Joint Decoding for Speech Recognition and Semantic Tagging

Anoop Deoras, Ruhi Sarikaya, Gokhan Tur, Dilek Hakkani-Tür

Microsoft Corporation, 1065 La Avenida, Mt. View, CA 94043 USA

{Anoop.Deoras, Ruhi.Sarikaya}@microsoft.com, {Gokhan.Tur, Dilek}@ieee.org

Abstract

Most conversational understanding (CU) systems today employ a cascade approach, where the best hypothesis from automatic speech recognizer (ASR) is fed into spoken language understanding (SLU) module, whose best hypothesis is then fed into other systems such as interpreter or dialog manager. In such approaches, errors from one statistical module irreversibly propagates into another module causing a serious degradation in the overall performance of the conversational understanding system. Thus it is desirable to jointly optimize all the statistical modules together. As a first step towards this, in this paper, we propose a joint decoding framework in which we predict the optimal word as well as slot (semantic tag) sequence jointly given the input acoustic stream. On Microsoft's CU system, we show 1.3% absolute reduction in word error rate (WER) and 1.2% absolute improvement in F measure for slot prediction when compared to a very strong cascade baseline comprising of the state-of-the-art recognizer followed by a slot sequence tagger.
Index Terms: ME, CRF, SLU, CU, ASR

1. Introduction

In today's state-of-the-art conversational understanding systems, it is a norm to employ a cascade approach where many statistical and rule based modules are connected to one another following a simple pipeline. Traditionally such systems connect ASR with SLU so that one best output from ASR is fed into SLU sub modules for slot sequences tagging and domain & intent classification. In this paper we focus on the slot sequence prediction task for SLU and propose a joint decoding approach where both recognition and semantic tagging performance is improved. Improving ASR further benefits tasks which are independent of slot sequence tagging.

More formally, given acoustic signal, $\mathbf{A}$, ASR outputs most likely word sequence, $\mathbf{W}^*$ given by

$$\mathbf{W}^* = \underset{\mathbf{W}\in\mathcal{W}}{\operatorname{argmax}} P(\mathbf{A}|\mathbf{W}) \times P(\mathbf{W}) \tag{1}$$

Typically in cascade systems, this ASR 1-best hypothesis is then fed into SLU system (say targeting slot sequence prediction) to output most likely sequence of slots, $\mathbf{C}^*$, given by:

$$\mathbf{C}^* = \underset{\mathbf{C}\in\mathcal{C}}{\operatorname{argmax}} P(\mathbf{C}|\mathbf{W}^*) \tag{2}$$

Unfortunately, ASR is usually far from perfect and typically $\mathbf{W}^*$ suffers from many word errors leading to errors in the prediction of slot sequence. Previously, researchers have addressed this issue by incorporating more information from ASR search spaces viz. confusion networks, N-best lists and word lattices. For instance, recently, Kurata et.al. [1] proposed an approach for named entity recognition on spoken utterances. In this work, they clustered confusion bins of confusion networks and trained a model maximizing the posterior probability of a named entity conditioned on some finite number of cluster IDs, representative of the words in some finite context surrounding the named entity. Such an approach approximates the probability of slot sequence directly given acoustics.[1] Previously, researchers have also tried to model $P(\mathbf{C}|\mathbf{A})$ explicitly. For instance, Yaman et.al. [2], modeled this distribution as the joint distribution of $\mathbf{C}$ and $\mathbf{W}$ given $\mathbf{A}$, summing over all possible word sequences: $P(\mathbf{C}|\mathbf{A}) = \sum_{\mathbf{W}\in\mathcal{W}} P(\mathbf{C}, \mathbf{W}|\mathbf{A})$. However, summing over full length word sequences is not a computationally easy task and hence they approximated the sum with a $\max$ and also approximated the space of word sequences $\mathcal{W}$ with the one obtained in the form of N-best lists after decoding $\mathbf{A}$ with an ASR. Similarly, instead of doing sequence prediction, they focused only on utterance classification task.

