Combining Several ASR Outputs in a Graph-Based SLU System

Marcos Calvo, Lluís-F. Hurtado, Fernando García, Emilio Sanchis

Departament de Sistemes Informàtics i Computació
Universitat Politècnica de València. Spain
{mcalvo,lhurtado,fgarcia,esanchis}@dsic.upv.es

Abstract. In this paper, we present an approach to Spoken Language Understanding (SLU) where we perform a combination of multiple hypotheses from several Automatic Speech Recognizers (ASRs) in order to reduce the impact of recognition errors in the SLU module. This combination is performed using a Grammatical Inference algorithm that provides a generalization of the input sentences by means of a weighted graph of words. We have also developed a specific SLU algorithm that is able to process these graphs of words according to a stochastic semantic modelling. The results show that the combinations of several hypotheses from the ASR module outperform the results obtained by taking just the 1-best transcription.

Keywords: Graph of words, graph of concepts, Spoken Language Understanding.

1 Introduction

Advances in speech technologies have allowed voice-driven human-computer interaction systems to be ubiquitous in our lives. All these systems have many features in common, and one of them is that they have to understand what the user said in order to provide a suitable answer. Spoken Language Understanding (SLU) aims to provide a semantic representation of the user's utterance.

The input to the SLU system is usually the 1-best transcription of the utterance provided by the ASR [6]. However, this approach makes it impossible to correct the mistakes made in the recognition stage. In recent years, there has been a growing interest in overcoming the limitations derived from using a single decoding of the utterance as the input to the SLU system by exploiting the information contained in the ASR lattices [7],[11]. Another way to address this problem is to combine a set of sentences provided by one or more ASRs, in order to reduce the effect of the errors introduced by any single sentence. One way to perform this combination is to use a voting algorithm [5], to obtain a new output that is made of segments corresponding to the original sentences. Another option, which is the one explored in this paper, is to build a graph of words from the set of sentences by using a Grammatical Inference method. This way, a set of extra sentences made up from chunks of the original sentences are represented in the graph of words along with the original sentences.

Many successful SLU systems are based on statistical models [3],[10],[8],[4]. This kind of modelization is able to represent the variability of the lexical realizations of concepts (meanings) as well as the different ways in which concepts can be arranged. Another important aspect of these models is that they can be learned from corpora. The training corpora must be large enough to allow an accurate estimation of the probabilities, and it must represent the lexical and syntactic variability that is used in the language to express the semantics as much as possible. Nevertheless, the training corpus may not be large enough to contain all the variability, and it is also important to have information about the errors that can be generated in the recognition process [11]. Since this information is ASR dependent, it is not usually included in the training process. For this reason, it can be a good approach to learn semantic models from a clean corpus and to enrich the input to the semantic decoding by means of multiple hypotheses. We have explored this approach and we have applied it to a task of an information system about railway timetables and fares in Spanish.

2 System description

Spoken Language Understanding is usually addressed as the task of finding the best sequence of concepts $\hat{C}$, given an utterance A:

$$\hat{C} = \underset{C}{\operatorname{argmax}}\, p(C|A) \tag{1}$$

By introducing the sequence of words W underlying the utterance Equation 1 can be written as:

$$\hat{C} = \underset{C}{\operatorname{argmax}} \max_{W} p(A|W) \cdot p(W|C) \cdot p(C) \tag{2}$$

In this work, we have used a decoupled modular architecture (see Figure 1). The key aspects of this architecture are the use of a Grammatical Inference algorithm to combine and generalize the outputs of one or more ASRs and a specific SLU algorithm that is able to take graphs of words as input.

