@informatik.uni-erlangen.de`

ABSTRACT

In natural languages, the words within an utterance are often correlated over large distances. Long-spanning contextual effects of this type cannot be efficiently and robustly captured by the traditional N-gram approaches of stochastic language modelling. We present a new kind of stochastic grammar — the *permugram* model. A permugram model is obtained by linear interpolation of a large number of conventional bigram, trigram, or polygram models which operate on different permutations of the input word sequence under consideration. This way, stochastic dependences between word pairs or word triples lying adjacent as well as remote in the input text can be captured simultaneously without the requirement of very large N-grams. Using the permugram model, we achieved test set perplexity reductions of 5–10% compared with interpolated N-gram models, depending on the application.

1. INTRODUCTION

In natural languages, the words within an utterance are often correlated over large distances; for instance, this is the case as a result of the insertion of subordinate clauses. Another example is the tight grammatical coupling between the parts of discontinuous verbal phrases which are very common in German language:

Der Eilzug	*fährt* *kommt* *läuft*	*am Bahnsteig zwei*	*ab* *an* *ein*

Long-spanning contextual effects of this type cannot be efficiently and robustly captured by the traditional N-gram approaches (see [5] for a survey) of stochastic language modelling. Even for moderate positional distances between grammatically related words, the estimates of higher-order N-grams are required in order to bridge the interword gaps, and the model runs into the danger of combinatorial explosion and overadaptation. The same argument is true for the polygram interpolation technique [8] as well as other smoothing and backing-off procedures [7, 14], which might avoid the overadaptation effect, but are still restricted to contiguous word histories when estimating conditional word probabilities.

In order to circumvent this problem, one may try to reduce the dimensionality of the parameter space by introducing gaps or wildcards into the word N-grams under consideration. A simple heuristic is to replace conditional N-grams by weighted averages of *gappy bigrams* $P(w_t|w_{t-\tau})$. This approach was followed in [12] and [15], using different approximations for the marginal distribution $P(w_t|w_{t-\tau})$. The notion of gappy bigrams was extended to N-grams in [4], and in [11] a product expression for the approximation of conditional N-grams by their marginals was proposed. Another approach is to employ a *tree classifier* to approximate the required probabilities [1], or to configure the desired statistical (in)dependences into a causal Bayesian network [9].

In this paper, we present a new kind of stochastic grammar — the permugram model — which is obtained by linear interpolation of a large number of conventional bigram, trigram, or polygram models which operate on different permutations of the input word sequence under consideration. This way, stochastic dependences between word pairs or word triples lying non-contiguous in the input text can be captured simply by choosing the appropriate permutation — this "choice" is of course a random process — that brings the respective word items into touch.

The remainder of the paper is organized as follows: section 2 reviews the formalism of polygram models. Permutations of the input word order are introduced in the central section 3; most of this part of the paper deals with the definition of local representations of permutations ("configurations") and a generalization of discrete-output HMM's ("hidden permutation model"). Finally, sections 4 and 5 will present the experimental results and a conclusion.

2. POLYGRAM LANGUAGE MODELS

The joint distribution $P(w_1, \ldots, w_T)$ for a given sequence $\underline{w} = w_1, \ldots, w_T$ of words from a vocabulary $\mathcal{V}$ of size L may be written as a conditional decomposition

$$\begin{aligned} P(w_1, \ldots, w_T) &= \prod_{t=1}^{T} P(w_t \mid w_1, \ldots, w_{t-1}) \qquad (1) \\ &= P(w_1) \cdot P(w_2 \mid w_1) \cdot \ldots \cdot P(w_T \mid w_1, \ldots, w_{T-1}) \, . \end{aligned}$$

The conditional N-gram probabilities on the right hand side of eq. (1) are usually replaced by the maximum likelihood (ML) estimators

$$\hat{P}(w_t \mid w_1, \ldots, w_{t-1}) = \frac{\#(w_1, \ldots, w_t)}{\#(w_1, \ldots, w_{t-1})} \qquad (2)$$

where the function $\#(\cdot)$ counts the frequency of a given word sequence in the training corpus.