In the above and related other approaches such as [3], it is the contention to improve slot sequence prediction performance by modeling only slot sequences directly given acoustics without worrying about improving ASR performance. Although improving ASR performance may not be that essential from a minimalistic view of just named entity recognition or slot sequence prediction task, however, conversational understanding encompasses many other tasks which are independent of slot sequence or named entity recognition. Tasks such as intent determination or interpretation of spoken utterance etc. rely on features extracted from not only transcribed word sequences but also hypothesized slot sequences. Thus it becomes important to improve performance of both ASR and SLU modules jointly. In this paper, we take a first step towards this and propose a joint decoding framework in which given acoustics, we model the joint distribution of $\mathbf{W}$ and $\mathbf{C}$, i.e., we seek pair of sequences comprising of words and slots such that jointly they maximize the posterior probability given acoustics. Formally:

$$\{\mathbf{C}, \mathbf{W}\}^* = \underset{\mathbf{C}\in\mathcal{C}, \mathbf{W}\in\mathcal{W}}{\operatorname{argmax}} P(\mathbf{C}, \mathbf{W}|\mathbf{A}) \tag{3}$$

The rest of the paper is organized as follows. In the next section – Sec. 2, we describe mathematical formulation for joint decoding. In Sec. 3, we describe in brief how we process lattices and do search on them for joint recognition and tagging. In Sec. 4, we describe our experimental setup and present results and finally conclude in Sec. 5.

2. Mathematical Formulation

As discussed above, we aim to find pair of sequences composed of words and slots such that the posterior probability of this pair

[1] $\arg\max_{\mathbf{C}\in\mathcal{C}} P(\mathbf{C}|\mathbf{A}) \approx \arg\max_{\mathbf{C}\in\mathcal{C}} P(\mathbf{C}|\phi(\mathbf{A}))$, where $\phi(\mathbf{A})$ can capture ASR confusions easily obtained from confusion networks, N-best lists or even word lattices.

is maximized given acoustics:

$$\begin{aligned}\{\mathbf{C},\mathbf{W}\}^* &= \underset{\mathbf{C}\in\mathcal{C},\mathbf{W}\in\mathcal{W}}{\operatorname{argmax}} P(\mathbf{C},\mathbf{W}|\mathbf{A}) \\ &= \underset{\mathbf{C}\in\mathcal{C},\mathbf{W}\in\mathcal{W}}{\operatorname{argmax}} P(\mathbf{C}|\mathbf{W},\mathbf{A})P(\mathbf{W}|\mathbf{A}) \\ &= \underset{\mathbf{C}\in\mathcal{C},\mathbf{W}\in\mathcal{W}}{\operatorname{argmax}} P(\mathbf{C}|\mathbf{W})P(\mathbf{A}|\mathbf{W})P(\mathbf{W}) \quad (4)\end{aligned}$$

where the first term of the last step follows from the assumption that given word sequences, slot sequences and acoustics are conditionally independent. Second term of the last step follows from the Bayes' rule and ignoring $P(\mathbf{A})$, since $\mathbf{A}$ is observed and hence fixed.[2] We restrict the space of word sequence hypotheses to those possible in the word lattice obtained after decoding acoustic observation with ASR engine.

Typically, $P(\mathbf{C}|\mathbf{W})$ is modeled directly either using Maximum Entropy (ME) [4] framework[3] or Conditional Random Field (CRF) [5] framework or more traditionally as a generative model by invoking Bayes' rule and putting a prior on the slot sequence i.e. $P(\mathbf{C})$. Formulation presented in (4) above has previously been proposed by Pieraccini et.al. [6] where they modeled the relation between $\mathbf{C}$ and $\mathbf{W}$ using a generative model rather than a discriminative one. However, due to the complexity of the task, they did not report any results. Servan et.al. [7] too formulated the joint decoding problem similarly. They demonstrated reductions in concept error rate sometimes at the expense of increased word error rate. We, in this paper, decompose the joint formulation differently and use a state-of-the-art discriminative model for $P(\mathbf{C}|\mathbf{W})$ as it has been shown that they outperform generative models such as the ones using Hidden Markov Models. The novelty of our work lies in the fact that use of powerful discriminative models on word lattices allow us to capture a variety of features, something which is extremely difficult in a finite state automaton representation of joint language model of concepts and words.

