Automatic Classification of Dialog Acts with Semantic Classification Trees and Polygrams *

Marion Mast and Heinrich Niemann and Elmar Nöth and Ernst Günter Schukat-Talamazzini

Lehrstuhl für Mustererkennung, Universität Erlangen-Nürnberg, Martensstr. 3, D-91058 Erlangen, {mast,niemann,noeth,schukat}@informatik.uni-erlangen.de, Phone: +49 9131 85 7799, Fax: +49 9131 303811

Abstract. This paper presents automatic methods for the classification of dialog acts. In the VERBMOBIL application (speech-to-speech translation of face-to-face dialogs) maximally 50 % of the utterances are analyzed in depth and for the rest, shallow processing takes place. The dialog component keeps track of the dialog with this shallow processing. For the classification of utterances without in depth processing two methods are presented: Semantic Classification Trees and Polygrams. For both methods the classification algorithm is trained automatically from a corpus of labeled data. The novel idea with respect to SCTs is the use of dialog state dependent CTs and with respect to Polygrams it is the use of competing language models for the classification of dialog acts.

Keywords: automatic learning, dialog act classification, hidden polygram models, polygrams, semantic classification trees.

1 Introduction

The VERBMOBIL-System delivers translations of spontaneous speech in negotiation dialogs. The current scenario is that two participants (with German and Japanese mother tongue resp.) want to find a date for a business meeting. They speak English unless they don't know how to express the next utterance in English. In this case they press the VERBMOBIL button and continue in their mother tongue. VERBMOBIL will then deliver a translation.

The system does not act as a participant of the dialog like in information retrieval dialogs (e.g. in the ATIS domain) but keeps track of the ongoing dialog. When VERBMOBIL is activated the system has to:

- recognize the words of the actual utterance
- make a linguistic analysis

* This work was funded by the German Federal Ministry of Education, Science, Research and Technology (BMBF) in the framework of the Verbmobil Project under Grant 01 IV 102 H/0. The responsibility for the contents of this study lies with the authors.

– update the dialog history
– generate an English translation and
– create speech output by speech synthesis

When VERBMOBIL is not activated some of these tasks can be omitted. The dialog component has to fulfill 4 major tasks (see AMR95):

– provide contextual information for other VERBMOBIL-components,
– predict the next admissible dialog acts which are needed e.g. for the recognition component,
– follow the dialog when VERBMOBIL is inactive which means both participants speak English and no translation is needed and
– control clarification dialogs between VERBMOBIL and its users.

The work presented here concerns the third of these tasks: maximally 50 % of the utterances are analyzed in depth and for the rest shallow processing takes place. This is done by the segmentation and classification of the dialog acts. The dialog acts are classified with semantic classification trees (SCTs), which are trained automatically. Another method for the classification of dialog acts are polygrams.

2 Dialog Component

The dialog is seen as a sequence of dialog acts, and the dialog model in figure 1 describes admissible sequences of dialog acts (see Mai94). The dialog consists of:

– an **introduction phase** in which the participants greet, if necessary introduce each other and introduce the dialog goal (appointment scheduling)
– a **negotiation phase** in which the participants negotiate a date for a business meeting and
– a **closing phase** in which the result of the negotiation can be repeated and confirmed, and the dialog ends or the participants change to another dialog goal (this can be a date for another meeting).

The edges in the model correspond to dialog acts and the nodes to dialog states. The model consists of 5 dialog states. One further state models deviations from this model. This state can be reached from any of the other states. After the deviation (e.g. a clarification subdialog) has been processed, the dialog jumps back to the state from which the deviation state has been reached.

