SCORING CRITERIA FOR TREE BASED DIALOGUE COURSE MANAGEMENT

Klaus Macherey
Lehrstuhl für Informatik VI, Computer Science Department
RWTH Aachen - University of Technology
k.macherey@informatik.rwth-aachen.de

Hermann Ney
Lehrstuhl für Informatik VI, Computer Science Department
RWTH Aachen - University of Technology
ney@informatik.rwth-aachen.de

Abstract In this paper, we propose different scoring criteria 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. Experiments were performed on a German inhouse corpus that covers the domain of a German telephone directory assistance.

Keywords: Dialogue course management, trees, domain-independence

Introduction

Nowadays, there are numerous spoken dialogue systems for a variety of applications, e.g. inquiry systems for hotel reservations, restaurant guides, train timetable information systems, etc. [Aust et al., 1995, Seneff and Polifroni, 1996]. 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 or graphs as they are often used in applications for artificial intelligence, e.g. logical reasoning systems. The graphs are constructed from the knowledge of each domain. The nodes encode concepts, and the edges of a graph represent relations between the concepts, [Russell and Norvig, 1995]. In the context of spoken dialogue systems, trees have been firstly used by [Potamianos et al., 2000] and [Ammicht et al., 2001] for the purpose of domain independent dialogue management and ambiguity resolution.

In this paper, we also use tree based data structures in order to obtain domain independent representations. Here, the analysis of different scoring criteria 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 scoring criteria are able to determine the best path that leads as quickly as possible to a final state which presumably meets the user's request.

1. 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 dialogue actions. 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.

1.1 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.

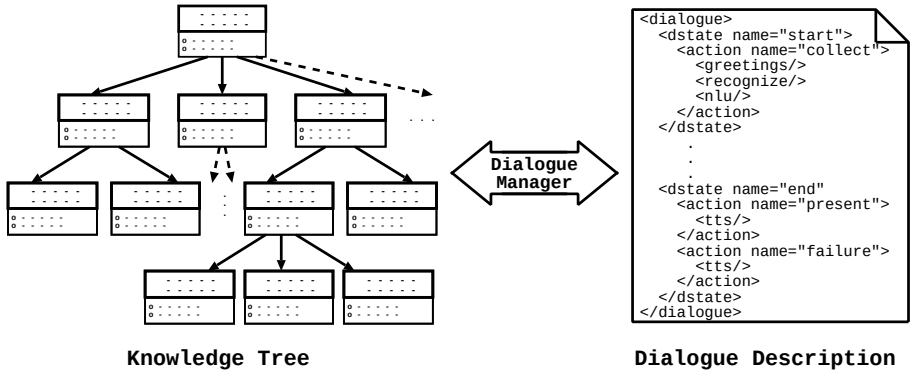

Figure 1. Basic structure of the dialogue system: The dialogue manager incorporates knowledge from the user input into the knowledge tree and determines the next dialogue action by analyzing the tree's information content.

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 (for presentation reasons, the attributes are not included in the figure). 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. For the given example, the specifications are requests for email addresses, fax numbers, and phone numbers, respectively. For a given utterance, the sequence of concepts is produced using methods derived from statistical machine translation, [Macherey et al., 2001]. 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, which are 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*.

1.2 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

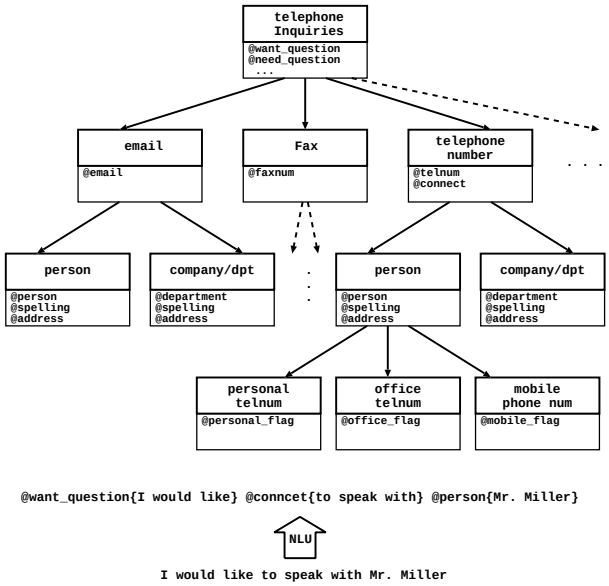

Figure 2. Tree based knowledge representation for a telephone directory assistance task. The sentence 'I would like to speak with Mr. Miller' is transformed into a concept representation via natural language understanding. After that, each concept/attribute pair is inserted into the corresponding tree nodes.

