Features for Tree Based Dialogue Course Management

Klaus Macherey and Hermann Ney

Lehrstuhl für Informatik VI, Computer Science Department
RWTH Aachen - University of Technology
D - 52056 Aachen, Germany
{k.macherey,ney}@informatik.rwth-aachen.de

Abstract

In this paper, we introduce different features for dialogue course management and investigate their effect on the system's behaviour for choosing the subsequent dialogue action during a dialogue session. Especially, we investigate whether the system is able to detect and resolve ambiguities, and if it always chooses that state which leads as quickly as possible to a final state that presumably meets the user's request. The criteria and used data structures are independently from the underlying domain and can therefore be used for different applications of spoken dialogue systems.

1. Introduction

Nowadays, there are numerous spoken dialogue systems for a variety of applications like inquiry systems for hotel reservations, restaurant guides, train timetable information systems, etc. [1, 2]. If several tasks and domains are to be treated by a single dialogue system without replacing or rewriting parts of the system, the need for an application independent dialogue manager arises. In order to develop an application independent dialogue manager one has to identify those steps that are equal for all of the above listed domains. These steps include:

- *asking for information*,
- *information collection and evaluation*,
- *resolving ambiguities, and*
- *information retrieval*.

Therefore, parameterizable data structures must be derived from the knowledge of each domain, and all other operations like storing and managing concept/attribute pairs describing the semantics of input data, ambiguity detection and resolution as well as dialogue course management should base on this structure. An appropriate data structure for this purpose are trees which are constructed from the knowledge of each domain. The nodes encode concepts, and the edges of a graph represent relations between the concepts [3, 4]

In this paper, we use tree based data structures in order to obtain domain independent representations. Here, the analysis of different features for dialogue course management and investigations on the system's behaviour for choosing the subsequent dialogue action based on a foregoing assessment are the main focus of attention. Especially, we investigate whether the proposed features are able to determine the best path that leads as quickly as possible to a final state which presumably meets the user's request.

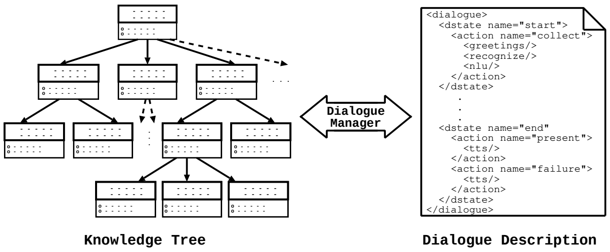

Figure 1: Basic structure of the dialogue system: The dialogue manager incorporates knowledge from the user input into the knowledge tree and determines the next dialogue action by analyzing the tree's information content.

2. Basic dialogue framework

The basic framework of the dialogue system is depicted in Figure 1. The XML-based dialogue description consists of different dialogue states, subdividing a dialogue into smaller sections. Each dialogue state may again be subdivided into several action states. During a dialogue session, the dialogue manager incorporates the knowledge from user input into the knowledge tree. If the subsequent dialogue state is not explicitly given by the dialogue description, the manager will determine the next state/action pair by analyzing the tree's information content.

2.1. Tree based representations

In order to obtain domain independent representations, we use trees as the fundamental data structure. An example is depicted in Figure 2. The tree is a knowledge representation for the specific task of a telephone directory assistance. Users can ask for information about telephone numbers, email addresses, and fax numbers of persons as well as companies. The upper half of each tree node describes the part of the dialogue which is processed by the corresponding subtree. The lower half of each node consists of a list of concepts that are associated with that specific node. As depicted in the figure, the root node's name is 'telephone inquiries' and the associated list consists of concepts which are related to different kinds of request verbalizations. The successor nodes are specifications of the corresponding parent node. During a dialogue session, concept sequences are extracted from user utterances by using a method derived from statistical machine translation [5]. Each concept together with its aligned words is then incorporated into all nodes, in which the concept's name occurs in the lists. Since all 'free' occurrences of these concepts being of the same name as the concept under consideration, are replaced by the instance given by the translated input sentence, we call the resulting tree an *instance tree*.

