Estimation of Semantic Confidences on Lattice Hierarchies

Robert Lieb, Tibor Fabian, Günther Ruske, Matthias Thomaes

Institute for Human-Machine Communication
Technische Universität München, Germany
{lie,fab,rus,tho}@mmk.ei.tum.de

Abstract

Inspired by the well-known method for confidence measure calculation via estimation of word posterior probabilities on the word graph, we devised a technique to estimate confidences on all levels of the hierarchically structured output of our one-stage decoder for interpretation of natural speech (ODINS). By constructing a nested lattice hierarchy, the generalized counterpart of the word graph, we estimate posterior probabilities for all nodes in the decoded semantic tree, namely for all contained semantic units and words. The obtained experimental results show that the tree node confidence measure performs significantly better than the confidence error base line, no matter if the evaluation is carried out on tree nodes representing semantic concepts, word classes, or words. Furthermore, the paper proposes possible applications of the tree node confidences to improve the grounding strategy of spoken dialog systems.

1. Introduction

Regarding a robust recognition of application-specific information, a spoken dialogue system can benefit a great deal from confidence measures delivered by the underlying speech recognition engine. On word level, there are efficient methods for computing confidence measures [1]. However, the speech interpreting component of the dialogue system usually derives a hierarchically structured semantic representation of the user’s utterance, that comprises more complex units than words, e.g. semantic concepts or word classes. Thus, in addition to word confidences, higher-level confidences related to these semantic units are needed by the dialogue system to safeguard the recognized structured content and to generate feedback in an adequate way.

Recent publications [2, 3] suggested to incorporate word confidences together with various other features extracted during the speech recognition and interpretation process into a classifier to assign confidences to each recognized semantic unit. The used classifiers (multi-layer perceptrons in [2] and decision trees in [3]) need explicit training before their application. A different approach is proposed by [4] which exclusively uses the primary knowledge sources of speech recognition and interpretation for confidence estimation. Here, the common method for word posterior probability calculation on the word graph [1] was extended to estimate concept posterior probabilities on a so-called concept graph, which is generated from an intermediate word graph by semantic parsing using stochastic context free grammars. However, the determined concept posterior probabilities have been applied to enhance word confidences and haven’t been evaluated as semantic confidences.

In this paper we present a general method to estimate confidences consistently for all semantic units and words that are part of the hierarchically structured output of our automatic speech interpretation system, called ODINS [5]. Applying a hierarchical language model consisting of arbitrarily deeply nested probabilistic transition networks together with standard speech recognition knowledge sources like a pronunciation lexicon and acoustic-phonetic models, ODINS determines the best fitting semantic tree directly from the speech signal in a single stage. In addition to the best solution the decoder optionally generates probable alternative semantic trees, compactly represented by a hierarchy of nested lattices. Following the basic idea of [4] to estimate confidences on a more complex graph than the word graph, we apply a generalized version of the underlying technique of [1] to estimate posterior probabilities for all sub-lattice instances in the generated lattice hierarchy. By intersection with the best fitting semantic tree, confidences for every tree node are computed from corresponding sub-lattice posterior probabilities. Thus, we clearly distinguish between the confidence for a semantic unit itself and confidences for its specific content, namely the confidences of its corresponding child nodes, carrying lower-level semantic units and/or words. To evaluate all computed confidences for a test set of recognized semantic trees we use the tree matching based evaluation scheme presented in [6] to retrieve the tree node mappings with the corresponding reference tree annotations. By adjusting a general threshold, every calculated confidence value can be evaluated whether it correctly detects the corresponding right or wrong tree node mapping, respectively.

The paper is organized as follows: Section 2 describes the lattice hierarchy representation which is constructed on a flat lattice created from the decoder’s backtracking information. Section 3 explains the estimation of posterior probabilities for sub-lattice instances and gives two possible definitions for the semantic tree node confidence. The examined confidence evaluation metrics are presented in Section 4. The experimental setup and the obtained results are discussed in Section 5. Finally, Section 6 summarizes the paper and points out possible applications of the presented work in spoken dialogue systems.