The dialog component (see AMR95) consists of 3 modules. The **finite state machine** provides an efficient and robust implementation of the dialog model. It checks the consistency of the uttered dialog acts with the dialog model. The **statistical layer** is an information-theoretic model (similar to language models for dialog acts) which is used for the prediction of dialog acts. The **dialog planner** keeps track of the plans of the users which means constructs and updates a discourse history. Plans are divided up into the different phases like negotiation

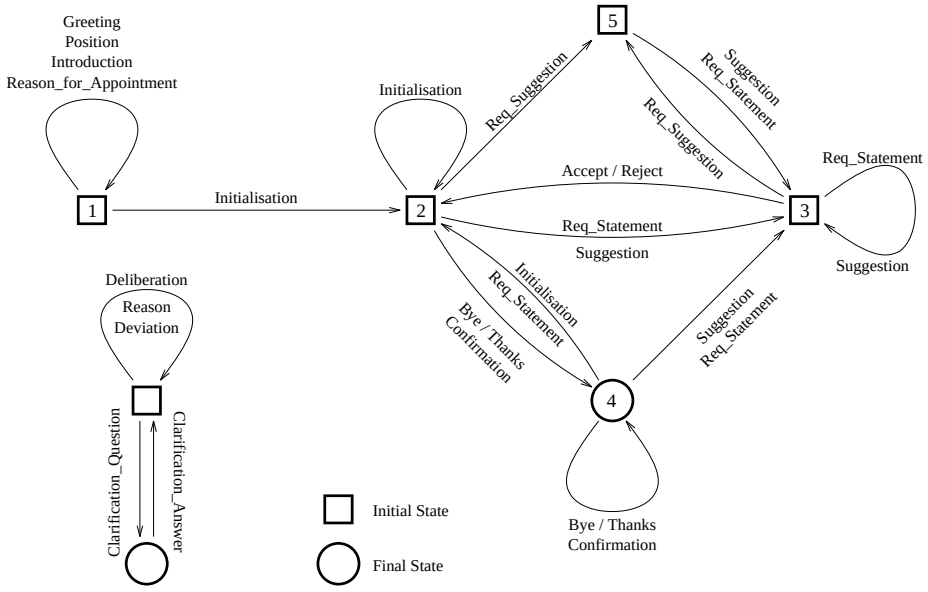

Fig. 1. The Dialog Model

and clarification. The planning process works top-down, which means that high level goals are divided into subgoals. This process ends with dialog acts as basic subgoals which can be identified from the utterances. To follow the plans of the users, the system has to identify the dialog acts.

3 Semantic Classification Trees

Classification trees are decision trees for the classification of patterns, where the decisions are made up by rules. They process the given textual information of a sequence of symbols $\boldsymbol{w} = w_1 \dots w_m$ of varying length m. The specific rules for a given task and the order of their application were trained automatically based on a corpus unlike in conventional rule based systems, where the rules are handcoded.

The structure of a binary classification tree is characterised by a binary **splitting rule** in each nonterminal node. Terminal nodes are labeled with a category and/or a scoring vector. For the automatic training of a classification tree splitting, stopping and labeling rules are needed in addition to a training corpus (see Kuh92). The binary splitting rules are YES-NO-questions. To avoid an overexpansion of the tree, which means that it's totally adapted to the train-

ing corpus and therefore looses its generality, **stopping rules** are needed to end the expansion process. These stopping rules define when a node is declared a terminal node and labeled with a category $\mathcal{C}_\kappa$ from the inventory $\{\mathcal{C}_1, \ldots, \mathcal{C}_K\}$ by a **labeling rule**.

The algorithm we used is described in Kuh92 and is used to build CHANEL – an SCT-Based Linguistic Analyzer for ATIS. CHANEL generates a SQL-like semantic representation language.

The questions (decision rules) in the nodes of the SCT refer to regular expressions consisting of keywords (represented by v) and non-zero gaps (represented by "+"). The keywords are selected automatically during the training process from the given corpus. At the root of the tree the known regular expression is <+>, representing all non-zero sequences of any symbols. "<" and ">" represent the beginning and the end of the regular expression, respectively. Questions are formed by replacing the known regular expression by another regular expression consisting of gaps and/or keywords. E. g. the first question could be: "Is the already known structure <+> of the form <+v+>, i.e. does v appear somewhere in the structure?"