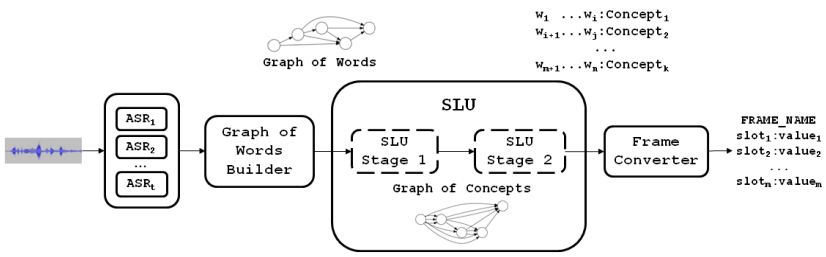

Fig. 1. Scheme of the architecture of our system.

In this architecture, the first module is dedicated to speech recognition. Since we want to combine and generalize multiple sentences provided by this module, its output will be either the n-best list provided by a single ASR or a set of 1-best decodifications provided by several ASRs working in parallel.

The second module combines the sentences provided by the first step by using a Grammatical Inference algorithm. The idea of Grammatical Inference is to generate a language (usually represented as an automaton or a graph of words) that generalizes a set of sentences that are provided as its input. Also, the algorithm that we have developed assigns a probability to each sentence of the new generalized set by means of a Maximum Likelihood criterion. This probability can be seen as a re-estimation of the distribution $p(A|W)$.

Next, the semantic decoding is carried out by means of a SLU module that is able to deal with graphs of words. For this system, we have developed a semantic decoding methodology that works in two stages. First, the graph of words is converted into a graph of concepts in which both syntactic and semantic information is included in the arcs of the graph. To build this graph of concepts, the first stage of the SLU algorithm uses both the graph of words and a set of Stochastic Finite State Automata (SFSA), which modelize the lexicalizations of the concepts of the task. Then the algorithm searches for matchings between the sequences of words that are represented in the graph of words and in each of the SFSA. The matchings of maximum probability become arcs in the graph of concepts. The weights of each arc in this graph are $p(A|W_i^j) \cdot p(W_i^j|c)$, where W_i^j stands for a chunk of a sentence represented between nodes i and j in the input graph of words and c is the concept it represents.

Then, this graph of concepts is processed in a second stage. In this stage, the algorithm searches for the best path in the graph based on the probabilities represented in both the graph of words and in a model that represents how the concepts are chained. The path of the maximum combined probability fulfills Equation 2. However, the output of this stage is not only the best sequence of concepts, it is also the underlying sequence of words and its segmentation in terms of the concepts.

Finally, the segmentation provided by the previous module is processed to extract and normalize the relevant semantic information and convert it into a frame representation.

3 A Grammatical Inference algorithm to build graphs of words

The goal of our Grammatical Inference algorithm is to generalize the syntactic structures of the sentences supplied by one or more ASRs by building a weighted graph of words. A graph of words represents a set of recognition alternatives that are built from the individual transcriptions of the utterance. This way, the SLU module can search among them for the most accurate sentence based on semantic constraints.

Correct utterance: *me puede decir horarios de trenes a Alicante*
(could you tell me train timetables to Alicante)

MSA Matrix with multiple ASR outputs:

me	puede	decir	horarios	de	trenes	–	Alicante
–	puede	decir	horas	de	trenes	–	Alicante
me	puede	decir	hola	–	trenes	a	Alicante

Graph of words:

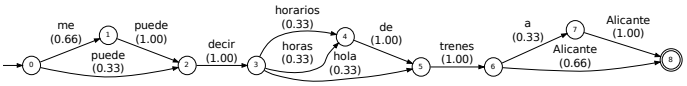

Fig. 2. Method to build a graph of words from multiple ASR outputs.

Our algorithm for building this weighted graph of words works in two steps. First, the different recognition alternatives are aligned using a Multiple Sequence Alignment (MSA) algorithm [1]. To carry out this process, we have modified the ClustalW [9] Multiple Sequence Alignment software.

The MSA process builds an alignment matrix. Each row in this matrix represents awords different sentence, and each column represents the alignment of each symbol. When a symbol cannot be aligned to any other symbol of any other sentence, the special symbol '-' is used (non-alignment points).