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.

2. Scoring criteria

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

2.1 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, [Wessel et al.,1998]. Word confidence measures can provide a score for the reliability of the concepts that have been produced by these words. 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:

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

For details, the reader is referred to [Wessel et al.,1998]. 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:

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

2.2 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:

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

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

2.3 Ambiguity degree

Ambiguities can result from several sources: misrecognized utterances, errors during the natural language understanding step, or ambiguous user language, [Ammicht et al., 2001]. 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:

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

2.4 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.

2.5 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:

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

2.6 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:

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

3. 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

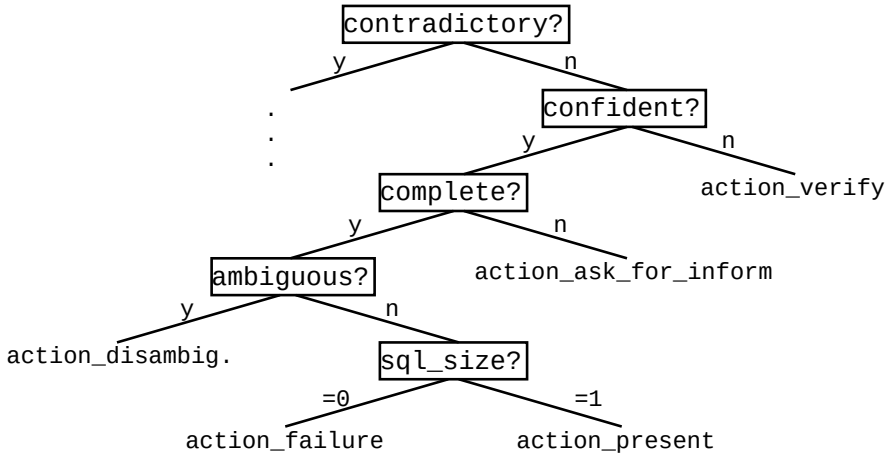

Figure 3. Excerpt of a decision tree for determining the subsequent dialogue action.

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. Although not expressed in the formulae, some of the cost functions' values depend on the chosen input modality. For example, when using text input via keyboard in contrast to speech input, the confidence is always set to 1.0. The verification function is set to 1, accordingly. 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.

4. 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

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

corpus	# session	# speakers	dur. [min]	# sentences	# words
train	131		101	1164	7310
dev	44	151	30	344	2039
eva	21		15	168	1013

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. 3) 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.

5. 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. 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 1305 words, including pronunciation variants.

We used word 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

Table 2. Recognition results using a class based trigram and confidence error rate for the evaluation corpus.

corpus	# WER [%]	baseline CER [%]	CER [%]
dev	21.5	–	–
eva	21.7	21.6	16.3

Table 3. Dialogue evaluation using speech as input modality.

# dialogues	# AER [%]	choice of best state [%]	successful sessions [%]
35	19.4	87.3	88.6

bigram case for realtime 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 evaluated the quality of the confidence measure using two criteria: the confidence error rate (CER) which is defined as the ratio between the number of false tags and the number of all recognized words, and a detection-error-trade-off curve (DET). Results for the CER are given in Table 2. A DET curve is depicted in Figure 4.

The natural language understanding component was trained using a method derived from statistical machine translation, [Macherey et al., 2001]. 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 number, the home number, position of a person in a company, etc. For evaluating the dialogue course manager, we analyzed 35 dialogue sessions. In 87% 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 put the emphasis on is using speech as input modality. Despite the high word error rate, the concept translation has proven to be very robust against recognition errors. Table 3 summarizes some results for the 35 speech based dialogue sessions. In case of poor recognition performance, the

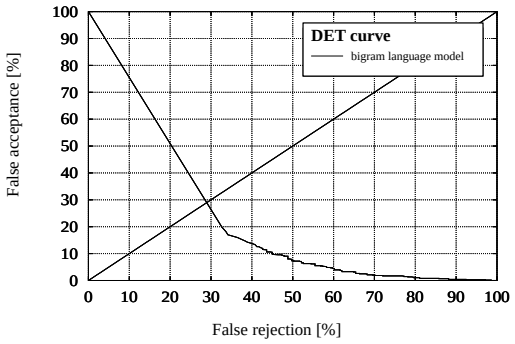

Figure 4. DET curve for the evaluation corpus.

confidence measure often cause the dialogue manager to explicitly verify erroneous data.