Eigth question types were used. Seven of them concern keywords and are build e.g. as follows:

- **Join**: they "join" the edges of a gap together; they are of the type "Is the + equal to v?" (v is set to every item in the vocabulary during the training phase)
- **More**: they establish that there is one or "more" symbol v in a gap; they are of the type "Is the + of the form + M(v) +, where M(v) can be of the form v, vv, or v + v, and there is no v on either side of the M() within the original +".

One question is about the length of the utterance.

The rule which selects the best question or decides that it should be a leaf node is based on an impurity measure I, where the impurity

- is always non-negative,
- takes a maximum value for a node containing equal proportions of all possible categories,
- is zero for a node containing only one of all possible categories.

Therefore in node T a question is chosen which maximizes ΔI.

The algorithm uses the Gini criterion as measure of impurity which always lies between 0 and 1. If T is a certain node and $f(j \mid T)$ is the proportion of items in the node that belong to category j, then the Gini impurity $I(T)$ of the node is defined as

$$I(T) \;=\; \sum_{j \neq k} f(j \mid T) f(k \mid T) \;=\; 1 - \sum_{j} f^2(j \mid T) \qquad (1)$$

For the growing of the tree a strategy with two stages is used. First a much to large tree is grown using a simple stopping rule, e.g. that each terminal node

contains fewer than N items (N close to 1) or the maximum value of ΔI is 0. Then the tree is pruned from the leaves upwards using an independent data set. Thereafter the tree is expanded on this second data set and pruned on the first. This process is iterated until two successive pruned trees are identical (a prove that this must be is found in ?).

In figure 2 a part of an SCT trained with the algorithm described in Kuh92 for the VM-application is given. The categories which are classified are the dialog acts in the dialog model (see figure 1).

Each terminal node is labeled with a dialog act and corresponds to a keyword pattern, which is build up by the positively answered questions on the path from the root. E.g. node 21 is labeled with category 'Bye'. Node 23 points at a subtree (which is not given due to space limitations). In each nonterminal node the keyword pattern of the so far positively answered questions is expanded by a question.

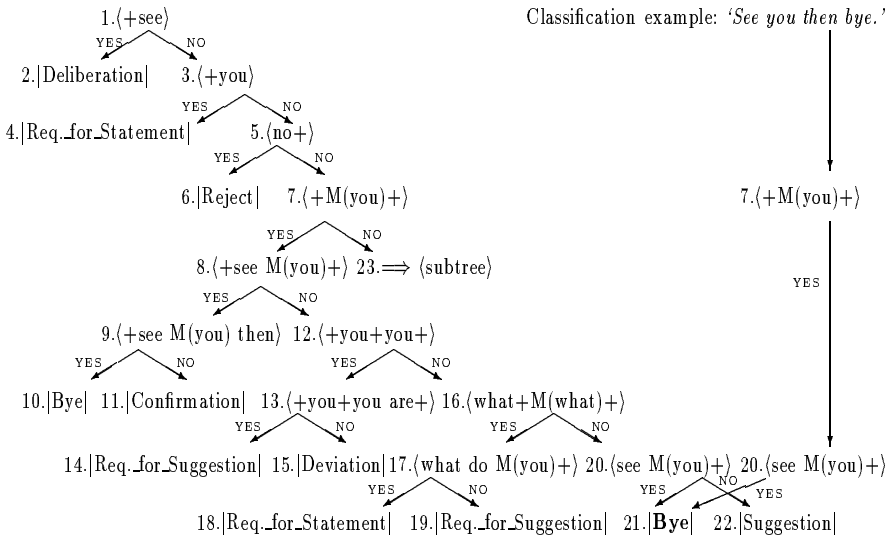

Fig. 2. Part of a classification tree for 16 dialog acts and an exemplary classification; the tree was trained automatically from 80 % of the English dialogs