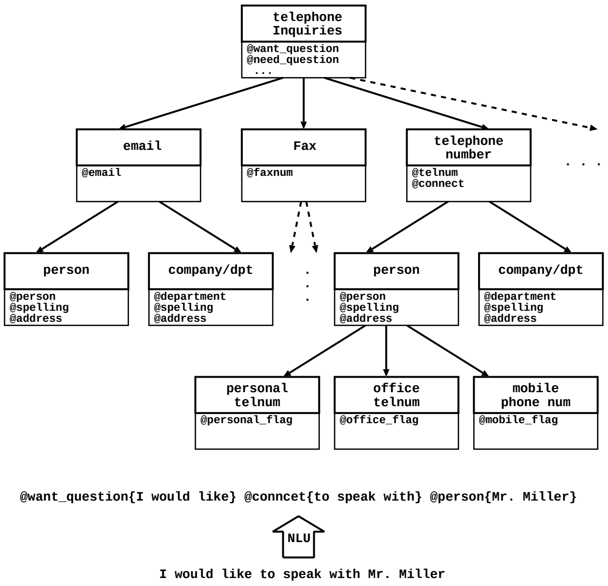

Figure 2: Tree based knowledge representation for a telephone directory assistance task. The sentence 'I would like to speak with Mr. Miller' is transformed into a concept representation via natural language understanding. After that, each concept/ attribute pair is inserted into the corresponding tree nodes. For presentation reasons, the attributes are not included in the figure.

2.2. Dialogue course management

During a dialogue session, instance trees are built from the original knowledge tree. Concept/attribute pairs which have been retrieved from the user input are incorporated into the instance tree. If there is only one path from the root to a leaf in such that all necessary concept/attribute pairs of the nodes along the path are filled, the user's request will be answered by the dialogue system. If more than one path exists, the data retrieved from the user is ambiguous and the user is required to constrain his request[1]. If there is no path from the root to a leaf, some of the nodes are still empty. In this case the system must ask for additional information in order to fill the remaining nodes. In general, there are several possibilities to continue a dialogue. Therefore, a cost function is introduced that computes a score for all nodes depending on several features which are described in the following section. Starting from the root node, the dialogue system chooses that empty node, whose corresponding subtree has minimal costs. Besides choosing the subsequent dialogue state, the dialogue system has also the possibility to verify information within a node (e.g. in case of low confidence of the recognized words), to resolve ambiguities, or to answer the user's request. The subsequent action of the dialogue system is determined by the proposed cost function.

3. Feature Functions

For the cost function, different features and knowledge sources can be taken into account. We have chosen the following node specific features.

[1]Of course, this simple strategy is not able to detect all kinds of ambiguities that may occur during a dialogue session. For example, ambiguities caused by homophones are not covered by this strategy. Additional methods for handling ambiguities are described in section 3.

3.1. Word confidence measures

In speech recognition, confidence measures can be employed for detecting possible errors and can therefore help to avoid undesirable verification turns in automatic inquiry systems. Especially, word posterior probabilities have been proven to be very effective in detecting misrecognized words [6]. The posterior word hypothesis probability $p([w, t_a, t_e]|x_1^T)$ for a word hypothesis w with starting and ending time t_a and t_e respectively, given a sequence of acoustic feature vectors x_1^T is computed in the framework of a forward-backward algorithm, summing up the posterior probabilities of all those word hypothesis sequences which contain the word hypothesis w with the same starting and ending time. The word confidence $\widetilde{p}(\cdot)$ is then computed by the following equation:

$$\widetilde{p}([w, t_a, t_e]|x_1^T) = \max_{t: t_a \le t \le t_e} \sum_{(t_i, t_j): t_i \le t \le t_j} p([w, t_i, t_j]|x_1^T)$$

For details, the reader is referred to [6]. Let $\mathcal{W}(c)$ be the set of words that are aligned to a concept c. Then the first feature for node n is defined as follows:

$$v_1(n) := \frac{1}{|\mathcal{W}(c_n)|} \prod_{w \in \mathcal{W}(c_n)} \widetilde{p}(w) \tag{1}$$

