Multi-Level Error Handling for Tree Based Dialogue Course Management

Klaus Macherey, Oliver Bender, Hermann Ney

Lehrstuhl für Informatik VI, Computer Science Department
RWTH Aachen - University of Technology
D - 52056 Aachen, Germany

{k.macherey,bender,ney}@informatik.rwth-aachen.de

Abstract

For spoken dialogue systems, errors can occur on different levels of the system's architecture. One of the principal causes for errors during a dialogue session are erroneous recognition results which often lead to incorrect semantic interpretations. Even if the speech input signal has been correctly recognized, a natural language understanding component can produce error-prone sentence meanings due to the limitations of its underlying model. To cope with this problem, we introduce a multi-level error-detection mechanism based on several features in order to find erroneous recognitions, error-prone semantic interpretations as well as ambiguities, and contradictions. Here, the confidence output of one level directly serves as an additional input for the subsequent level. The proposed features and scoring criteria are passed to the dialogue manager which then determines the subsequent dialogue action.

1. Introduction

Spoken dialogue systems incorporate components like automatic speech recognition (ASR) units and natural language understanding (NLU) modules. Due to the limitations of the underlying models, the outputs that are produced by these components can be error-prone. During a dialogue session, the dialogue manager collects the information provided by these components and decides on the subsequent dialogue action. In case of system errors, this may result in an incorrect dialogue course. Since system errors cannot be avoided, the dialogue manager should at least be 'aware' of such errors if they occur. This implies a measure for estimating the output quality of each component. Confidence measures provide a general framework for this purpose. However, confidence measures are often used for the ASR level, only. Since the NLU module can produce error-prone outputs, even if the ASR has done well, additional confidence measures for the NLU level are desirable. Therefore, we define appropriate confidence measures for the NLU part.

For a spoken dialogue system, the input of the NLU module is usually the transcription of a speech input signal. In case of recognition errors, the question arises, whether the NLU component can benefit from the confidence scores of the ASR unit. Instead of just forwarding the ASR confidence scores to the dialogue manager, one could also use these confidence scores as an additional knowledge source for the NLU computation. The NLU approach that is used in our dialogue system, is defined within a *maximum entropy* (ME) framework [1]. Since ME approaches are based on various feature functions, we can easily integrate the confidence scores of the ASR unit as an additional feature for the NLU computation.

The remainder of this paper is organized as follows. In section 2 we give a short survey of related work. Section 3 briefly describes the framework of our dialogue system. Section 4 introduces the confidence measures that are used for the ASR and NLU unit. Section 5 describes an extension of the ME based NLU approach to additional confidence features. Section 6 describes the integration of the confidence measures in the dialogue system and explains the used confirmation strategies. Section 7 outlines the experiments that were carried out for this investigation and section 8 concludes with a discussion and an outlook on future works.

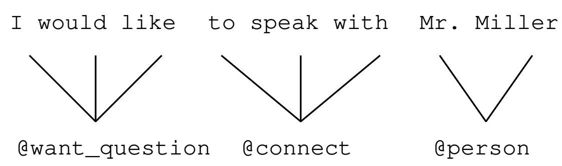

Figure 1: Example of a word concept mapping. The word sequence "I would like to speak with Mr. Miller" is mapped onto a sequence of flat concepts.

2. Related Work

In [2] confidence measures for two different levels of a spoken dialogue system are proposed. One confidence score is calculated for every content word and is based on n-best posterior probabilities, the other confidence measure is defined for semantic attributes of each content word and is based on inverse document frequencies. In [3] an approach similar to classification trees is used in order to detect error-prone recognitions. The approach uses several features like confidence measures from the ASR unit, features measuring dialogue efficiency, dialogue quality features which take the number of played rejection prompts and played timeout prompts into account, and textual features. In [4] user corrections of system errors are investigated which are often uttered with a different prosody compared to non-corrections. A detailed analysis of positive and negative cues that are employed by users in human machine interactions is given in [5]

3. Basic Dialogue Framework

In addition to an automatic speech recognizer, our dialogue system comprises a maximum entropy based natural language understanding component and a tree based dialogue course manager. Both are briefly described in the following.