4 Classification with Polygrams

Direct application of Bayes rule is an alternative approach to identify the dialog act category of a given word sequence. Bayes rule selects that particular label

κ^* for classification which maximizes the a posteriori probability

$$P(\mathcal{C}_\kappa \mid \boldsymbol{w}) \;=\; \frac{P(\boldsymbol{w}, \mathcal{C}_\kappa)}{P(\boldsymbol{w})} \;=\; \frac{P(\boldsymbol{w} \mid \mathcal{C}_\kappa) \cdot P(\mathcal{C}_\kappa)}{\sum_{\lambda=1}^{K} P(\boldsymbol{w} \mid \mathcal{C}_\lambda) \cdot P(\mathcal{C}_\lambda)} \tag{2}$$

of the input text. In the above equation, $P(\boldsymbol{w}, \mathcal{C}_\kappa)$ may be interpreted as the (joint) probability distribution of a two-stage, discrete random process. In the first stage, a dialog act category $\mathcal{C}_\kappa$ is selected from the inventory $\{\mathcal{C}_1, \ldots, \mathcal{C}_K\}$ according to the a priori probability distribution $P(\mathcal{C}_\kappa)$, $\kappa = 1, \ldots, K$. In the second stage, a word sequence $\boldsymbol{w} = w_1 \ldots w_m$ of varying length m is produced; this second generation process is controlled by the conditional production probabilities $P(\boldsymbol{w}|\mathcal{C}_\kappa)$, where κ denotes the category index selected before.

Estimates for the parameters $P(\mathcal{C}_\kappa)$ may be easily obtained from a training corpus by computing the relative frequencies of word strings in category κ. The actual problem, however, is to model the class-dependent distributions $P(\boldsymbol{w}|\mathcal{C}_\kappa)$. In our study, we decided to create stochastic grammars. For each $\mathcal{C}_\kappa$, a representative training corpus of example realizations has to be provided in order to get reasonable estimates for the conditional probabilities $P(\boldsymbol{w}|\mathcal{C}_\kappa)$. In the rest of this section we will describe the kind of language models used in our dialog act classifier: the polygram models KNST94, STKN94. Note that in the equations below explicit reference to the class names in the conditional probabilities $P(\boldsymbol{w}|\mathcal{C}_\kappa)$ has been dropped for notational convenience.

It is the task of probabilistic language modeling to find a mathematically tractable parametric form $P(\boldsymbol{w}|\boldsymbol{\Theta})$ together with reliable estimates $\hat{\boldsymbol{\Theta}}$ of its free statistical parameters $\boldsymbol{\Theta}$ which approximates the true joint distribution $P(\boldsymbol{w}) = P(w_1 \ldots w_m)$ of a given word sequence. This expression can be decomposed into a product

$$P(w_1 \ldots w_m) \;=\; P(w_1) \cdot \textstyle\prod_{n=2}^{m} P(w_n \mid \underbrace{w_1 w_2 \ldots w_{n-2} w_{n-1}}_{\text{word history}}) \tag{3}$$

of conditional n-gram probabilities. In principle, estimates for these values may be obtained by the Maximum Likelihood (ML) approach

$$\hat{P}(w_n \mid w_1 \ldots w_{n-1}) \;=\; \frac{\#(w_1 \ldots w_{n-1} w_n)}{\sum_v \#(w_1 \ldots w_{n-1} v)} \;=\; \frac{\#(w_1 \ldots w_n)}{\#(w_1 \ldots w_{n-1})} \tag{4}$$

where $\#(\cdot)$ denotes an operator that counts the word n-gram occurrences in the training data. However, since the estimate (4) becomes rapidly unreliable with increasing order n, the word history in Eq. (3) has to be restricted to a few recent words. A language model that is exclusively based on word histories of duration $n-1$ is referred to as n-gram model Jel90. In practice, only bigram models

$$P(\boldsymbol{w}) \;=\; P(w_1) \cdot \prod_{n=2}^{m} P(w_n \mid w_{n-1}) \tag{5}$$

or trigram models

$$P(\boldsymbol{w}) \;=\; P(w_1) \cdot P(w_2|w_1) \cdot \prod_{n=3}^{m} P(w_n \mid w_{n-2} w_{n-1}) \tag{6}$$

can be robustly trained.

