ESTIMATING SEMANTIC CONFIDENCE FOR SPOKEN DIALOGUE SYSTEMS

Sameer S. Pradhan and Wayne H. Ward

Center for Spoken Language Research
University of Colorado
Boulder, CO 80309-0594, USA
{spradhan,whw}@cslr.colorado.edu

ABSTRACT

In order for a spoken dialogue system to carry on a fluent conversation, it must be able to estimate confidence in its interpretation of the input that it is receiving. It must realize when it doesn't understand the user and interact to correct the problem. To this end, most systems have some form of confidence assessment mechanism. These algorithms generally estimate confidence on a word-by-word basis and sometimes use these estimates to accept or reject an utterance as a whole. This paper presents the confidence assessment mechanism developed for the CU Communicator spoken dialogue system. The focus of this mechanism is to assess confidence in the semantic representation extracted from an utterance by the system rather than in the string of words produced by the recognizer. We use a decision tree classifier with features based on acoustic models, language models, word lattice density, parser output and dialogue context. The classification performance is evaluated on a test set from the CU Communicator data. CU Communicator is a telephone based spoken dialogue system for getting information on air travel, hotels and rental cars.

1. INTRODUCTION

1.1. Motivation

In order for a spoken dialogue system to carry on a fluent conversation, it must be able to estimate confidence in its interpretation of the input that it is receiving. It must realize when it doesn't understand the user and interact to correct the problem. Speech recognition systems make errors, and systems can make understanding errors even when the speech is decoded correctly. In the absence of an accurate estimation of confidence, the system must adopt sub-optimal strategies for verifying information. Confirming each piece of information after each user utterance leads to a very tedious and inefficient interaction. Not confirming information at all, and waiting until much later in the dialogue for the user to notice a problem can be even worse. It is more difficult to recover when the problem is noticed later. Also, if the system is not confident about the state of a conversation, it tends to become more directed, decreasing the naturalness of the dialogue. A desirable solution is to have confirmation and rejection of input be guided by a confidence estimation procedure.

Most spoken dialogue systems have some form of confidence assessment mechanism. These algorithms generally estimate confidence on a word-by-word basis and sometimes use these estimates to accept or reject an utterance as a whole [1]. The latter tends to be a overkill, as a completely correct utterance is not a necessary precedent to a confident information extraction. We want to use the confidence estimate to guide our information verification process, not just for utterance level rejection. Our strategy is to assess confidence in the semantic representation extracted from an utterance by the system rather than in the string of words produced by the recognizer. It is this semantic representation that the system confirms in interactions with the user. It is this representation that drives the system actions. It is useful therefore to be able to estimate confidence in each element of the semantic representation. This measure can then be used directly in guiding confirmation interactions.

We train decision trees – using features based on acoustic models, language models, word lattice density, parser output and dialogue context – to assign confidence to the semantic representations.

This work was supported by DARPA through SPAWAR under grant #N66001-00-2-8906

1.2. The System

The CU Communicator [3] is an interactive, spoken dialogue system which complies with the Galaxy Hub architecture. It combines continuous speech recognition, robust semantic parsing and dialogue management, together with a live database backend, to assist telephone callers in scheduling airline, hotel and rental car reservations.

2. METHODOLOGY

2.1. Related Research

Until recently, most of the dialogue confidence measures were derived from the confidence given to individual words hypothesized by the decoder. Most previous research [2] tends to indicate that lattice density and language model features perform better than acoustic features, and their combination sometimes works better when complemented with parse-level features derived from the output generated by a semantic parser like Phoenix [4]. However, recently, researchers have started using utterance-level confidence measures to better assist the dialogue manager in taking a binary accept/reject decision [1]. Since accepting/rejecting an entire utterance is not an ideal strategy in most cases, we decided to explore one at a sub-utterance level – on extracted semantic representations spanning one or more words. This strategy requires changes in generation of features that were found to be most useful in previous experiments. Also, some other features – parallel to the word-level features, can now be generated from the semantic representations. We analyze the utility of these features by using them to classify the semantic extractions.

2.2. Extracted Representation

The parser produces a semantic representation of the extracted information. Each line of the extracted parse contains: a frame name, a slot name and a value. Our confidence annotation mechanism assigns a confidence score to each element. This confidence score represents the joint probability of the slot and its filler value.

An example will be helpful for clarification:

```
UTTERANCE: I WOULD LIKE TO GO FROM DENVER TO CHICAGO TOMORROW
           AFTERNOON
```

The extracted semantic representation is

```
Air:[Depart_Loc].DENVER
Air:[Arrive_Loc].CHICAGO
Air:[Date_Time].TOMORROW AFTERNOON
```

2.3. Data Preparation and Training

Our data set comprises 23,472 utterances collected using the CU Communicator system over a two-year period.