3.1. Natural Language Understanding

The objective of the NLU component is to extract all the information contained in a natural language based input that is relevant for a specific task. For the NLU component, we use a concept based meaning representation as formal target language. An example is depicted in figure 1. During a dialogue session, the transcription of the speech input signal is passed to the NLU component which then determines the most likely sequence of concepts with respect to its underlying models. Here, we apply a NLU method that is defined within the maximum entropy (ME) framework [6]. We are given a sequence of input words $w_1^N = w_1, \ldots, w_N$. Then the probability of generating the sequence of concepts $c_1^I = c_1, \ldots, c_I$ can be computed as follows:

$$\hat{c}_1^I = \operatorname*{arg\,max}_{c_1^I} \{ Pr(c_1^I | w_1^N) \} \tag{1}$$

$$\begin{aligned} Pr(c_1^I|w_1^N) &= p_{\lambda_1^M}(c_1^I|w_1^N) \\ &= \frac{\exp\left[\sum_{m=1}^{M} \lambda_m h_m(c_1^I, w_1^N)\right]}{\sum_{c'^I_1} \exp\left[\sum_{m=1}^{M} \lambda_m h_m(c'^I_1, w_1^N)\right]} . \end{aligned} \tag{2}$$

For each feature function h_m, there is a model parameter λ_m. These model parameters are estimated using the *Generalized Iterative Scaling* (GIS) algorithm [7]. The feature functions cover lexical features, word features, transition features, and concept prior features. For their exact definition, see [1]. Applying log linear models to NLU has been firstly suggested by [8, 9]. In [1] it was shown that a ME based NLU approach outperforms a translation based approach on several tasks.

3.2. Tree based Dialogue Course Management

In order to receive a certain degree of domain independence for the dialogue manager, we use trees as the fundamental data structure. An example for the specific task of a telephone directory assistance is depicted in figure 2. The upper half of each tree node describes the part of the dialogue which is processed by the corresponding subtree. The lower part of each node consists of a list of concepts that are associated with that specific node.

During a dialogue session, instance trees are built from the original knowledge tree. Concept / Attribute pairs which have been retrieved from 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 that 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 more likely to be ambiguous and the user is required to constrain his request. 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 used in order to determine the subsequent dialogue action. For more details, the reader is referred to [10]. The knowledge of a domain is inherently contained within the structure of the tree. Since the actions of the dialogue manager are completely defined as tree operations, we receive a certain degree of domain independence. The cost function is computed for each tree path and can easily be extended to new features, especially to features for error handling that are introduced in the following section.

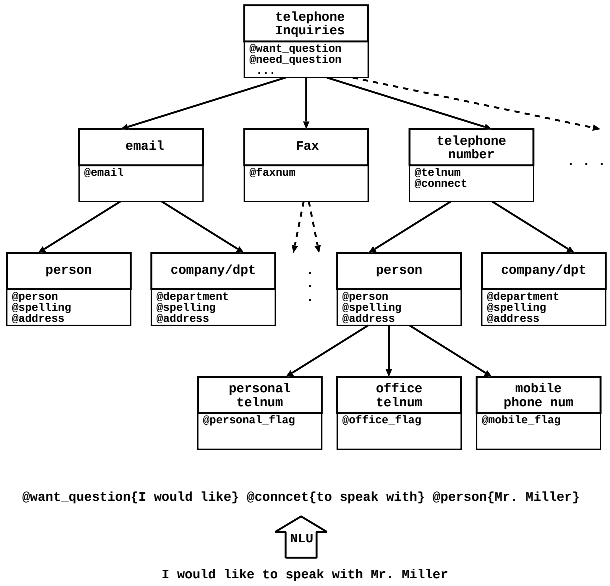

Figure 2: Tree based knowledge representation for a telephone directory assistance task. The sentence 'I would like to speak with Mr. Miller' is transformed into a concept representation via natural language understanding. After that, each concept/ attribute pair is inserted into the corresponding tree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

4. Multi-Level Confidence Measures

In speech recognition, confidence measures are employed for detecting recognition errors. If an erroneous recognition has been detected, the dialogue manager can initiate an appropriate confirmation strategy. In case of correctly recognized word sequences, confidence measures can help to avoid undesirable verification turns in automatic inquiry systems. However, the NLU component can produce error-prone results, too, even if the ASR unit has produced a perfect recognition. Therefore, we define confidence measures for both the ASR and the NLU unit.