The situation improves as soon as lower order statistics are recruited in order to smooth higher order ones. Consequently, the polygram model approximates the true conditional n-gram distributions by successively reducing the word history and computing a convex combination

$$\begin{aligned} \tilde{P}(w_n \mid w_1 \ldots w_{n-1}) \;=\; & \lambda_0 \cdot \tfrac{1}{L} \;+\; \lambda_1 \cdot \hat{P}(w_n) \;+\; \lambda_2 \cdot \hat{P}(w_n|w_{n-1}) \\ & +\; \textstyle\sum_{i=2}^{n} \lambda_i \cdot \hat{P}(w_n \mid w_{n-i+1} \ldots w_{n-1}) \end{aligned} \tag{7}$$

of the related ML estimates. The interpolation weights λ_i are usually assumed dependent on some statistics of the word history $w_1 \ldots w_{n-1}$. Our choice was to create a functional dependence based on the essential width

$$\eta \;=\; \max \{\nu \mid \#(w_{n-\nu} \ldots w_{n-1}) > 0\} \tag{8}$$

of the n-gram context. The weight vectors defined this way can be systematically optimized with respect to a cross validation data set using the EM algorithm JM80. In contrast to the nonlinear recursive "back-off" procedure in Kat87 the polygram model enables smoothing of conditional n-grams with a low but non-zero number of occurrences, too; moreover, the (linear) interpolation weights are systematically designed in order to maximize the model-dependent validation set likelihoods.

It is evident from the zerogram component $1/L$ of the interpolation formula (7) (where L denotes the size of the vocabulary) that the smoothed probabilities $\tilde{P}(\cdot)$ will get positive values even if the higher order ML estimators disappear. As a consequence, we may evaluate the right hand side of the decomposition formula (3) without artificially cutting down the word histories of conditional n-grams provided the ML estimators in Eq. (4) are replaced by the smoothed counterparts in Eq. (7). Thus, a polygram language model can be expected to capture even long-spanning contextual dependences among the words of an utterance, provided the training corpus is sufficiently large and representative.

Note that this language model is required to store the complete set of training data statistics; the frequencies of all word polygrams (i.e. unigrams, bigrams, trigrams, and so forth) observed at least once in the training material are involved in the probability computations. In order to bound the complexity of the model, a certain maximum length N of n-grams considered in the interpolation formula should be set.

5 Data

For the training of SCTs and polygrams a classified set of utterances is needed. In VERBMOBIL German and English appointment scheduling dialogs were recorded and transcribed. Non-speech phenomena like breathing and noises (e.g. paper rustle) were removed from the transcription. 214 German and 56 English dialogs were labeled with the dialog acts occuring in the dialog model (see figure 1, app. 6000 German and 1600 Englisch dialog acts). For a description of all used dialog acts see Mai94.

6 Results

6.1 Classification with SCTs

For the classification of dialog acts different SCTs were grown. First one SCT for the classification of all dialog acts for the English and German data, respectively, was trained. Therefore, 80 % of the labeled data were used for training and the remaining 20 % for testing the SCT (training set $\neq$ test set). In figure 2 a part of the classification tree for the English data is given. For the classification of 16 dialog acts a recognition rate of 46 % and 59 % for the German and English data was reached, respectively. The better results for the English data could be influenced by the more uniform scenario for the English dialogs. The German dialogs were recorded at 3 sites and the scenario slightly differed. In some dialogs private appointments were scheduled, in others business appointments. This can influence the dialog structure and the realisation of utterances (see KM93). The results show a tendency for the most frequently occuring dialog acts to be best recognized. This is due to the still insufficient amount of training data.