The following two examples describe typical dialogue sessions as they occurred during the recordings of the 35 dialogues that have been evaluated. Recognized words that have been rejected by the confidence measure are underlined. The output of the natural language understanding component is given as a sequence of concepts for every input sentence. The tree's state and the chosen dialogue action are described verbally. The first example is a dialogue where no recognition errors occurred. The second dialogue covers the case, where the confidence measure cause the dialogue manager to explicitly verify some information. For understanding purposes, the dialogue sessions are translated into English.

Example 1 for the telephone directory assistance task.

System: Hello. This is the RWTH telephone directory assistance. You can ask questions about telephone numbers, fax numbers, and email addresses of persons as well as companies. How may I help you?

User: Uhm, what shall I do now?

recog: [hesitation] what shall I do now

nlu: @FILLER{what shall I do now}

tree: all tree nodes empty → select action `ask_for_information`

System: Do you need a telephone number, a fax number, or an email address?

User: Can you give me the number of Mr Macherey?

rec: can you give me the number of [pause] Mr. Macherey

nlu: @can_question{can you give me} @telnum{the number of} @person{Mr. Macherey}

tree: found path with completely filled nodes, SQL-size $> 1 \rightarrow$ select action `disambiguation`

System: I have more than one entry of persons with name Macherey in my database. Please constrain your request.

User: Mr. Wolfgang Macherey

rec: Mr. Wolfgang Macherey

nlu: @person{Mr. Wolfgang Macherey}

tree: found path with completely filled nodes, SQL-size $= 1 \rightarrow$ select action `present`

System: The telephone number of Mr. Wolfgang Macherey is 80 21613. Shall I repeat the number?
...

Example 2 for the telephone directory assistance task.

System: Hello. This is the RWTH telephone directory assistance. You can ask questions about telephone numbers, fax numbers, and email addresses of persons as well as companies. How may I help you?

User: I need the number of Mrs. Hummel

recog: [noise] need the number of Mrs. Mail

nlu: @need_question{need the} @telnum{number of} @person{Mrs.} @UNKNOWN{Mail}

tree: last name required, confidence low $\rightarrow$ select action `verification`

System: Excuse me but I did not understand the last name. Please repeat.

User: Hummel

rec: Hummel

nlu: @person{Hummel}

tree: found path with completely filled nodes, SQL-size = 1 → select action `present`

System: The telephone number of Mrs. Hummel is ...

6. Summary and outlook

In this paper, we have proposed different scoring criteria for spoken dialogue course management. We have investigated whether the proposed criteria are able to select those dialogue states during a dialogue session which leads as quickly as possible to a final state that presumably meets the user's request. In 87% of all cases, the dialogue manager was able to choose the best successor state during a dialogue session. The principal cause for the high word error rate is the acoustic pruning threshold which has been set to a small value for realtime aspects. Although the word error rate is relatively high, the statistical natural language understanding component has proven to be very robust against recognition errors.

One of the advantages of this approach is that the features do not depend on the given domain. Of course one can incorporate additional features for the cost vectors, for example the length of a tree path or one can use a-priori probabilities for different paths. Also, there are a lot of possibilities to combine feature functions for selecting subsequent dialogue actions. For the future it would be interesting to use a maximum entropy framework in order to select the subsequent dialogue action.

References

[Ammicht et al., 2001] Ammicht, E., Potamianos, A., Fosler-Lussier, E. (2001). Ambiguity Representation and Resolution in Spoken Dialogue Systems *EuroSpeech 2001*, 3:2217–2220.

[Aust et al., 1995] Aust, H., Oerder, M., Seide, F., Steinbiss, V. (1995) The Philips automatic train timetable information system *Speech Communication 1995*, 17:249–262.

[Macherey et al., 2001] Macherey, K., Och, F.J., Ney, H. (2001). Natural Language Understanding Using Statistical Machine Translation *EuroSpeech 2001*, 3:2205–2208.

[Potamianos et al., 2000] Potamianos, A., Ammicht, E., Kuo, H.-K. J. (2000). Dialogue Management in the Bell Labs Communicator System *ICSLP 2000*, 2:603–606.

[Russell and Norvig, 1995] Russell, S., Norvig, P., (1995). Artificial Intelligence - A modern Approach *Prentice Hall International Editions, Upper Saddle River, NJ, 1995*, 10

[Seneff and Polifroni, 1996] Seneff, S., Polifroni, J. (1996) A New Restaurant Guide Conversational System: Issues In Rapid Prototyping For Specialized Domains *ICSLP 1996*, 2:665-668.

[Wessel et al.,1998] Wessel, F., Macherey, K., Schlüter, R. (1998). Using Word Probabilities as Confidence Measures *ICASSP 1998*, 1:225–228.