4.1. Confidence Measures for Speech Recognition

Among several approaches to confidence measures in ASR, word posterior probabilities have been proven to be very effective in detecting misrecognized words [11]. 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 the complete sequence of acoustic feature vectors x_1^T is computed in the framework of a forward-backward algorithm on word graphs by summing over all incoming partial paths W_a starting at the graph's source and ending at time $t_a - 1$ and all partial paths W_e starting at time $t_e + 1$ and ending in the graph's sink. That is, we sum 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 summing over all word hypothesis probabilities $p(\cdot)$ which share the same word label and which have at least one common time frame t (for details, see [11, 12]):

$$\begin{aligned} &\widetilde{p}([w, t_a, t_e]|x_1^T) = \\ &\quad = \max_{t: t_a \le t \le t_e} \sum_{(t_i, t_j): t_i \le t \le t_j} p([w, t_i, t_j]|x_1^T) \end{aligned} \tag{3}$$

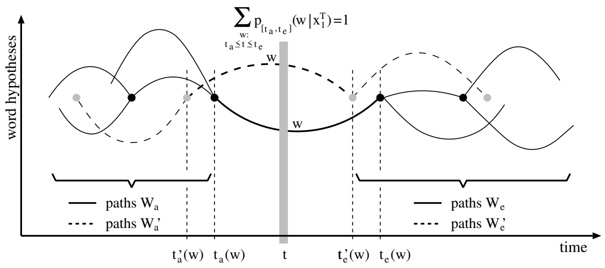

Figure 3: Basic principle of the word graph based confidence computation. The word posterior probabilities are computed within the framework of a forward backward algorithm with the constraint that for each time frame t, all word hypothesis probabilities that intersect this time frame must sum to unity.

The basic principle is depicted in figure 3. This confidence measure is probabilistic and exploits only information which is contained in the output word graph of the ASR unit. After computing the confidence, each recognized word is tagged as either correct or wrong, depending on whether its confidence exceeds a given threshold τ.

4.2. Confidence Measures for Language Understanding

Starting from speech recognition, confidence measures for language understanding can be defined in a similar way. The posterior probability of a concept c_m at position m, given the sequence of input words w_1^N can also be used as a confidence measure, thus providing a score for the reliability of each concept that has been produced by the NLU unit. Here, the computation of the concept posterior probabilities is done on N best lists. Since the generated concept sequences may have different lengths and the position of a concept may slightly differ within the sentence hypotheses of the N best list, we determine the Levenshtein alignment on the sentences according to the best target sentence [13]. That is, for each concept c_i in the best concept sequence $\hat{c}_1^I$, we determine the corresponding concept $\tilde{c}$ in any of the other sentences in the N best list. We denote the Levenshtein alignment of two sentences $\hat{c}_1^I$ and $\tilde{c}_1^{I_n}$ by $\mathcal{L} = \mathcal{L}(\hat{c}_1^I, \tilde{c}_1^{I_n})$ for $n = 2, \ldots, N$.

Using the Levenshtein alignment, we can easily compute word posterior probabilities for each concept $\hat{c}_i$ in the best concept sentence. We sum over the probabilities of all sentences containing the word in a position that is aligned to i in the Levenshtein alignment:

$$p_i(\hat{c}_i | w_1^N, \hat{c}_1^I, \mathcal{L}) = \\ = \frac{\sum_{n=1}^{N} p(\tilde{c}_1^{I_n} | w_1^N) \cdot \delta(\hat{c}_i, \mathcal{L}_i(\hat{c}_1^I, \tilde{c}_1^{I_n}))}{\sum_{n=1}^{N} p(\tilde{c}_1^{I_n} | w_1^N)}, \quad (4)$$

where $\delta(.,.)$ is the Kronecker function.

5. Confidence Dependent Semantic Analysis

Word confidence measures can provide a score for the reliability of the concepts that have been produced by these words. Instead of just forwarding the ASR confidence scores to the dialogue manager, the question arises, whether the NLU component can benefit from these scores by using them as an additional knowledge source.