The second step consists of building a weighted directed acyclic graph of words from the information contained in the MSA alignment matrix. The graph construction algorithm starts creating as many nodes as columns in the alignment matrix, plus one for the initial state. Then, for each cell in the matrix that contains a symbol different to '-', we create an arc in the graph of words. Each arc is labeled with the word attached to the cell and has a counter of the number of times it is used. Finally, we weight the arcs by normalizing the counters attached to them. The final node of the graph is the node that represents the last column of the matrix.

Figure 2 shows an example of how the graph-builder algorithm works. As shown, this graph represents not only the input sentences, but also a set of sentences of similar characteristics. For example, the correct sentence *me puede decir horarios de trenes a Alicante* (could you tell me train timetables to Alicante) was not among the transcriptions provided, but it can be recovered using this mechanism. Furthermore, any full path from the initial to the final node in the graph represents an alternative transcription of the original utterance, and its probability is the product of the probabilities of the individual arcs of the path. Hence, this graph provides a re-estimation of the probability distribution $p(A|W)$, considering only a generalization of the individual transcriptions provided by the ASR module and weighting them according to a Maximum Likelihood criterion.

4 Semantic decoding

Our SLU method works in two stages, both of which use stastistical semantic models. The first stage converts a graph of words into a graph of concepts using the information provided by the semantic model about the lexical structures that are associated to each concept. The graph of concepts has the same nodes as the graph of words. However, each arc represents a path in the graph of words whose underlying sequence of words is associated to a concept. Hence, each arc is labeled with the corresponding sequence of words and the concept it is attached to. Each arc is also weighted using a combination of the probabilities represented in the graph of words and those provided by the semantic model. The second stage finds the best sequence of concepts by searching for the best path in the graph of concepts, based also on the information about the concatenation of concepts included in the semantic model. The method for building the graph of concepts finds paths between any pair of nodes in the graph of words that represent sequences of words that are associated to any of the concepts of the task. To modelize the probability of a sequence of words for a given concept, we train a bigram Language Model (LM) for every concept. Thus, given a sequence of words W_i^j induced by a path from node i to node j in the graph of words, the LM associated to concept c computes the probability $p(W_i^j|c)$.

An n-gram LM can be represented as a Stochastic Finite State Automaton (SFSA). Hence, the problem of searching for relevant sequences of words in the graph of words for each concept can be stated as the search for common paths in both the graph of words and the automaton that represents the LM for each concept. However, due to the nature of this problem, we can add two restrictions to this statement. Let LM_c be the LM attached to the concept c and let q_c be a state of this automaton. The first restriction is that any path in LM_c must start at its initial state, but it can end at any state q_c. The second restriction is related to the second stage of the semantic decoding process. We search for the best path in the graph of concepts and the score for any path is the product of the probabilities of its edges combined with the score provided by a LM of sequences of concepts. Hence, in the first stage, for any pair of nodes i, j and any concept c, only the path in the graph of words that maximizes the score $p(A|W_i^j) \cdot p(W_i^j|c)$ becomes an arc in the graph of concepts. Therefore, the graph of concepts can be built by using the following Dynamic Programming algorithm[1]:

$$M(i,j,q_c) = \begin{cases} 1 & \text{if } i = j \wedge q_c \text{ is the initial state of } LM_c \\ 0 & \text{if } i = j \wedge q_c \text{ is not the initial state of } LM_c \\ 0 & \text{if } j < i \\ \max\limits_{\substack{\forall a \in E_{GW}:\mathrm{dest}(a)=j \\ \forall (q'_c,\mathrm{wd}(a),q_c) \in LM_c}} M(i,\mathrm{src}(a),q'_c) \cdot p(q'_c,\mathrm{wd}(a),q_c) \cdot \mathrm{wt}(a) & \text{otherwise} \end{cases} \tag{3}$$

[1] We say that for every two nodes i, j in the graph of words, it holds that $i < j$ if i comes before j in the topological order of the nodes of the graph.

where dest(a) stands for the destination node of the arc a in the graph of words, src(a) refers to its source node, and wd(a) and wt(a) refer to the word and the weight attached to the arc, respectively. Also, $(q'_c, \text{wd}(a), q_c)$ represents a transition from the state q'_c to the state q_c labeled with wd(a) in the SFSA that represents LM_c. In consequence, $M(i, j, q_c)$ represents the best path in the graph of words that starts in the node i, ends in the node j, and whose underlying sequence of words reaches the state q_c in LM_c.