2. Lattice hierarchy representation

Hierarchical language modeling along with one-stage decoding permits an immediate retrieval of the semantic structure of probable recognition outputs from the backtracking records collected during token passing search [7]. The backtracking records chain together visited nodes in the search network hierarchy marking the beginnings and endings of encountered semantic units and words. By recording the n-best tokens that recombine at each search network node in every time frame, it is possible to disclose alternative probable search paths in form of a flat lattice containing entry and exit nodes of semantic units...
Briefly explained, the lattice hierarchy is generated from the flat lattice representation by the recursive construction of nested temporary lattices that contain all possible entry nodes for each exit node encountered during a backward depth-first search on the flat lattice. For every entry node of a finalized temporary lattice a corresponding sub-lattice instance is generated. Recursion stops with the construction of terminal word instances. By means of bookkeeping, already constructed word and sub-lattice instances are reused with the corresponding instance index, that was assigned beforehand.

3. Semantic tree node confidences

The lattice hierarchy constitutes a structural indexing of the flat lattice, because every included sub-lattice instance and word instance corresponds to a specific pair of entry and exit node inside the flat lattice. Posterior probabilities are estimated by applying the forward-backward algorithm on the flat lattice. Let \([I_L; t_i, t_j]\) designate a sub-lattice or word instance with label \(L\) and starting and ending times \(t_i\) and \(t_j\) corresponding to the entry and exit nodes \(i\) and \(j\) inside the flat lattice. On logarithmic scale the posterior probability for \([I_L; t_i, t_j]\) given the observed feature vectors \(x_T\) is estimated by the confidence \(C([I_L; t_i, t_j])\) which is calculated by forward and backward scores in the following way:

\[
-\log p \left( \frac{[I_L; t_i, t_j]}{x^T} \right) \approx \arg\max \left( \sum_{i} \alpha^{T_i} \beta^{B_i} \right)
\]

(1)

\(f_i\) denotes the forward score at the entry node, \(b_j\) the backward score at the exit node. The term \(f_j\) represents the forward score calculated between the entry and exit node and \(f_N\) the total forward score at the exit node of the flat lattice that is used for normalization, and thus has negative sign. The calculation of \(f_j\)' is done by the recursive procedure

\[
\forall q \in [1 \ldots N]: f_q' = \begin{cases} 0 & \text{if } q = i \\ \infty & \text{if } q \neq i \end{cases}
\]

(2)

that assumes flat lattice nodes sorted in topological order. \(P(q)\) denotes the set of predecessors of node \(q\). After recursion the result simply is \(f_j = f'_j\). Just like in [1], acoustic and language model scores on flat lattice edges are scaled by factors \(\alpha\) and \(\beta\) empirically optimized by cross validation experiments.

The best fitting semantic tree is equivalent with the best path through the decoded lattice hierarchy. Thus, semantic tree nodes correspond to sub-lattice instances visited by this best path, and the confidence for a semantic tree node \([T_L; t_i, t_j]\) can simply be defined equivalent to the confidence of the corresponding sub-lattice instance:

\[
C([T_L; t_i, t_j]) = C([I_L; t_i, t_j])
\]

(3)

Similar to [1] we investigated a more sophisticated definition of the semantic tree node confidence that takes into account all sub-lattice instances with the same label \(L\) intersecting the tree node’s time interval \(\{t_i \ldots t_j\}\):

\[
C_{acc}([T_L; t_i, t_j]) = \arg\max \left( \sum_{i} \alpha^{T_i} \beta^{B_i} \right) \forall \{t_i \ldots t_j\}: \{t_i \ldots t_j\} \neq \emptyset
\]

(4)
To approximate the logarithmic probability constraint \( C_{sec} \geq 0 \), we introduced the intersection ratio \( a_{sec} \), that scales posterior probabilities according to the degree of intersection of the time intervals of the semantic tree node \([t_i; t_j]\) and the corresponding sub-lattice instances \([l_i; l_k]\). Assuming intersecting time intervals, \( a_{sec} \) is calculated by

\[
a_{sec} (t_i, t_j, t_k, t_l) = \frac{\min(t_i, t_k) - \max(t_l, t_j)}{\max(t_i, t_j) - \min(t_k, t_l)} \tag{5}
\]