Classification of the dialog acts for each dialog state: In each state of the dialog model (see figure 1) only a subset of all dialog acts can follow. Therefore, for each dialog state one SCT is grown which classifies only the (in this state) possible dialog acts. The recognition rates for each dialog state are given in figure 3 (The outgoing edges for state 2 and 4 are the same).

6.2 Classification with Polygrams

For both languages, eight polygram classifiers were trained. Each classifier involved 16 competing language models, one for each speech act category. The order, i.e. the maximum allowable context length N of the polygram models ranged from $N = 1$ to $N = 8$. As with the classification trees, 80 % of the training data was used to build the models, and 20 % was used as independent test set. The partitioning of the material was the same in both experiments.

The recognition rates that could be achieved with polygrams are indicated in fig.4. For the larger sized German corpus, the maximum recognition performance of 68.7 % correct decisions was obtained using the pentagram model, i.e., N-gram

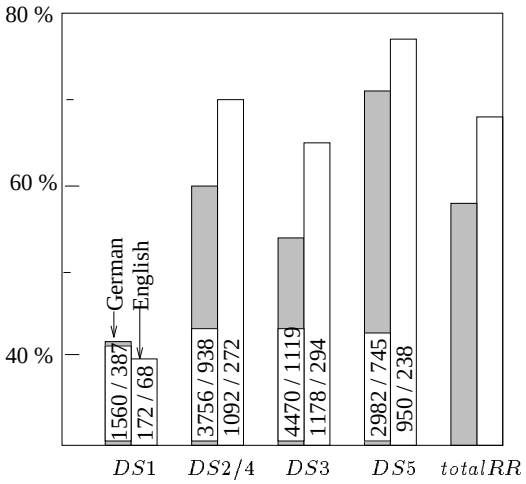

Fig. 3. Recognition rates for each dialog state (DS1-DS5), that is, one SCT per dialog state. The numbers are the samples in the test / training set.

statistics up to an order of $N = 5$. In contrast, the trigram performance (67.3 %) for the considerably smaller English corpus could not be improved further by increasing the model order.

It is interesting to observe that near-optimum recognition rates (67.1 % and 67.0 %, respectively) are already attainable with a simple bigram classifier. Moreover, even the unigram-based speech act labeling, which completely ignores word order and thus essentially amounts to score the input utterance by a words-in-a-bag strategy, enables correct identification by a rate of roughly 60 %.

These findings suggest – in fact confirm our expectation – that word identity is much more important in our task than phrase structure. The rapid saturation of recognition rate with respect to the model order gives some evidence that higher order interactions among adjacent words play a minor role. However, one can suspect that speech act categories are often triggered by associations of word tuples which are widely scattered over the input text positions. A mathematical model that deals with long-spanning contextual effects is described in STHKN95.

7 Conclusion and Future Work

In the VERBMOBIL-system shallow processing takes place when the user doesn't need a translation. In these phases the dialog module tracks the ongoing dialog. The dialog planner constructs and updates the dialog history based on the recognised dialog acts.