3.2. Importance and filling degree of concepts

The importance of a concept c as well as an attribute a depends on the given domain. For the telephone directory example, the last name of a person is more important than its first name. Therefore, we introduce a ranking r describing the relevance of concepts and attributes. Consequently, a person's last name is *mandatory* ($r(a) = 1$) whereas the first name is *supplementary* ($r(a) = 0$). The ranking of concepts and attributes is taken into account by summing over all concepts and attributes respectively, for which the associated attribute value is required but has not yet been 'filled' by user input. For an attribute a of a concept c, we compute:

$$\begin{aligned} f(a) &\longmapsto \begin{cases} 1 & \text{if attribute } a \text{ is assigned a value} \\ 0 & \text{otherwise.} \end{cases} \\ v_2(n) &:= \sum_{c \in \mathcal{C}(n)} \sum_{a \in \mathcal{A}(c)} \delta(r(a), 1) \cdot \delta(f(a), 0) \end{aligned} \tag{2}$$

where $\mathcal{C}(n)$ is the set of concepts of a node n and $\mathcal{A}(c)$ is the set of attributes belonging to a concept c. The importance $r(c)$ of a concept c can be derived from the maximum ranking of its related attributes. However, for some nodes, it is more convenient to fix a concept's rating independently from the related attributes. Therefore, we write:

$$\begin{aligned} f'(c) &\longmapsto \begin{cases} 1 & \text{if concept } c \text{ is sufficiently filled} \\ 0 & \text{otherwise.} \end{cases} \\ v_3(n) &:= \sum_{c \in \mathcal{C}(n)} \delta(r(c), 1) \cdot \delta(f'(c), 0) \end{aligned} \tag{3}$$

A concept c is sufficiently filled if all its related attributes with ranking $r(a) = 1$ have already been assigned values, i.e. $|\{a \in \mathcal{A}(c)|r(a) = 1, f(a) = 1\}| = \sum_{a \in \mathcal{A}(c)} r(a)$.

3.3. Ambiguity degree

Ambiguities can result from several sources: misrecognized utterances, errors during the natural language understanding step, or ambiguous user language [4]. In the context of a telephone directory assistance, ambiguities may also occur from homophones, that is, proper names which have the same pronunciation but different spellings can result in additional ambiguities that must be detected and resolved by the dialogue system. In the latter case, the speech recognizer simply constructs a list of all possible sentences with different homophone names and leaves it to the dialogue manager to resolve this kind of ambiguity. The dialogue manager constructs an instance tree for every sentence hypothesis that is delivered by the recognition system. Thus, the dialogue system does not only try to detect ambiguities within a single tree, but also tries to find ambiguities over all trees. However, if an ambiguity has been detected in a node n, this is annotated in the cost vector:

$$\begin{aligned} \mathrm{amb}(n) &\longmapsto \begin{cases} 1 & \text{if node } n \text{ has ambiguous information} \\ 0 & \text{otherwise.} \end{cases} \\ v_4(n) &:= \mathrm{amb}(n) \end{aligned} \tag{4}$$

3.4. Contradictory information

A node n contains contradictory information if an already 'filled' attribute is overwritten by a new value that is inconsistent with the old value. In this case, the following feature is set to 1:

$$\begin{aligned} \mathrm{cnt}(n) &\longmapsto \begin{cases} 1 & \text{if node } n \text{ has contradictory information} \\ 0 & \text{otherwise.} \end{cases} \\ v_5(n) &:= \mathrm{cnt}(n) \end{aligned} \tag{5}$$

Note that the size of a database query for a node containing contradictory information is always 0.

3.5. Number of SQL results

If the number of database entries as returned from a database query is too large, the user should restrict his request. If no database entry has been returned, the answer to the user's request is either not covered by the database or the user should modify some statements. The number of SQL results is taken as an additional feature for the cost vector. Let $t(q)$ be the table that is returned by the database query q. Then, the feature v_6 is defined as follows:

$$v_6(n) := |t(q_n)| \tag{6}$$

3.6. Verification of information

Since automatic speech recognition is error prone, it is reasonable to allow for verification questions in order to verify the attribute values of some concepts, particularly, if the confidence of the aligned words is low. Verification of node information is taken as an additional feature for the cost vector:

$$\begin{aligned} \mathrm{verf}(n) &\longmapsto \begin{cases} 1 & \text{if information in } n \text{ has been verified} \\ 0 & \text{otherwise.} \end{cases} \\ v_7(n) &:= \mathrm{verf}(n) \end{aligned} \tag{7}$$