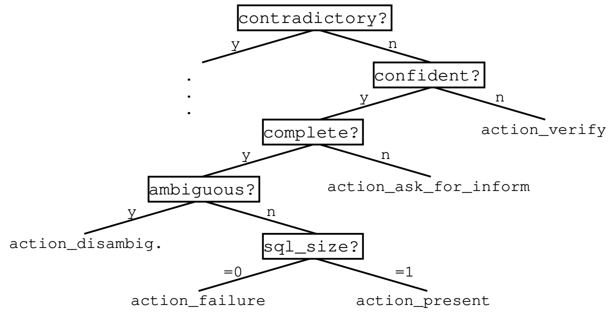

Figure 4: Excerpt of the decision tree that determines the subsequent dialogue action.

The approach to NLU as used in our dialogue system is defined within a ME framework. In this framework, we have a set of M feature functions h_m with model parameters λ_m. Among other things, the feature functions cover lexical features, word features and transition features. If transcriptions from the speech recognizer serve as input for the NLU component, one can also use their confidence scores as an additional feature for the ME computation. For this purpose, we simply employ the usual threshold based confidence tagging function, thus yielding the following binary feature:

$$h([w, t_a, t_e], x_1^T; \tau) \quad \longmapsto \quad \begin{cases} 1 & \text{if } \tilde{p}([w, t_a, t_e] | x_1^T) \geq \tau \\ 0 & \text{otherwise.} \end{cases}$$

6. Confirmation Strategy

For each user input, 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. 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. For further details concerning the computation of the cost functions, the reader is referred to [10]. If there are different possible paths that may continue the dialogue, the dialogue manager will choose the path with the best score in order to proceed the dialogue.

For error-handling, the interesting part is the 'confident?' node of the decision tree in figure 4. As an extension to [10], this node combines both the ASR and the NLU confidence handling. For the ASR confidence handling, we work with two different thresholds τ_1 and τ_2, separating the interval $[0, 1]$ into three disjoint sets $[0, \tau_1)$, $[\tau_1, \tau_2)$, and $[\tau_2, 1]$. If the confidence of the recognized utterance falls into the first interval, an explicit confirmation dialogue will be started. If the confidence is element of the second interval, the dialogue manager will continue the dialogue with an implicit confirmation question. In the third case, no confirmation strategy is used. For the NLU confidence handling, we always choose an explicit confirmation strategy if the NLU confidence score exceeds a given threshold.

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

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	16.5	14.8	9.4
eva	16.1	13.9	9.8

7. Experimental Results

Experiments were carried out for TELDIR, a German in-house telephone directory assistance task. 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 recording conditions cover clean speech, office noise, and traffic noise. The corpus statistics are summarized in table 1.

7.1. Recognition Results

The speech 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. No speaker-adaptive or normalization methods were applied. Due to the bandwidth of telephones, the signal analysis generates feature vectors with 25 dimensions, that is 12 cepstral coefficients with 12 first derivatives and the second derivative of the energy.

Table 2 shows recognition results for the development and the 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.

7.2. Language Understanding Results

The NLU part was trained using 1164 annotated sentence pairs from the training set. Each pair consist of the source input sentence and the target tag sequence. The used features are lexical features, capitalization features, list features, left contexts of words, prefix as well as suffix features, and concept prior features. For the exact definition of the features, see [1]. For training the NLU component, we used 1000 iterations of the GIS algorithm in combination with a heuristic speed-up method [14]. In order to improve the quality of the NLU approach, we used a set of word categories. Since it is unlikely that every proper name is observed during training, all proper names are mapped onto categories. An excerpt of the used categories is depicted in table 3.

Table 3: Excerpt of used word categories.

Category	Examples
$DEPARTMENT	• Rechnungsstelle
	• Sekretariat
$COMPANY	• BASF AG
	• Porsche
$FORENAME	• Klaus
	• Stefan
$SURNAME	• Schlegel
	• Wagner

The performance of the NLU part is evaluated using the *tagging error rate*[1] (TER). The TER is defined as the ratio of deleted, inserted, and substituted concept tags w.r.t. a Levenshtein alignment for a given reference concept tag string, and the total number of concept tags in all reference strings. Results are summarized in table 4. Note, that in contrast to [1], these error rates were produced on *recognized* transcriptions of the development and the test set. The relatively high error rate of around 20% mainly results from the fact that the training of the NLU component was only done on uncorrupt text data without recognition errors. A recognition on the training data yields a word error rate which is around 2%, so one cannot expect a major contribution by additionally using the recognized sentences from training data. Since the whole corpus is quite small, a further splitting of the training into two parts is beyond question. If we use the recognized transcriptions of the development set as additional training data, the performance of the NLU component will significantly improve (see second row in table 5).