Unfortunately, even for small t the above frequency ratios are far from being reliable estimates. Smoothing of these statistics can be achieved by pruning the word histories or by partitioning the vocabulary into word categories. The *polygram* model [8] does without history pruning and evaluates the conditional N-gram probabilities by the linear interpolation formula

$$\begin{aligned} \tilde{P}(w_t \mid w_1, \ldots, w_{t-1}) &= \rho_t \hat{P}(w_t \mid w_1, \ldots, w_{t-1}) + \\ &\ldots + \rho_2 \hat{P}(w_t \mid w_{t-1}) + \rho_1 \hat{P}(w_t) + \rho_0 \hat{P}_{\text{uniform}} \qquad (3) \end{aligned}$$

where $\hat{P}(\cdot)$ denotes the ML estimate, $\hat{P}_{\text{uniform}}$ is the zerogram (or uniform) word probability $1/L$, and the weights $\rho_1, \ldots, \rho_i$ are iteratively optimized by the estimation-maximization (EM) algorithm [2, 6] using a cross-validation data set.

Since the interpolation weights tend to become very small for higher-order N-gram statistics if estimated globally, a dependence of the weights on the actual word history is introduced: we let $\rho_i = \rho_i(\eta)$ with

$$\eta = \max\{\nu \mid \#(w_{t-\nu}, \ldots, w_{t-1}) \neq 0\} \qquad (4)$$

In order to limit the storage requirements of the model and to avoid overadaptation to the training set, a suitable upper bound N for the maximum order of N-grams to consider in the model should be chosen.

Polygram models have been introduced in [8, 16]; a similar approach using heuristically determined interpolation weights was presented in [13], too.

3. PERMUTATIONS

Assume we are going to rearrange the word order in $w_1, \ldots, w_T$ according to a permutation

$$\pi : \{1, \ldots, T\} \xrightarrow{1-1} \{1, \ldots, T\} \qquad (5)$$

The conditional decomposition of $P(\underline{w})$ remains valid after reordering of the input sequence:

$$\begin{aligned} P(\underline{w}) &= P_\pi(\underline{w}) = P(w_{\pi(1)}, \ldots, w_{\pi(T)}) \\ &= \prod_{t=1}^{m} P(w_{\pi(t)} \mid w_{\pi(1)}, \ldots, w_{\pi(t-1)}) \end{aligned} \qquad (6)$$

Note that in the above equation the expression $w_{\pi(t)}$ is a shorthand for $o_{\pi(t)} = w_{\pi(t)}$, denoting the event that word item $w_{\pi(t)}$ happens to occur in sentence position $\pi(i)$. In subsequent formulas we will drop the random variable o provided that the subscripts of o and w coincide.

For instance, if $\pi(1) = 5$ and $\pi(2) = 3$, $P(w_{\pi(1)})$ refers to the unigram distribution of all words in the fifth sentence position, and $P(w_{\pi(2)}|w_{\pi(1)})$ denotes a statistical dependence between two non-adjacent word items moving backward in time.

As a matter of fact, the identity $P(\underline{w}) = P_\pi(\underline{w})$ becomes inapplicable as soon as the permuted N-gram probabilities are replaced by the polygram-like interpolation rule (3). Consequently, it is tempting to formulate a (perfect) permugram model by the linear combination

$$\tilde{P}(\underline{w}) = \sum_{\pi \in \mathcal{P}} \lambda_\pi \tilde{P}_\pi(\underline{w}) \qquad (7)$$

of permutation-dependent joint probability estimates, ranging over the set $\mathcal{P}$ of all possible permutations of $\{1, \ldots, T\}$. Observe that our permugram formula in fact incorporates conditional bigram probabilities $P(w_j|w_i)$ for each possible pair i, j of relative word positions; the same is true for trigrams, and tetragrams, and so forth. Consequently, stochastic dependences between word pairs or triples lying widely separated in the input text can already be modelled without the need of higher-order statistics of the word generation process.

The essential challenge in permugram modelling is to check the combinatorial explosion caused by the vast amount of theoretically possible sentence permutations. We shall particularize in the remainder of this section how this task has been solved by selecting an appropriate subset of $\mathcal{P}$ and rearranging its elements in a probabilistic finite state network.

The key idea comes from the observation that sentence probabilities with respect to similar permutations usually share several of their product terms, too. This fact is best exploited by recombining the partial probability products of competing permutations in a dynamic programming manner.

3.1. Coinciding local probabilities

At first glance it may appear that the local probability scores of permutations π and σ at time t just coincide if and only if the actual as well as all past positions of π and σ coincide, i.e., if $\pi(s) = \sigma(s)$ for each $s \leq t$ is valid. Fortunately, two formal properties of our stochastic language models, limited model order and homogenity, allow a much larger degree of recombination.