CRF models the posterior distribution of a slot sequence given a word sequence as shown below:

$$P(\mathbf{C}|\mathbf{W}) = \frac{1}{Z(\mathbf{W})}\prod_{t=1}^{T}\psi(c_t,\phi(c_1^{t-1}),\gamma_t(\mathbf{W})) \quad (5)$$

where, $\phi(c_1^{t-1})$ is an equivalence classification of slot sub sequence up to $(t-1)^{th}$ position. In first order linear chain CRFs, this function returns c_{t-1}. Similarly, $\gamma_t(\mathbf{W})$ is a position specific equivalence classification of the entire word sequence. Typically, this function returns w_{t-n+1}^{t+n-1} i.e., n-grams around the t^{th} word position.[4] $\psi(\cdot)$ is an exponentiated weighted sum of active features.

ME models, on the other hand, model local distribution of a particular slot type given some context of slots and words. Thus, the posterior distribution of slot sequence given word sequence is first broken down using chain rule and then each component is modeled using a ME model:

$$\begin{aligned}P(\mathbf{C}|\mathbf{W}) &= \prod_{t=1}^{T} P(c_t|c_1^{t-1},\mathbf{W}) \\ &\approx \prod_{t=1}^{T} P(c_t|\phi(c_1^{t-1}),\gamma_t(\mathbf{W}))\end{aligned}$$

where, similar to CRF, $\phi(c_1^{t-1})$ returns c_{t-1} and $\gamma_t(\mathbf{W})$ returns w_{t-n+1}^{t+n-1}. Each individual distribution is then modeled as shown below:

$$P(c_t|\phi(c_1^{t-1}),\gamma_t(\mathbf{W})) = \frac{\psi(c_t,\phi(c_1^{t-1}),\gamma_t(\mathbf{W}))}{Z(\phi(c_1^{t-1}),\gamma_t(\mathbf{W}))} \quad (6)$$

where $\psi(\cdot)$ is an exponentiated weighted sum of active features and $Z(\cdot)$ is a normalization constant.

From (4), (5) and (6), it is clear that the search for optimal pair of slot and word sequences on word lattice can become intractable under the CRF model as this model requires to explicitly compute $Z(\mathbf{W})$ when there are multiple $\mathbf{W}$ to be compared. On the other hand, due to local nature of ME models, we can make use of dynamic programming on word lattices, something much more tractable. Computation of Z for ME models is not very expensive because exponentiated weighted feature sum of active features has to be evaluated only for a handful of predicted slot types (in our case, 50). We thus propose to use ME models in conjunction with ASR acoustic model and language models to solve (4). We invoke chain rule to obtain:

$$\begin{aligned}&P(\mathbf{C}|\mathbf{W})P(\mathbf{A}|\mathbf{W})P(\mathbf{W}) \\ &= \prod_{t=1}^{T} P(c_t|c_1^{t-1},\mathbf{W})P(\mathbf{a}_t|w_t)P(w_t|w_1^{t-1}) \\ &\approx \prod_{t=1}^{T} P(c_t|c_{t-1},\gamma_t(\mathbf{W}))P(\mathbf{a}_t|w_t)P(w_t|w_{t-n+1}^{t-1})\end{aligned}$$

where we are assuming that for given $\mathbf{A}$ and $\mathbf{W}$, we can obtain segmentation of the acoustics from the speech lattice and once we have such a segmentation, the posterior probability of any particular acoustic sub string, $\mathbf{a}_t$, belonging to some t^{th} segment is independent of acoustic sub strings up-to $t-1$ segments and all words of $\mathbf{W}$ but w_t when conditioned on w_t.[5] We also work with n-gram language model (LM) and restrict the context at any t^{th} position to contain previous $n-1$ words only.

3. Lattice Expansion and Viterbi Decoding

In order to assign LM probability for a word on an arc and probability of some slot type given the context of previous words and previous slot type, it is essential that the lattice maintains at-least as much unique context, at every arc, as is required for the prediction. Weng et.al. [8] describe an algorithm to achieve node splitting for trigram LM rescoring on word lattices produced using a bigram LM. We propose a node splitting algorithm, which is similar in spirit to Weng et.al.'s algorithm, but differs in the fact that it is generalized to n-gram context i.e. beyond 2 words as well as in the key aspect that for any node under consideration, information from only the preceding arcs

[2] In practice, we use a scaling parameter on tagging model, to balance the dynamic range of all component models.

[3] ME models are traditionally referred to as local classifiers and not sequence classifiers, however, we can use series of ME classifiers coupled with a Viterbi search algorithm without any beam pruning, for doing an optimal sequence tagging.

[4] This assumes the model uses only lexical features.