4. Computing path costs

For each input sentence, a semantic analysis is performed. The concept/attribute pairs are extracted and inserted into temporary arrays for all tree nodes that are associated with these pairs. Temporary arrays are used in order to detect contradictory information. The cost vectors are computed for all nodes of an instance tree. For this purpose, the cost vectors of the leaves are propagated to their parent nodes and are combined with the parent nodes' local costs. This is done likewise for the inner nodes, resulting in a depth first traversal.

For many applications, a knowledge tree has only a moderate number of leaves. Since a tree has as many paths as leaves, there is no need to combine the costs of different paths within all parent nodes. Instead, all paths of a tree are treated separately. However, when comparing two different paths of an instance tree (or comparing different instance trees themselves), a combine function as well as a comparison function is necessary. For the combination of two cost vectors, we proceed as follows: except for the number of SQL results, we simply concatenate the lists of the respective features. For the SQL feature, we internally expand the SQL query by additional 'where' constraints that are given by the information stored in the nodes along the path. For the comparison function, we use a decision tree. At the end of the computation, each path is assigned a score corresponding to the costs that arise for continuing the dialogue along that path. If there are different possible paths that may conclude the dialogue, the dialogue manager will choose the path with the best score in order to proceed the dialogue.

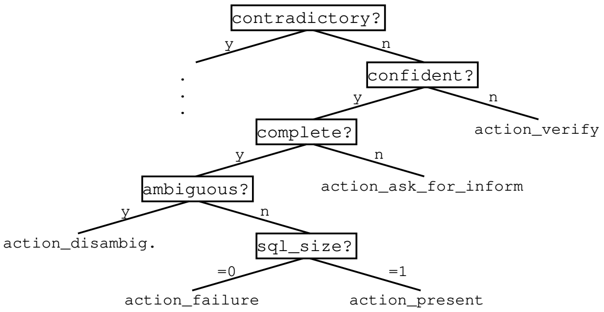

Figure 3: Excerpt of a decision tree for determining the subsequent dialogue action.

5. Selection of dialogue state/action pairs

There are different dialogue actions that can be chosen by the dialogue manager in order to continue the dialogue. Typical dialogue actions are collecting information or presenting database query results to the user. The choice of the subsequent dialogue action depends on the costs that have been computed for each path of a tree. Since the best path as well as the subsequent dialogue action are determined by decision trees, the structure of the decision trees have an immediate influence on the dialogue strategy. An excerpt of the decision tree for choosing the subsequent dialogue action is depicted in Figure 3. If the confidence of some information is lower than a given threshold, the information stored in the node with the lowest confidence is explicitly verified by further requests. If the best scored path includes ambiguous information (which is marked by the ambiguity function, cf. Eq. 4) that cannot be resolved by the system, the user is queried by the system in order to resolve this ambiguity. If the best scored path is incomplete in such that at least one node is empty, the system asks for additional information in order to complete this path. If there is a complete path with a moderate number of SQL answers, the system replies to the user's request.

Table 1: Corpus allocation for the German telephone directory assistance task.

corpus	# sess	# spks	dur. [min]	# snt	# wrds
train	131		101	1164	7310
dev	44	151	30	344	2039
eva	21		15	168	1013

Table 2: Recognition results using a class based trigram language model and confidence error rates for the development and the evaluation corpus. The confidence measure's free parameters were optimized on the development test set beforehand.

corpus	# WER [%]	baseline CER [%]	CER [%]
dev	15.7	14.3	10.5
eva	16.4	14.1	9.4

6. Results

Experiments were performed for a German in-house telephone directory assistance corpus. The objective is to answer naturally spoken requests for telephone numbers, fax numbers, and email addresses of persons as well as companies and organizations. The data for training the speech recognition system and the natural language understanding component have been recorded over several months from fixed telephones as well as wireless and mobile phones under different conditions. The conditions cover clean speech, office noise, and traffic noise. The corpus allocation is summarized in Table 1.