A model order of N restricts the word history of conditional probabilities to the last $N-1$ input positions; "last" refers to the process time t rather than word order. Accordingly, the condition $\pi(s) = \sigma(s)$ for $t - N < s \leq t$ is now sufficient for coincidence.

Homogenity in word position assumes that only relative word positions matter; for instance, we do not want to distinguish between probabilities $P(o_2 = w_2|o_1 = w_1)$ and $P(o_7 = w_2|o_6 = w_1)$; this assumption is used in traditional N-gram models, too. Homogenity in process time states complete independence of t which is already implicit in our notation; in other words, the value $P(o_7 = w_2|o_6 = w_1)$ is independent of the time when π has reached sentence positions 6 or 7.

3.2. Configurations

A *configuration* as defined below is meant to describe the situation we encounter when the conditional N-gram probability of a word $w_{\pi(t)}$ is to be computed. This includes information about the head position $\pi(t)$, the sentence positions of the last $N-1$ history words, and markers indicating which sentence positions have already been processed and which have not.

We define an N-gram configuration to be a finite string $\underline{c} = [c_1 \ldots c_K]$ from the alphabet $\{1, \ldots, N, \blacksquare, \square\}$. The items c_k describe the state of processing of particular sentence positions, from left to right, in the natural word order. Each of the numbers $1, \ldots, N$ have to appear exactly once in the string; N denotes the head word position, whereas each $\nu < N$ refers to the significant history positions. The marker $\square$ is attached to places which have not yet been reached by the word production process; any number (including zero) of $\square$'s may appear in $\underline{c}$, and trailing $\square$'s are omitted. The marker $\blacksquare$ indicates positions which have been encountered in a very early phase of the sentence generation process and which have now left the scope of N-gram memory. Just like the open positions $\square$, an arbitrary number of closed (or forgotten) positions $\blacksquare$ may occur; however, this time we will drop *leading* occurrences of $\blacksquare$-markers.

It is the latter arrangement that exploits the homogenity assumptions discussed above, and which makes the absolute reference positions of the markers c_k implicit, because we cannot figure out the number of pruned $\blacksquare$-positions to the left of c_1. The example configurations

$$\boxed{1\ 2\ 3} \quad \text{and} \quad \boxed{2\ \square\ \blacksquare\ 3\ 1} \qquad (8)$$

denote conditional probabilities of type $P(w_t|w_{t-2}, w_{t-1})$ or $P(w_t|w_{t-3}, w_{t+1})$, respectively. The former is a standard trigram probability. The latter one is non-standard since it incorporates gaps and order inversion; moreover, we are informed that all words left to to w_{t-4} as well as word w_{t-1} have been processed and forgotten, and words w_{t-2} as well as w_{w+2} and beyond have not yet been visited.

3.3. Hidden permutation models (HΠM)

By means of configurations, which represent the local probability contribution of word permutations, we are able to define a doubly stochastic process that generates sentences according to a convex combination of permuted polygram models. Of course, only a restricted class of subsets of all possible permutations can be realized in a finite state process of reasonable size.

A *hidden permutation model* (HΠM) consists of a set $\{S_1, \ldots, S_M\}$ of states S_i with associated configurations $\underline{c}(S_i)$. The non-deterministic sentence production process starts in S_1 (with the associated start configuration $\underline{c}(S_i) = [1]$) and moves from S_i to S_j with probability a_{ij}. The state identity at time t is hidden to the observer, but using the configuration attached to the state and its corresponding permutation, an open word position $w_{\pi(t)}$ is filled according to the selected conditional (non-standard) N-gram distribution.

It is another important feature of the HΠM that state transitions $i \rightarrow j$ are illegal unless $\underline{c}(S_j)$ is a *possible successor* of $\underline{c}(S_i)$. The successor relation between configurations can be summarized as follows: we may expand an N-gram configuration towards an $(N+1)$-gram configuration by replacing one of its $\square$'s by the new head marker $(N+1)$; note that there are infinitely many implicit $\square$ symbols in the right hand side continuation of the configuration string. From the above trigram examples, the tetragrams

$$\boxed{1\ 2\ 3\ 4} \quad \text{and} \quad \boxed{2\ 4\ \blacksquare\ 3\ 1} \tag{9}$$

can be deduced. Moreover, it is possible to reduce the configuration order in the same step. A reduction from $(N+1)$ to N is technically achieved by subtracting 1 from all positive c_i's, then replacing the unique item $c_i = 0$ by $c_i = \blacksquare$, and finally removing leading $\blacksquare$'s. For the standard tetragram above, successive order reductions lead to the standard configurations

$$\boxed{1\ 2\ 3}\ , \quad \boxed{1\ 2}\ , \quad \boxed{1}\ ; \tag{10}$$

the non-standard tetragram reduces to

$$\boxed{1\ 3\ \blacksquare\ 2\ \blacksquare}\ , \quad \boxed{2\ \blacksquare\ 1\ \blacksquare}\ , \quad \boxed{1\ \blacksquare\ \blacksquare\ \blacksquare}\ . \tag{11}$$