[5] In our work, if $\gamma_t(\mathbf{W})$ returns w_{t-n+1}^{t+n-1}, we will refer the model to have left and right (**LR**) context. If it returns w_{t-n+1}^{t}, we will refer the model to have just left (**L**) context.

is used to either split or merge the nodes, thus avoiding the need to process all outgoing arcs from the current node. Details are as follows:

The algorithm iteratively splits and merges the node of the input lattice to resolve the n-gram ambiguities. Algorithm 1 illustrates the steps for maintaining unambiguous left context of $n-1$ words at every arc. We will introduce some notation. Let us represent the input lattice by G_L and output lattice by G_Q. The lattices are directed acyclic graphs and hence we will represent the set of vertices and edges in them by (V_L, E_L) and (V_Q, E_Q) respectively. For any node $r \in V_L$, we will denote the set of expanded nodes ($\in V_Q$) by $\mathcal{S}_r$ and initialize $\mathcal{S}_0$ to have state 0 implying that the start state of input lattice expands to just one start state in the output lattice.

The algorithm then proceeds exploring each node of G_L in a topological order. For every such node r, it iterates over all incoming arcs and for every such incoming arc, e, it finds out its start state s and then iterates over all of s's expanded states as recorded in G_Q. For each such an expanded state, p_0, in G_Q, it finds out the sub string of $n-2$ words, $\mathbf{v}$, on previous $n-2$ arcs eventually ending at p_0 (since G_Q maintains the unambiguous context for n-gram processing at each arc, the words $\mathbf{v}$ will be the same no matter which $n-2$ arcs end at p_0). It then iterates over all arcs between s and r in the input lattice G_L and for each arc, finds the word associated with it. For each such word, w and its associated cost c, it finds out if the word sequence $\mathbf{v}, w$ exists on any $n-1$ consecutive arcs in G_Q.[6] If it exists, then end state for these consecutive arcs is found out ($\in G_Q$) and for this state, m', a new arc $\{p_0, m'\}$ with word label w and cost c is added to G_Q. If it does not exist, then G_Q is updated with new node m, new edge $\{p_0, m\}$ with word label w and cost c. We also map node r ($\in G_L$) to the set of its expanded nodes $\mathcal{S}_r$ ($\in G_Q$), now also containing the newly created node m. Value of m is then incremented by 1. The resulting lattice is an equivalent lattice[7] and maintains a unique left context of $n-1$ words thus allowing for unambiguous n-gram processing. Fig. 1 shows a toy word graph and its expanded representations, in order to maintain unambiguous left bi-gram context. Unlike exhaustively splitting all the nodes (Fig. 1(b)), the above node splitting and merging technique avoids state space blowing up issue (Fig. 1(c)) by merging states 3 & 5 and states 4 & 6.

Our speech lattices contain pauses and silence symbols, which do not contribute towards tagging and hence we remove them by first mapping them to epsilon symbols and then removing them using `fstrmepsilon` functionality offered by OpenFST toolkit [9].[8] Although we remove these special tokens, it is made sure that the overall score of the complete path in the word lattice is not altered.

In our work, we not only need left unambiguous context, but also right. In order to do that, we first represent our word lattice as a weighted finite state machine and then take the following steps:

1. We use Alg. 1 to expand the lattice to have left unambiguous context.
2. We then reverse the FSA representation of lattice. We make use of `fstreverse` functionality of OpenFST.
3. We again use Alg. 1 to expand the reversed lattice (this takes care of maintaining right context in the original lattice).
4. We reverse the lattice back. This results in lattice with both left and right unambiguous context at every arc.

[6] this is done using a Hash function, which is reset for every r, s pair to make sure that G_Q does not add any new paths with respect to the original lattice.

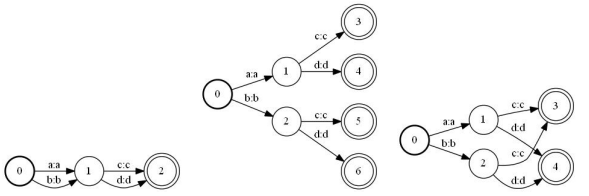
[7] by equivalent we mean that language accepted by G_L is same as that accepted by G_Q and vice versa.

[8] Word lattices are directed acyclic graphs and can be represented as weighted finite state automaton (W-FSA).