For the speech recognition part we used an online speech recognizer which is based on the RWTH *Verbmobil* recognizer [7]. The recognizer uses a time-synchronous beam search algorithm based on the concept of word-dependent tree copies and integrates the trigram language model constraints in a single pass. The feature vectors consist of 25 dimensions, i.e. 12 cepstral coefficients with 12 first derivatives and the second derivative of the energy.

Table 2 shows recognition results for the development and evaluation test set of the collected data. Since there was only a small subset of proper names covered by the collected data, a class based trigram was used for recognition purposes. The recognizer vocabulary has a size of 1343 words, including pronunciation variants.

We used word posterior probabilities as confidence measures. Although the recognizer performs an integrated trigram recognition, the forward/backward probabilities were computed on the generated word lattices only in bigram case for real-time aspects. All free parameters of the confidence measure, i.e. the acoustic scaling factor, the language model scaling factor, and the tagging threshold, were optimized on the separate cross-validation development test set beforehand. We used the confidence error rate (CER) in order to evaluate the quality of our confidence measure [8]. Results for the CER are given in Table 2.

The natural language understanding component was trained using a method derived from statistical machine translation, [5]. The collected data were transcribed and semantically annotated. A set of 21 concepts has been used as target language. The underlying database includes approximately 500 German and foreign proper names as well as personal related data, e.g. the office phone number, the home number, position of a person in a company, etc. For evaluating the dialogue course manager, we analyzed 40 dialogue sessions. In 88% of all cases, the dialogue manager was able to choose the correct subsequent action and finished the dialogue successfully. For text input, the attribute error rate (AER) is lower than 5%. Therefore, the interesting case to place the emphasis on is using speech as input modality. Table 3 summarizes some results for the 40 speech based dialogue sessions. In case of poor recognition performance, the confidence measure often causes the dialogue manager to explicitly verify erroneous data.

Table 3: Dialogue evaluation using speech as input modality. As evaluation criteria, the attribute error rate (AER), the percentage of correct chosen successor states, and the percentage of successfully finished dialogue sessions are used.

# dialogues	# AER [%]	choice of best state [%]	successful sessions [%]
40	18.4	88.4	90.0

7. Acknowledgments

This work was partially funded by Ericsson Eurolab Deutschland GmbH. The authors wish to thank Drs. S. Dobler, K. Reinhard, and J. Junkawitsch for their support.

8. References

[1] H. Aust, M. Oerder, F. Seide, and V. Steinbiss, "The Philips automatic train timetable information system," *Speech Communication*, vol. 17, pp. 249–262, November 1995.

[2] S. Seneff and J. Polifroni, "A New Restaurant Guide Conversational System: Issues In Rapid Prototyping For Specialized Domains," in *Proceedings of ICSLP*, vol. 2, pp. 665–668, October 1996.

[3] S. Russell and P. Norvig, *Artificial Intelligence - A modern Approach*. Upper Saddle River, NJ: Prentice Hall International Editions, 1995.

[4] E. Ammicht, A. Potamianos, and E. Fosler-Lussier, "Ambiguity Representation and Resolution in Spoken Dialogue Systems," in *Proceedings of EuroSpeech*, vol. 3, pp. 2217–2220, September 2001.

[5] K. Macherey, F. J. Och, and H. Ney, "Natural Language Understanding Using Statistical Machine Translation," in *Proc. European Conference on Speech Communication and Technology, Aalborg, Denmark*, vol. 3, pp. 2205–2208, September 2001.

[6] F. Wessel, K. Macherey, and R. Schlüter, "Using Word Probabilities as Confidence Measures," in *Proceedings of ICASSP*, vol. 1, pp. 225–228, May 1998.

[7] S. Kanthak, A. Sixtus, S. Molau, R. Schlüter, and H. Ney, "Fast search for large vocabulary speech recognition," *Verbmobil: Foundations of Speech-to-Speech Translation*, pp. 63–78, November 2000.

[8] M. Weintraub, F. Beaufays, Z. Rivlin, Y. Konig, and A. Stolcke, "Neural-Network Based Measures of Confidence for Word Recognition," in *Proceedings of ICASSP*, vol. 2, pp. 887–890, April 1997.