To summarize, we call the N'-gram configuration $\underline{c}'$ a legal successor of the N-gram configuration $\underline{c}$ if and only if $\underline{c}'$ results from one expansion of $\underline{c}$ as well as $N+1-N'$ subsequent order reductions. By the *shift* $\tau(\underline{c}, \underline{c}')$ we denote the total number of closed position markers $\blacksquare$ that have been extinguished from the intermediate configuration strings during that transformation. The shifts of standard trigrams are, for example, $\tau([123],[1234]) = 0$, $\tau([123],[123]) = 1$, and so forth.

If t was the absolute word position pointed to by the first entry c_1 of $\underline{c}$, then $t' = t + \tau(\underline{c}, \underline{c}')$ is the absolute position related to (the onset of) configuration $\underline{c}'$. Assuming that the start configuration $\underline{c}(S_1)$ of the model points to the initial sentence position w_1, a consistent sentence-configuration alignment is guaranteed during the entire HΠM word production process.

Quite similar to HMM's, the probability that the HΠM produces a word sequence $\underline{w}$ can be obtained by forward recursions; additionally, ML estimates of the transition probabilities a_{ij} (the partial permutation weights) are provided by running the EM algorithm (for further details consult [3]). A probability distribution based on an HΠM as defined above will be called a *permugram* language model.

3.4. Sentence boundaries

In order to capture the particular word statistics at or near sentence boundaries, auxiliary vocabulary items for the out-of-sentence positions are introduced, and every input sentence is expanded into an infinite word series $\underline{w}_\$ = \$ w_1 w_2 \ldots w_{T-1} w_T \$_0 \$_1 \$_2 \ldots$; with respect to the permugram model, the additional boundary markers $\$$, $\$_r$ are treated just like ordinary words.

Formally, the conditional decomposition $P_\pi(\underline{w}_\$)$ turns into an infinite probability product for the enlarged word sequences. However, it is easily shown that all but a finite number of factors may be omitted from the product; actually, each conditional probability of the form $P(o_t = w | o_s = \$_r, \ldots)$, $t \leq s - r$ assumes the value 1 if only $w = \$_{t-s+r}$ is valid and 0 else.

4. EXPERIMENTAL RESULTS

In order to compare the permugram approach with traditional N-gram models, test set perplexities have been computed for two German language modelling tasks, using a collection of standard as well as non-standard models.

4.1. Data

Two German text corpora served as basis for our experiments: the *Intercity* corpus is made up of train timetable inquiries, whilst the *Verbmobil* data contain transliterations of face-to-face negotiation dialogs [17]. The total number of dialog turns, words, and the vocabulary size of both text collections is given in Table 1.

corpus	# turns	# words	vocabulary size
Intercity	2535	13.368	711
Verbmobil	2931	65.578	2112

Table 1. Statistics of text corpora used in the experiments

Each data set is divided into three disjoint subsets: the training data ($\approx$ 80%) are used to estimate all non/standard N-gram counts, the cross-validation data ($\approx$ 10%) are fed into the EM algorithm in order to optimize the permutation weights a_{ij} and the configuration-dependent interpolation coefficients $\rho_k(\nu, S_i)$, and the test data ($\approx$ 10%) serve to compute the test set perplexities.

4.2. Permugram configurations

The standard polygram (or interpolated N-gram) model according to eqs. (1), (2), (3) is a special case of a permugram model; the HΠM's for bigrams and trigrams are shown in Figure 1. The numbers outside the configuration boxes indicate the sentence position shift related to an HΠM transition. Both models desribe a deterministic word production process, moving through the word sequence in original order (with identity permutation); from the beginning of the sentence, the configuration order is increased by one from position to position until the model order N (2 and 3 in the examples) is arrived at.