Table 4: Tagging error rates (TER) and confidence error rates (CER) for the language understanding component of the dialogue system applied on the recognized transcriptions of the German telephone directory assistance corpus TELDIR.

corpus	TER [%]	baseline CER [%]	CER [%]
dev	19.8	17.8	16.1
eva	20.4	17.9	15.8

7.3. Confidence Results

After computing the confidence, each generated word is tagged as either *correct* or *wrong*, depending on whether its confidence exceeds the tagging threshold that has been optimized on the development set beforehand. The performance of the ASR confidence measure is evaluated using two different metrics:

- **C**onfidence **E**rror **R**ate
  The *confidence error rate* (CER) is defined as the number

[1] Basically, the TER corresponds to the concept error rate that we used in former publications. Because this error rate is also abbreviated by CER, we write TER here in order to distinguish it from the confidence error rate.

Table 5: Tagging error rates (TER) of the evaluation test set with and without using the ASR confidence scores as additional features.

used training set	TER [%]
train	20.4
+dev*	17.9
+confidence	17.0

of incorrectly assigned tags divided by the total number of generated words in the recognized sentence. The baseline CER is given by the number of substitutions and insertions, divided by the number of generated words. The CER strongly depends on the tagging threshold. Therefore, the tagging threshold is adjusted beforehand on a development corpus *distinct* from the test set.

- **D**etection **E**rror **T**radeoff curve
  The *detection error tradeoff* (DET) curve plots the *false rejection rate* versus the *false acceptance rate* for different values of the tagging threshold. The false rejection rate is defined as the number of correctly recognized words that have been tagged as wrong, divided by the total number of correctly recognized words. It is also referred to as *type I error*. The false acceptance rate (or *type II error*) is calculated as the number of incorrect words that have been accepted, divided by the total number of incorrectly recognized words. If the type I error is restricted by a given $\alpha > 0$, the type II errors usually cannot be restricted; there is a tradeoff between both error types.

Results for the ASR confidence measures are summarized in table 2 for both the development and the test set. A DET curve is depicted in figure 5. In contrast to the ASR confidence threshold, we have used sentence length dependent confidence thresholds for evaluating the performance of the NLU confidence measure. These thresholds were adjusted on the development set beforehand. Results are shown in table 4.

7.4. Confidence Dependent Semantic Analysis Results

For the confidence dependent semantic analysis, we used a subset of 260 sentences from the recognized transcriptions of the development corpus as additional training data. Using only a subset of the development set is motivated by the fact that for each pair the source sentence and the target tag sequence should have the same length, so we do not need to reannotate our data.

The first row in table 5 corresponds to the case of using only the annotated texts from the training corpus. Using additional data from the recognized transcriptions of the development corpus and retraining the log linear models significantly decreases the TER to 17.9%. This value was reached without using the confidence feature as introduced in section 5. If we use the confidence tag as an additional binary feature, the TER will be reduced to 17.0%

7.5. Dialogue System Results

We have recorded 35 dialogue sessions and evaluated them manually by comparing the system's decisions with human judged decisions. The results are listed in table 6. Additionally, the table includes the *attribute error rate* (AER) which is defined as the number of falsely tagged concept values divided by the total number of concept values which are provided by the NLU component.

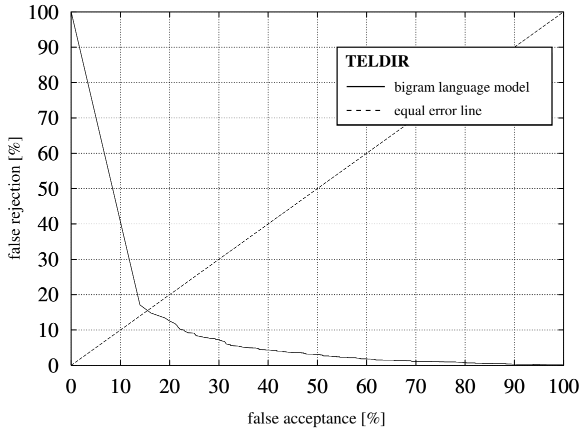

Figure 5: DET curve of the TELDIR test corpus. The point of intersection between the vertical line and the DET curve indicates the equal error rate.