The second SLU stage searches for the best path in the graph of concepts, taking into account a bigram LM of sequences of concepts, which modelizes the probability distribution of the sequences of concepts $p(C)$. This search is performed via Dynamic Programming. The result is the best sequence of concepts as well as the underlying sequence of words and its segmentation in terms of the concepts.

Finally, this segmentation is converted into a frame representation (Table 1), which involves deleting irrelevant segments, reordering concepts and attributes, and automatically instantiating certain task-dependent values, among others.

Table 1. Example of semantic segmentation and its frame.

Input utterance	*hola quería saber los horarios para ir a Madrid* *(hello I'd like to know the timetables to go to Madrid)*
Semantic segments	*hola* : courtesy *quería saber* : query *los horarios para ir* : <time> *a Madrid* : destination_city
Frame	(TIME?) DEST_CITY : Madrid

5 Experimental results

To evaluate the proposed approach, we have performed a set of experiments using the DIHANA task [2]. This task consists of a telephone-based information system for trains in Spanish. It has a corpus of 900 dialogs of spontaneous telephonic speech (which were acquired using the Wizard of Oz) that amount to 6,229 user turns from 225 speakers. This set of user turns was split into a subset of 4,889 utterances for training and 1,340 for testing. The orthographic transcriptions of all the user turns are available and are semi-automatically segmented and labeled using a set of 30 concepts.

We used the HTK, Loquendo, and Google ASRs. Table 2 shows if the Acoustic Model and the Language Model of each ASR were trained with the information from the training corpus. It also shows the resulting Word Error Rates (WERs). As expected, the greater the amount of information provided to the ASR from the corpus, the lower the WER.

In order to validate our approach, we performed three types of SLU experiments. The first type constitutes the baseline and consists of taking the 1-best of each ASR separately. In the second type, we took the n-best from the Google

Table 2. Information of the task provided to each ASR.

ASR	Acoustic Model	Lang. Model	WER
HTK	yes	yes	17.55
Loquendo	no	yes	20.12
Google	no	no	29.73

ASR since it is the one that best modelizes a real-world situation, and we built graphs of words using them. Finally, we took the 1-best from each ASR and combined them into a graph of words. To evaluate each experiment we measured the WER, the Concept Error Rate (CER), and the Frame-Slot Error Rate (FSER), which refers to errors in the semantic frames.

The results obtained in our experiments are shown in Table 3. These results show that, in terms of FSER, the combination of multiple hypotheses from the ASR module outperformed the respective baselines. The same happened for CER, except when comparing the CER achieved using the combination of the sentences from all the ASRs with the one obtained using the 1-best sentence from HTK. The reason for this is related to the data we used for training each ASR. The LMs for HTK and Loquendo were trained with data from the task, while the Google ASR had no information from the task. This way, it was easier for HTK and Loquendo to recognize in-vocabulary words, but when an out-of-vocabulary word appeared they failed, while the Google ASR could provide the correct transcription. Thus, when we combined the three ASRs, the Google ASR helped to identify some semantic segments with important keywords (which may have been out-of-vocabulary words), but in some cases it generated more variability in the graph due to its transcription errors. The results also show that in most cases the FSER is lower than the CER, which means that most of the errors were done in semantically irrelevant segments, such as courtesies. In terms of WER, the quality of the transcription achieved using a combination of several hypotheses and the proposed semantic decoding method was better than the respective baselines in all cases. This helped to improve the FSER, as the values of the frame slots were better recognized. Thus, we confirm our hypothesis that the sentences obtained through a generalization process by means of a Grammatical Inference algorithm lead to an improvement in the overall behavior of the system.