This paper presents two methods – SCTs and polygramms – for the automatic classification of dialog acts. The classification algorithms don't need input from syntax and semantics. The algorithms classify the utterances based on the

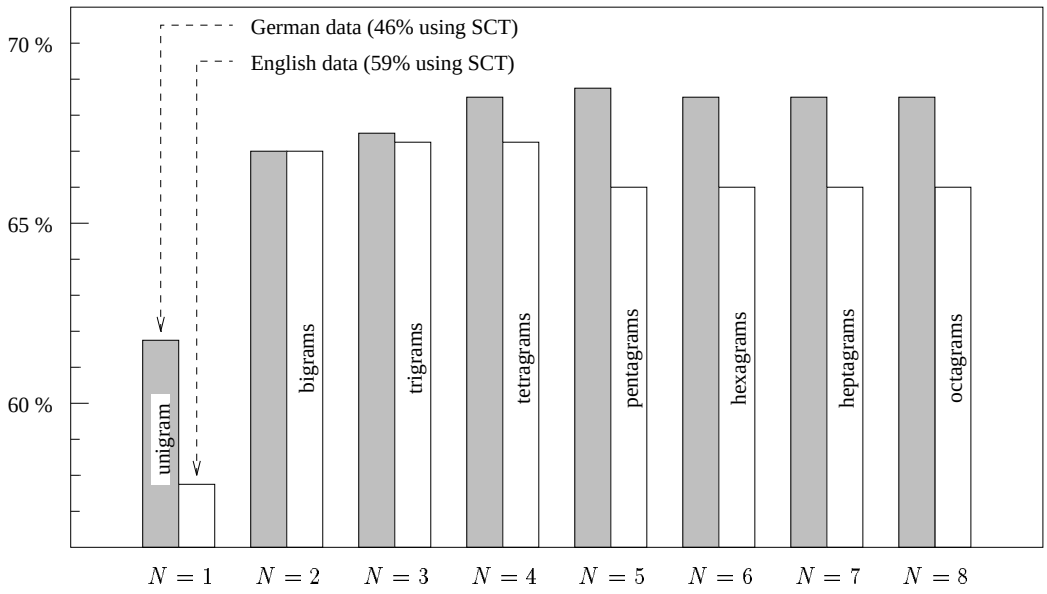

Fig. 4. Recognition rates for polyphone classifiers of different orders

textual representation. For the training of these algorithms a corpus of utterances – labeled with dialog acts – is needed.

For the classification of 16 dialog acts recognition rates with SCTs of 46 % and 59 % for the German and English data are reached, respectively. If a classification tree for each dialog state – classifying all dialog acts which can follow in this state – is trained, the recognition rates are 58 % and 68 %, respectively.

The recognition rates with polygrams are even better (see figure 5). For the German data a maximum of 69 % (with pentagrams) and for the english data a maximum of 67 % (with trigrams) was reached. Remarkable is that:

- with SCT the classification of the English data works better than of the German data, with polygrams it's vice versa.
- the polygrams are generally working better than the SCTs for this classification problem.
- polygrams even regarding only bi- or trigrams result in an almost optimal recognition rate.

7.1 SCTs Including Prosodic Information

Improvement of these results could be reached by the integration of prosodic features for the classification process. In Fis94 prosodic information was integrated in the SCT-Algorithm to find phrase boundaries in utterances. In this case the SCT is based not only on the textual information given by an utterance but

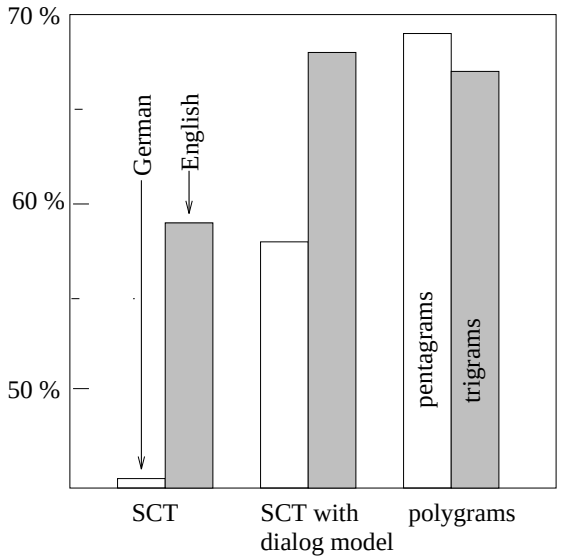

Fig. 5. Recognition rates for SCTs with and without the dialog model and polygrams.

also uses prosodic features for the classification. In the training phase questions about keywords and prosodic features can be learned. This could also be helpful for the classification of dialog acts. Useful prosodic features are **sentence mode** and **accentuation**.