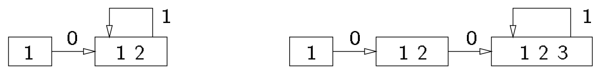

Figure 1. HΠM representation of bigrams (l) and trigrams (r)

Perhaps the simplest way to characterize (a generic set of) permutations of arbitrary length with finite means is to choose a periodic repetition in time, consisting of multiple copies of a small local index permutation pattern. An obvious candidate is the *meandric* pattern shown in Figure 2.

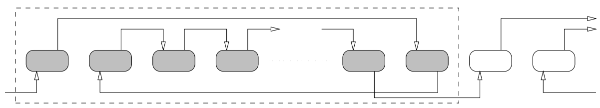

Figure 2. Meandric permutation pattern

The dashed box delimits exactly one period of the meandric permutation. Beginning from the leftmost word position t, κ places are skipped and position $t+\kappa+1$ is occupied. Subsequently, each of the places skipped before is processed stepwise from left to right until the entire sequence of word positions $t+1$ through $t+\kappa$ has been covered. The HΠM corresponding to a meandric permutation with context order $N = 2$ and gap size $\kappa = 2$ is found in Figure 3. We denote permugram models of this type by $\mathcal{M}_\kappa^N$.

In order to allow more flexibility in word order rearrangement, mixture models of meandric forms were defined. Particularly, we considered permugram models Σ_κ^N which are the convex combination of the permuted N-gram models $\mathcal{M}_\lambda^N$, $\lambda = 1, \ldots, \kappa$ together with the standard N-gram. Note that the HΠM of this mixture is easily constructed by connecting in parallel the HΠM's of all the submodels $\mathcal{M}_\lambda^N$.

The most complex HΠM structure that has been tested allows the production process to choose freely any sequence of meander periods. This model is built by providing possible transitions from each $\mathcal{M}_\lambda^N$ offset towards

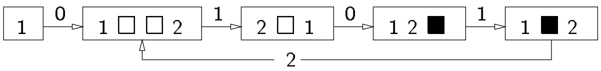

Figure 3. The HΠM structure of meander $\mathcal{M}_2^2$

each $\mathcal{M}_\lambda^N$ onset in the above mixture HΠM; we will refer to this "connected meander" model as Π_κ^N.

4.3. Perplexity figures

Test set perplexities were calculated for models with bigram ($N = 2$) and with trigram ($N = 3$) context. Besides the standard interpolated N-grams, single meander ($\mathcal{M}_\kappa^N$), mixture meander (Σ_κ^N), and connected meander (Π_κ^N) models have been run, the maximum gap size κ ranging between 1 and 6. The two rightmost columns of Table 2 leave κ unspecified; actually, results for the best performing κ are presented.

	N-gram	$\mathcal{M}_1^N$	Σ_κ^N	Π_κ^N
Intercity				
$N = 2$	26.4	38.6	24.7	25.1
$N = 3$	25.5	29.5	23.2	23.5
Verbmobil				
$N = 2$	139.7	217.1	138.0	134.3
$N = 3$	129.4	178.3	129.1	123.1

Table 2. Test set perplexities of selected permugram models

Obviously, the meandric permutations *alone* perform much worse than the standard N-gram with its original word order; this is true for $\mathcal{M}_\kappa^N$'s with larger gap $\kappa > 1$, too. However, if the word production process is allowed to choose between competing permutations as is the case in the mixture model, the test set perplexity can be substantially reduced. Note that in the Intercity domain, the bigram mixture even outperforms the trigram standard model.

For the Verbmobil corpus with its extremely long dialog turns, the periodically repeated patterns of the Σ_κ^N components do not provide sufficient flexibility to capture the intrinsic dependency structure. Much more could be gained when applying the connected meanders, which in turn showed little (additional) effect for the Intercity corpus.

5. CONCLUSION AND FUTURE WORK

We presented a new kind of stochastic grammar — the *permugram* model. A permugram model is obtained by linear interpolation of a large number of conventional bigram, trigram, or polygram models which operate on different permutations of the input word sequence under consideration. Using the permugram model, we achieved test set perplexity reductions of 5–10% compared with interpolated N-gram models, depending on the application.

In spite of this success, we feel that more dramatic improvements were possible if sentence scores could be computed in a decision-directed fashion, i.e., by evaluating the HΠM using the Viterbi algorithm instead of Baum-Welch forward decoding. Under this condition, the permugram model would decide on the best-fitting word permutation for each input sequence rather than averaging over the entire subspace spanned by the HΠM.