4. Evaluation metrics

For evaluation we use the tree matching scheme presented in [6]. A minimum tree edit distance algorithm determines the best tree match between a recognized semantic tree and its corresponding reference tree annotation by minimizing the costs caused by substituted, inserted and deleted tree nodes. For this best match the algorithm returns the specific mappings of correct, substituted, inserted and deleted tree nodes. The recognition performance is measured by the tree node accuracy

\[
Acc = \frac{N_C - N_I}{N_C + N_S + N_D} \tag{6}
\]

which takes into account the total number of tree node mappings that have been counted as correct (\(N_C\)), substituted (\(N_S\)), inserted (\(N_I\)), or deleted (\(N_D\)) over the whole set of tested utterances.

To quantify the performance of the semantic tree node confidences we define a threshold \( \varepsilon \) to decide whether a specific tree node mapping is classified as accepted or rejected. If it is accepted we count an error if the mapping indicates a substitution or an insertion. Respectively we count an error for a rejected correct mapping. In this way we get the total number of classification errors over all test utterances, namely the number of false accepted (\(N_{FA}\)) and false rejected (\(N_{FR}\)) tree node mappings. Confidence evaluation is only possible for mappings of correct, substituted and inserted tree nodes, because for a deleted reference tree node there exists no confidence value that could be classified.

A common confidence evaluation metric is the confidence error rate (\( CER \), see [1]) which is the ratio of classification errors and total number of evaluated mappings:

\[
CER = \frac{N_{FA} + N_{FR}}{N_C + N_S + N_I} \tag{7}
\]

It is compared with the decoder classification baseline which is obtained as the confidence error rate that results from the strategy that all mappings are tagged as accepted, without taking into account any confidence values at all. Thus the decoder baseline \( CER_{BL} \), only includes errors from false accepted substitutions and insertions:

\[
CER_{BL} = \frac{N_S + N_I}{N_C + N_S + N_I} \tag{8}
\]

If the confidence measure performs well, the confidence error rate drops below the decoder base line because the confidence classification identifies more substitutions and insertions than it produces errors on correct mappings by false rejection.

Another common confidence evaluation metric is the receiver-operator characteristic which is depicted by ROC-curves (also called DET-curves, see [1]). This diagram plots the false acceptance rate (\( FAR \)) and false rejection rate (\( FRR \)) at various settings for the classification threshold \( \varepsilon \). The false acceptance rate is the ratio of the number of false accepted mappings and the number of wrong (substituted and inserted) mappings.

### Table 1: Recognition performance and confidence error rate evaluation for confidence definitions \( C \) and \( C_{sec} \) on semantic concept (\( CO \)), word class (\( WC \)) and word (\( W \)) evaluation level, as well as over all tree nodes (\( TOT \)).

<table>
<thead>
<tr>
<th></th>
<th>( Acc )</th>
<th>( CER_{BL} )</th>
<th>( CER(C) )</th>
<th>( CER(C_{sec}) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( CO )</td>
<td>76.9</td>
<td>14.4</td>
<td>13.7</td>
<td>9.6</td>
</tr>
<tr>
<td>( WC )</td>
<td>94.0</td>
<td>5.3</td>
<td>2.7</td>
<td>1.9</td>
</tr>
<tr>
<td>( W )</td>
<td>83.2</td>
<td>13.4</td>
<td>11.6</td>
<td>10.7</td>
</tr>
<tr>
<td>( TOT )</td>
<td>82.8</td>
<td>12.6</td>
<td>11.0</td>
<td>9.2</td>
</tr>
</tbody>
</table>

Respectively the false rejection rate is the ratio of the number of false rejected mappings and correct mappings:

\[
FAR = \frac{N_{FA}}{N_S + N_I} \quad FRR = \frac{N_{FR}}{N_C} \tag{9}
\]

The ROC-curve shows the tradeoff between false acceptance and false rejection rate. For a well performing confidence measure the ROC-curve runs close to abscissa and ordinate of the diagram.