All the utterances (both the hypothesized and actual) are first parsed using Phoenix. Some of them do not produce a parse, so we simply discard them. Then, we annotate the sentences using just the frame and slot pair from each output line (generated from the transcribed utterances). We then partition the entire dataset by dialogue context (refer Section 3) and train a trigram language model for each context (henceforth referred to as *slot language model* conditioned on dialogue context). Our example utterance produces the following sequence:

```
Air:[Depart_Loc] Air:[Arrive_Loc] Air:[Date_Time]
```

The training examples comprise features extracted from the annotated sentence associated with the corresponding hypothesized sentences. Assuming that the recognizer generated a perfect hypothesis in our example, we would get the following annotated sentence:

```
Air:[Depart_Loc] (DENVER) Air:[Arrive_Loc] (CHICAGO)
Air:[Date_Time] (TOMORROW AFTERNOON)
```

Each frame-slot-words triplet, along with the associated statistics, contributes towards generating the feature values in an example used for training the decision tree. The examples produced using a sentence in a hypothesis are labeled correct or incorrect depending on whether or not the corresponding frame-slot-words triplet in the transcribed utterance is identical.

3. THE FEATURES

We consider the following features in our experiments

- ▷ **Dialogue Context** – In our system the dialogue context is modeled using the most recent system prompt. These can be clustered into distinct categories depending on the nature of the information they seek. We use 20 different contexts in the Communicator system.
- ▷ **Word LM Probability** – This is the average of the class-based trigram language model probability conditioned on the dialogue context.
- ▷ **Word LM Backoff** – This is the probability that a given n-gram backoff sequence over a given window around the word – in our case, two on either side – corresponds to a correctly hypothesized word. This is averaged over the number of words present in the semantic unit [4].

- ▷ **Slot LM Probability** – This is the slot language model probability of the slot under consideration – also conditioned on dialogue context.
- ▷ **Slot LM Backoff** – This is calculated just like its word counterpart, using slot language model conditioned on the dialogue context instead.
- ▷ **Word Confidence** – This measure is an estimate of the confidence of an hypothesized word directly as its posterior probability, computed on the word graphs using a forward-backward algorithm [5].

4. EXPERIMENTS AND RESULTS

After processing the 23,472 utterances as mentioned in Section 2.3, we are left with a total of 25,315 examples. We performed stratified sampling on the dialogue context to partition the data into training (60%) and test (40%) sets – so that we could perform two types of classification experiments – one, using the dialogue context as one of the features and training a single decision tree to classify all examples, and the other, by generating a separate tree for each dialogue context and testing the collective performance on the entire test set.

Feature Set (Dialogue Context +)	Equal Cost			10% False Accepts		
	%FA	%FR	%ACC	%FA	%FR	%ACC
Slot-Prob (1)	94.65	0.52	89.41	9.96	68.02	**38.19**
Word-Prob (2)	100.00	0.00	89.30	9.96	68.15	**38.07**
Slot-Backoff (3)	76.20	2.00	90.06	11.53	56.48	**48.33**
Word-Backoff (4)	85.06	1.14	89.88	9.50	52.71	**51.91**
Confidence (5)	74.72	1.97	90.25	9.78	63.49	**42.26**
1+2+3+4 (6)	76.38	1.76	90.26	9.96	49.02	**55.16**
1+2+3+4+5 (7)	67.44	1.91	91.08	10.06	43.50	**60.08**

Table 1. Performance of trees generated using dialogue context as one of the features.

Feature Set	Equal Cost			10% False Accept		
	%FA	%FR	%ACC	%FA	%FR	%ACC
Slot-Prob (1)	95.57	0.31	91.33	3.51	91.77	**20.17**
Word-Prob (2)	93.63	1.36	90.95	8.67	79.31	**25.66**
Slot-Backoff (3)	77.03	1.75	91.68	2.21	94.35	**20.42**
Word-Backoff (4)	80.44	1.94	90.57	5.07	84.02	**28.01**
Confidence (5)	75.74	1.92	91.77	4.70	92.87	**24.73**
1+2+3+4 (6)	68.36	3.43	90.22	8.58	76.04	**39.63**
1+2+3+4+5 (7)	67.80	2.15	90.52	10.79	51.57	**57.37**

Table 2. Performance of separate trees generated for each dialogue context.

We used the Weka machine learning toolkit's [6] implementation of C4.5 to train the decision trees. The semantic error, i.e., the total number of semantic units that were labeled incorrectly, in the entire dataset was about 10.54%. After partitioning, the training and test sets had semantic errors of 10.44% and 10.69% respectively.

Table 1 shows the classification accuracies (ACC) on the test set, using a single decision tree that has the dialogue context as one of its features – for various combination of feature sets. Alongside are the false acceptance (FA) and false rejection (FR) percentages for the same. FA is calculated as using the following formula (FR is calculated in a similar fashion).

$$FA = \frac{Number\ False\ Positives}{Total\ Negative\ Examples} \times 100$$

Accuracy is calculated as

$$ACC = \frac{Number\ Correctly\ Classified}{Total\ Number\ of\ Examples} \times 100$$

Table 2 shows parallel classification statistics when a different tree is used for each dialogue context.