Unfortunately, the Viterbi permugram scores do no longer form a valid distribution, and the missing renormalization coefficient is not accessible. Hence, an experimental assessment of Viterbi permugrams on the basis of perplexity is impossible. We envisage automatic subcorpus classification [10] as an appropriate testbed for unnormalizable language models.

REFERENCES

[1] L.R. Bahl, P.F. Brown, P.V. De Souza, and R.L. Mercer. A Tree-Based Statistical Language Model for Natural Language Speech Recognition. *IEEE Trans. on Acoustics, Speech, and Signal Processing*, 37(7):1001–1008, 1989.

[2] A.P. Dempster, N.M. Laird, and D.B. Rubin. Maximum Likelihood from Incomplete Data via the EM Algorithm. *J. Royal Statist. Soc. Ser. B*, 39(1):1–22, 1977.

[3] R. Hendrych. Permutierte Polygramme zur grammatischen Sprachmodellierung. Diplomarbeit IMMD5 (Mustererkennung), Universität Erlangen, 1995.

[4] X. Huang, F. Alleva, H.W. Hon, M.Y. Hwang, K.F. Lee, and R. Rosenfeld. The SPHINX-II Speech Recognition System: an Overview. *Computer Speech & Language*, 7(2):137–148, 1993.

[5] F. Jelinek. Self-Organized Language Modeling for Speech Recognition. In A. Waibel and K.F. Lee, editors, *Readings in Speech Recognition*, pages 450–506. Morgan Kaufmann, San Mateo, CA, 1990.

[6] F. Jelinek and R.L. Mercer. Interpolated Estimation of Markov Source Parameters from Sparse Data. In E.S. Gelsema and L.N. Kanal, editors, *Pattern Recognition in Practice*, pages 381–397. North Holland, 1980.

[7] S.M. Katz. Estimation of Probabilities from Sparse Data for the Language Model Component of a Speech Recognizer. *IEEE Trans. on Acoustics, Speech, and Signal Processing*, 35(3):400–401, 1987.

[8] T. Kuhn, H. Niemann, and E.G. Schukat-Talamazzini. Ergodic Hidden Markov Models and Polygrams for Language Modeling. In *Proc. Int. Conf. on Acoustics, Speech, and Signal Processing*, volume 1, pages 357–360, Adelaide, Australia, 1994.

[9] H. Lucke. Bayesian Belief Networks as a Tool for Stochastic Parsing. *Speech Communication*, 16(1):89–118, 1995.

[10] M. Mast, E. Nöth, H. Niemann, and E.G. Schukat-Talamazzini. Automatic Classification of Speech Acts with Semantic Classification Trees and Polygrams. In *Proc. Int. Joint Conf. on Artificial Intelligence*, to appear 1995.

[11] S. Nakagawa and I. Murase. Relationship among Phoneme/Word Recognition Rate, Perplexity and Sentence Recognition and Comparison of Language Models. In *Proc. Int. Conf. on Acoustics, Speech, and Signal Processing*, volume 1, pages 589–592, San Francisco, 1992.

[12] H. Ney, U. Essen, and R. Kneser. On Structuring Probabilistic Dependences on Stochastic Language Modelling. *Computer Speech & Language*, 8(1):1–38, 1994.

[13] P. O'Boyle, M. Owens, and F.J. Smith. A Weighted Average n-Gram Model of Natural Language. *Computer Speech & Language*, 8(8):337–349, 1994.

[14] P. Placeway, R. Schwartz, P. Fung, and L. Nguyen. The Estimation of Powerful Language Models from Small and Large Corpora. In *Proc. Int. Conf. on Acoustics, Speech, and Signal Processing*, volume 2, pages 33–36, Minneapolis, 1993.

[15] R. Rosenfeld. *Adaptive Statistical Language Modelling: a Maximum Entropy Approach.* PhD thesis, Pittsburg, 1994.

[16] E.G. Schukat-Talamazzini, T. Kuhn, and H. Niemann. Speech Recognition for Spoken Dialog Systems. In H. Niemann, R. De Mori, and G. Hanrieder, editors, *Progress and Prospects of Speech Research and Technology*, number 1 in Proceedings in Artificial Intelligence, pages 110–120. Infix, 1994.

[17] W. Wahlster. Verbmobil — Translation of Face–To–Face Dialogs. In *Proc. European Conf. on Speech Technology*, volume "Opening and Plenary Sessions", pages 29–38, Berlin, 1993.