Table 3. Results obtained using the different compositions of ASR outputs as well as the individual 1-bests.

Input graphs of words	WER	CER	FSER
HTK 1-best	17.55	14.15	12.81
Loquendo 1-best	20.12	24.10	22.65
Google 1-best	29.73	32.50	32.69
Google 3-best	27.04	24.28	23.77
Google 5-best	26.85	23.85	23.00
HTK + Google + Loquendo 1-bests	14.87	15.58	10.48

6 Conclusions

In this work, we have presented an approach to SLU based on the combination of several ASR outputs in a graph-based system. We have developed a Grammatical Inference algorithm that takes several recognitions provided by the ASR module and builds a graph of words that represents a generalization of the original sentences. We have also developed a two-stage SLU method, which is based on Dynamic Programming algorithms. We have evaluated this approach using the Spanish DIHANA task. The results show that an appropiate combination and generalization of the transcriptions provided by the ASR module improves the overall behavior of the system.

Acknowledgements This work is partially supported by the Spanish MEC under contract TIN2014-54288-C4-3-R and FPU Grant AP2010-4193

References

1. Bangalore, S., Bordel, G., Riccardi, G.: Computing Consensus Translation from Multiple Machine Translation Systems. In: ASRU. pp. 351–354 (2001)
2. Benedí, J.M., Lleida, E., Varona, A., Castro, M.J., Galiano, I., Justo, R., López de Letona, I.n., Miguel, A.: Design and acquisition of a telephone spontaneous speech dialogue corpus in Spanish: DIHANA. In: LREC. pp. 1636–1639 (2006)
3. Bonneau-Maynard, H., Lefèvre, F.: Investigating stochastic speech understanding. In: IEEE Automatic Speech Recognition and Understanding Workshop (ASRU). pp. 260–263 (2001)
4. Calvo, M., García, F., Hurtado, L.F., Jiménez, S., Sanchis, E.: Exploiting multiple hypotheses for multilingual spoken language understanding. CoNLL pp. 193–201 (2013)
5. Fiscus, J.G.: A post-processing system to yield reduced word error rates: Recognizer output voting error reduction (ROVER). In: IEEE Workshop on Automatic Speech Recognition and Understanding, 1997. pp. 347–354 (1997)
6. Hahn, S., Dinarelli, M., Raymond, C., Lefèvre, F., Lehnen, P., De Mori, R., Moschitti, A., Ney, H., Riccardi, G.: Comparing stochastic approaches to spoken language understanding in multiple languages. IEEE Transactions on Audio, Speech, and Language Processing 6(99), 1569–1583 (2010)
7. Hakkani-Tür, D., Béchet, F., Riccardi, G., Tür, G.: Beyond ASR 1-best: Using word confusion networks in spoken language understanding. Computer Speech & Language 20(4), 495–514 (2006)
8. He, Y., Young, S.: Spoken language understanding using the hidden vector state model. Speech Communication 48, 262–275 (2006)
9. Larkin, M.A., Blackshields, G., Brown, N.P., Chenna, R., McGettigan, P.A., McWilliam, H., Valentin, F., Wallace, I.M., Wilm, A., Lopez, R., Thompson, J.D., Gibson, T.J., Higgins, D.G.: ClustalW and ClustalX version 2.0. Bioinformatics 23(21), 2947–2948 (Nov 2007)
10. Segarra, E., Sanchis, E., Galiano, M., García, F., Hurtado, L.: Extracting Semantic Information Through Automatic Learning Techniques. IJPRAI 16(3), 301–307 (2002)
11. Tür, G., Deoras, A., Hakkani-Tür, D.: Semantic Parsing Using Word Confusion Networks With Conditional Random Fields. In: INTERSPEECH (2013)