The default decision tree objective function tries to maximize overall classification accuracy assuming equal cost for both the false acceptances and the false rejections, however, in this particular classification problem, false acceptances are more damaging than false rejections, since the assignment of a wrong value to a slot can lead to many user-initiated correction stages creating a greater possibility of misunderstanding, whereas, a false rejection means the system implicitly or explicitly confirms the value. Therefore, we present two sets of values – one with the default equal cost function, and other generated by varying the cost threshold such that the false acceptance percentage is close to a target value of 10%[1]. Though, this has a detrimental effect on the overall classification accuracy, it may be more optimal for system performance.

It can be seen that the tree created using dialogue context as a feature has slightly better performance than the cumulative performance of using a different tree for each dialogue context. Also, in the first case, in terms of accuracy as well as in terms of false rejection, all the features together are significantly better than any individual feature, bringing down the false rejection from a range of 50-70% to about 43%, which means less decision making about whether an implicit, explicit, or no confirmation is warranted.

We also considered using the hypothesized slot as one of the tree features, but it turns out to be a redundant or possibly detrimental addition at low false

[1]Since the false acceptance percentages cannot be finely controlled by adjusting the cost thresholds, the best we can do is to present the values that are nearest to 10%

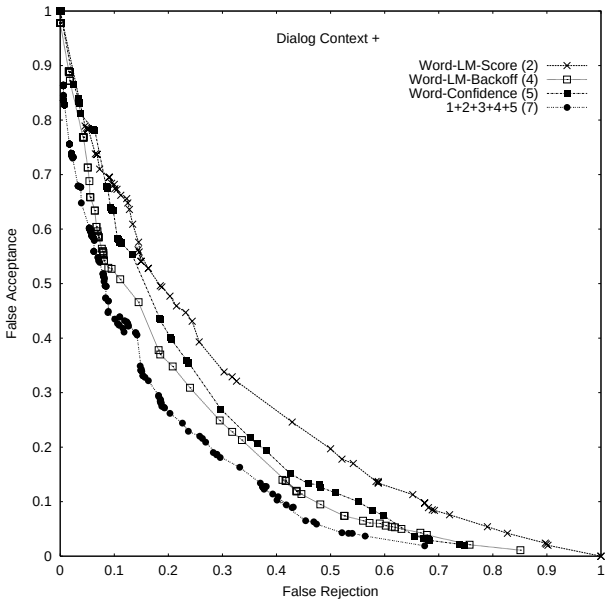

Fig. 1. ROC curves using different cost thresholds to train decision tree with dialogue context as a feature.

acceptance percentages. The ROC curves for trees with different feature combinations along with dialogue context as one of its features are shown in Figure 1. Since there was no significant difference between the ROC curves for slot and word language model score, we just show the one for word language model score. Also, the curve for slot language model backoff nearly coincided with the one for word confidence, so we show only the latter.

One of the particularly notorious problems that we have seen recurring in the CU Communicator is that of spurious city name misrecognitions, and so we decided to test another possible formulation of the classification task, which is to partition the data on the basis of the slot hypothesized, and train a decision tree on each set. To begin with, we extracted the examples that had city name as their hypothesis (3,423 examples) and trained a tree on it. It is interesting to note that the performance of this tree was not significantly different than the one trained on the entire data (15,182 examples). Also, the false rejection percentage at 10% false acceptance is about the same as that over all different hypotheses.

5. CONCLUSION

In this paper we discussed a mechanism for annotating confidence of the semantic representations that a dialogue manager uses while making verification decisions. We investigated the discrimination powers of various features: word lattice, word and slot trigram language model, parsing, dialogue context, and acoustic – at different cost thresholds, as indicated by the ROC curves. We then showed that a combination of all features is better than any one in isolation, based on the overall classification accuracy at a 10% false acceptance level. In our future work we will compare the performance of our current system with one that uses confidence annotation as a basis for verification.

6. REFERENCES

[1] Carpenter, P., Jin, C., Wilson, D., Zhang, R., Bohus, D., Rudnicky, A., "Is This Conversation on Track?", *Proceedings of Eurospeech 2001*, Aalborg, Denmark, Volume 3, September 2001, 2121–2124.

[2] Chase, L., "Error-Responsive Feedback Mechanisms for Speech Recognizers", *Ph.D. Thesis*, Carnegie Mellon University, Technical Report, CMU-RI-TR-97-18, April 1997.

[3] Pellom, B., Ward, W., Pradhan, S., "The CU Communicator: an Architecture for Dialogue Systems", *Proceedings of the International Conference Speech and Language Processing* (ICSLP-2000), Beijing, China, Volume II, October 2000, 723–726.

[4] Rubén, S., Pellom, B., Hacioglu, K., Ward, W., "Confidence Measures for Spoken Dialogue Systems", *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing* (ICASSP-2001), Salt Lake City, Volume I, May 2001, 393–396.

[5] Wessel, F., Schlüter, R., Macherey, K., Ney, H., "Confidence Measures for Large Vocabulary Continuous Speech Recognition", *IEEE Transactions on Speech and Audio Processing*, Volume 9, Number 2, March 2001, 288–298.

[6] Witten, I., Frank, E., *et al.*, "Weka 3 – Machine Learning Software in Java", `http://www.cs.waikato.ac.nz/~ml/weka/index.html`