For the segmentation of turns into utterances the SCT-Algorithm could be used as well. For this classification problem prosodic information especially prosodically marked phrase boundaries are even more important. First experiments show that utterance boundaries coincide with phrase boundaries in 96 %.

7.2 Hidden Polygram Models

The results indicate that application of Bayes rule using competing polygram models performs considerably better in dialog act classification than SCTs. Moreover, it becomes evident from figure 3 that additional improvement can be gained if the structural restrictions of the dialog model are exploited during analysis.

It is thus a promising idea to develop an integrated probabilistic model of dialog-driven word production. The **Hidden Polygram Model** (HPM) is a mathematical device which descibes a two-stage random process:

- The particular dialog act categories are selected according to a probabilistic finite state machine $(\boldsymbol{\pi}, \boldsymbol{A})$ with a vector $\boldsymbol{\pi}$ of initial probabilities π_i and a matrix $\boldsymbol{A}$ of transition probabilities a_{ij}.
- Each time a dialog state has been occupied, a sequence of words is produced. This second process, in turn, is controlled by a state-dependent polygram language model with the conditional output distribution $P_j(w|\boldsymbol{v})$.

Obviously, an HPM is a straightforward generalization of an ordinary discrete-valued **Hidden Markov Model** (HMM) **?** in that the probabilistic output of the model is made statistically dependent on previous output symbols (word in our application).

The parameters of the HPM can be optimized with respect to a training corpus using a modification of the Baum-Welch algorithm. Note that there are two possibilities to treat the finite state probabilities ($\boldsymbol{\pi}$, $\boldsymbol{A}$). On one hand, they can be kept fixed during training in order to fit a prespecified dialog model; an interesting alternative is to tune the parameters to the training data which amounts to an unsupervised learning of dialog structure.

References

J. Alexandersson, E. Maier, and N. Reithinger. A robust and efficient three-layered dialogue component for a speech-to-speech translation system. In *EACL 1995*, to appear 1995.

J. Fischer. Integrierte Erkennung von Phrasengrenzen und Phrasenakzenten mit Klassifikationsbäumen. Diplomarbeit, Lehrstuhl für Mustererkennung (Informatik 5), Universität Erlangen-Nürnberg, 1994.

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.

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.

S. Katz. Estimation of Probability from Sparse Data for the Language Model Component at a Speech Recognizer. *IEEE Trans. on Acoustics, Speech and Signal Processing*, ASSP-35(3):400–401, 1987.

S. Kameyama and I. Maleck. Konstellation und Szenario von Terminabsprachen, Verbmobil–Report–23–93, Dezember 1993.

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.

R. Kuhn. Keyword Classification Trees for Speech Understanding Systems. Technical report, CRIM, Montreal, Canada, 1992.

E. Maier, editor. *Dialogmodellierung in VERBMOBIL - Festlegung der Sprechhandlungen für den Demonstrator Verbmobil–Memo–31–94.* Juli 1994.

E.G. Schukat-Talamazzini, R. Hendrych, R. Kompe, and H. Niemann. Permugram language models. In *Proc. European Conf. on Speech Communication and Technology*, page (to appear), Madrid, September 1995.

E.G. Schukat-Talamazzini, T. Kuhn, and H. Niemann. Speech Recognition for Spoken Dialogue Systems. In Niemann, de Mori, and Hanrieder, editors, *Progress and Prospects of Speech Research and Technology: Proc. of the CRIM/FORWISS Workshop (München, Sept. 1994)*, pages 110–120, Sankt Augustin, 1994. infix.