5. Experimental results

The results presented in this paper were produced with the same experimental setup that had been used in [6]: Both training and evaluation are based on a hierarchically annotated spontaneous speech corpus, that was collected in a wizard-of-oz simulation of a spoken dialogue system for an airport information system (the training subset covers 1446 utterances of 17 speakers, the cross validation subset 320 utterances of 3 speakers, and the test subset 233 utterances of 3 speakers). The used hierarchical language model consists of 47 semantic concepts, 11 word classes and 574 words. The acoustic modeling is performed by speaker-independent tied intra-word triphone HMMs with about 25k Gaussian mixture components, as described in [5].

Because we are particularly interested in semantic confidences, that is to say in confidences concerning semantic concepts, the evaluation is performed separately on different subsets of tree nodes that belong to the following hierarchy level categories: semantic concepts (\( CO \)), word classes (\( WC \)) and words (\( W \)). In addition we carried out an overall evaluation (\( TOT \)) that covers all semantic tree nodes independently of their hierarchy level category. The evaluation of the tree node accuracy, confidence error rate and ROC-curve for each evaluation level was performed on the test set containing new utterances of speakers who are not part of the training set. The scaling factors \( \alpha \) and \( \beta \) (see Eq. 2) were adjusted on the cross validation set. As expected, these experiments showed good performance for the setting \( \alpha = 1/s \) and \( \beta = 1 \), where \( s \) is the language model factor used during the decoding process to scale all weights of the hierarchical language model. The confidence classification threshold \( \varepsilon \) has been adjusted on the cross validation set as well. We found the minimum confidence error rates on all evaluation levels with only one specific setting of \( \varepsilon \), as expected. The obtained settings for \( \alpha \), \( \beta \), and \( \varepsilon \) have been left unchanged during the evaluation of the test set.

Table 1 shows the results of the final test set evaluation for the tree node accuracy, confidence error rate base line, and the confidence error rates for the semantic tree node confidence definitions \( C \) (Eq. 3) and \( C_{sec} \) (Eq. 4). On all evaluation levels there is a significant reduction of the confidence error rates as compared to the base line values, resulting in a total relative improvement of 27% for \( CER_{TOT}(C_{sec}) \). Furthermore the semantic tree node confidence definition \( C_{sec} \) performs significantly better than the simple definition \( C \). The setting of the
6. Conclusions and future work

Based on our speech interpretation framework ODINS, which combines hierarchical language modeling and one-stage decoding, we presented a method to estimate confidences for every node of a recognized semantic tree, which represents the application-specific semantic structure of a user utterance. No matter whether a tree node refers to a semantic concept or a simple word, the corresponding tree node confidence is estimated uniformly as a posterior probability on the implicit flat lattice representation of probable alternative semantic trees, with the aid of the explicit lattice hierarchy representation that provides the necessary structural information.

Because the presented offline evaluation shows promising results, we are planning to apply the tree node confidences inside the dialogue management module of the spoken dialogue system prototype, which has been developed in the NADIA research project. This system prototype realizes a cooperative, mixed-initiative spoken dialogue in the airport information domain and copes with spontaneous speech input. Tree node confidences provide the basis for grounding unsafe information in a differentiated way. By evaluating the confidence of a semantic concept and the confidences of its child nodes, the dialogue management has the ability to decide, whether to ask the user to clarify whole parts of his last utterance on a more abstract level, or to ask the user to confirm a specific data slot value. An example could be the decision, whether it’s better to ask the user, if he was talking about a time or about a flight code, or to prompt him to confirm the exact digits of a flight number. By exploiting the confidence information in addition with probable alternative paths in the lattice hierarchy, we expect to improve the dialogue grounding strategy by avoiding system queries which prompt the user to repeat the whole last utterance, as well as inappropriate system clarification queries, which confuse the user.

7. References