Table 6: Results of 35 recorded dialogue sessions. The attribute error rate (AER) describes the percentage of falsely tagged concept values that are generated by the NLU component.

# dialogues	# AER [%]	successful sessions [%]
35	15.0	88.6

8. Summary

In this paper, we have proposed a multi-level error handling strategy. For this purpose, we have defined confidence measures based on posterior probabilities for both the ASR and the NLU level. The confidence scores of both components are passed as features to the dialogue manager which then determines the subsequent dialogue action. Additionally, the ASR confidence scores were directly passed to the ME-based NLU module which can use the ASR confidence scores as an extra knowledge source. For online evaluation, we have logged and analyzed 35 dialogue sessions.

The dialogue manager decides on the subsequent dialogue action by evaluating feature vectors by means of a decision tree. For future investigations, we plan to replace this decision tree based approach by a ME-based approach. Since the structure of the decision tree has a major influence on subsequent dialogue actions, a more probabilistic approach like ME can decouple the decisions from the rigid structure of the decision tree.

9. References

[1] O. Bender, K. Macherey, F.-J. Och, and H. Ney, "Comparison of Alignment Templates and Maximum Entropy Models for Natural Language Understanding," in *EACL*, (Budapest, Hungary), pp. 11–18, April 2003.

[2] K. Komatani and T. Kawahara, "Generating Effective Confirmation and Guidance Using Two-Level Confidence Measures for Dialogue Systems," in *Int. Conf. on Spoken Language Processing (ICSLP)*, vol. 2, (Beijing, China), pp. 648–651, October 2000.

[3] D. J. Litman, M. A. Walker, and M. S. Kearns, "Automatic Detection of Poor Speech Recognition at the Dialogue Level," in *Proceedings of ACL*, (University of Maryland), pp. 309–316, June 1999.

[4] M. Swerts, D. Litman, and J. Hirschberg, "Corrections in Spoken Dialogue Systems," in *Int. Conf. on Spoken Language Processing (ICSLP)*, vol. 2, (Beijing, China), pp. 615–618, October 2000.

[5] E. Krahmer, M. Swerts, M. Theune, and M. Weegels, "Error Detection in Spoken Human-Machine Interaction," *International Journal of Speech Technology*, vol. 4, no. 1, pp. 19–30, 2001.

[6] A. L. Berger, S. A. Della Pietra, and V. J. Della Pietra, "A maximum entropy approach to natural language processing," *Computational Linguistics*, vol. 22, pp. 39–72, March 1996.

[7] J. N. Darroch and D. Ratcliff, "Generalized iterative scaling for log-linear models," *Annals of Mathematical Statistics*, vol. 43, pp. 1470–1480, 1972.

[8] K. A. Papineni, S. Roukos, and R. T. Ward, "Feature-based language understanding," in *European Conference on Speech Communication and Technology (EUROSPEECH)*, (Rhodes, Greece), pp. 1435–1438, September 1997.

[9] K. A. Papineni, S. Roukos, and R. T. Ward, "Maximum likelihood and discriminative training of direct translation models," in *Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, (Seattle, Washington, USA), pp. 189–192, May 1998.

[10] K. Macherey and H. Ney, "Scoring Criteria for Tree based Dialogue Course Management," in *ISCA Tutorial and Research Workshop Multi-Modal Dialogue in Mobile Environments*, (Kloster Irsee, Germany), June 2002.

[11] F. Wessel, K. Macherey, and R. Schlüter, "Using Word Probabilities as Confidence Measures," in *Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 1, (Seattle, Washington, USA), pp. 225–228, May 1998.

[12] F. Wessel, R. Schlüter, K. Macherey, and H. Ney, "Confidence Measures for Large Vocabulary Continuous Speech Recognition," *IEEE Transactions on Speech and Audio Processing*, vol. 9, pp. 288–298, March 2001.

[13] V. I. Levenshtein, "Binary Codes capable of correcting deletions, insertions and reversals," *Soviet Phys. Dokl.*, vol. 10, pp. 707–710, 1966.

[14] D. Keysers, F. J. Och, and H. Ney, "Efficient Maximum Entropy Training for Statistical Object Recognition," in *Informatiktage 2002 der Gesellschaft für Informatik*, (Bad Schussenried, Germany), pp. 342–345, November 2002.