Algorithm 1 Node Splitting Algorithm

Require: Lattice: $G_L = (V_L, E_L)$
  return Lattice: $G_Q = (V_Q, E_Q)$
  $m = 1$
  Hash $\mathcal{H}$, mapping sequence of words to a state
  $V_Q = \{0\}$, $E_Q = \phi$
  $\mathcal{S}_0 = 0$
  for $r \in V_L$ in topological order **do**
    for $s \in V_L : \{s, r\} \in E_L$ **do**
      for $p_0 \in V_Q : p_0 \in \mathcal{S}_s$ **do**
        $\mathbf{v} = \text{word}(\{p_{n-2}, p_{n-3}\}) \cdot \ldots \cdot \text{word}(\{p_2, p_1\})$
          $\cdot\text{word}(\{p_1, p_0\})$
          $\forall i \in \{0, \ldots, n-3\} : \{p_{i+1}, p_i\} \in E_Q$
        for $w \in \text{word}(\{s, r\})$ **do**
          $c = \text{cost}_w(\{s, r\})$
          {// Merge and Split part}
          if exists $\mathcal{H}(\{\mathbf{v}, w\})$ **then**
            $m' = \mathcal{H}(\{\mathbf{v}, w\})$
            $\text{word}(\{p_0, m'\}) = w$; $\text{cost}(\{p_0, m\}) = c$
            $E_Q = \{E_Q, \{p_0, m'\}\}$
          else
            $\mathcal{H}(\{\mathbf{v}, w\}) = m$
            $\mathcal{S}_r = \{\mathcal{S}_r, m\}$; $V_Q = \{V_Q, m\}$
            $E_Q = \{E_Q, \{p_0, m\}\}$
            $\text{word}(\{p_0, m\})$=w; $\text{cost}(\{p_0, m\})$=c; m++
          end if
        end for
      end for
      Reset $\mathcal{H}$
    end for
  end for

Once we obtain the expanded lattice, we traverse the lattice in a topological order and at every arc, we extract the unambiguous word context (left and right) and for every possible slot type on the previous arc, we obtain a distribution over all slot types on the current arc using ME model. We then use Viterbi decoding to find out an optimal path comprising of words and tags in the lattice such that the joint probability is maximized given acoustics.[9]



(a) Word graph (b) Exhaustive node splitting (c) Compact node splitting

Figure 1: Toy word graph and its expanded representations

[9] In our implementation of Viterbi decoding, at any word arc in the lattice and for every slot type on that arc, we remember the best incoming word-slot pair transition by evaluating all possible slots on all previous word arcs.

4. Experiments and Results

We have focused on slot sequence tagging for spoken language understanding on Microsoft's CU system. The acoustic and language model for our ASR were trained from thousands of hours of annotated speech from a large variety of domains, including voice search. Our n-gram language model (LM) is a class grammar where named entity classes (movie names, actor names etc.) are expanded in turn to a full set of exhaustive entries. Our conversational understanding task pertains to movies domain, where a user issues a natural language query to retrieve movies and/or information there of. For instance, a user could say "`show me movies with brad pitt`" or "`who is the director of titanic`" etc. Our slot sequence tagger was trained with 2 kinds of discriminative models – Maximum Entropy (ME) and Conditional Random Fields (CRF). We made use of Wapiti tool [10] to train these models. We trained these models on a set of about 12K utterances, comprising of queries such as the ones mentioned above. We set aside another 4K utterances, which we split into two to form two data sets, each comprising about 2K utterances. We used one of them as a held-out / development data set to find out optimal scaling parameter for the tagging model. Our ME and CRF models were trained with lexical features. We used the current word and words from the window of 2 words around the current word to extract local features. For ME model training, we used both left (3 words) as well as left and right features (5 words). In all our experiments with CRFs, we used left and right context so as to have the best baseline possible. The total number of slot types is 50.

Table 1 shows performance of various methods under various configurations. ME-L and ME-LR correspond to ME model using left alone and left & right context respectively. Similarly, CRF-LR corresponds to CRF model using both left and right context. We compare performance of various models under 4 configurations: (a) Manual Transcription (Manual), (b) Speech word lattice oracle word sequence, the one which minimizes word error rate (Lat OB), (c) Speech word lattice maximum a posteriori 1 best word sequence (Lat 1B) and (d) Our proposed joint decoding framework, which maximizes the joint probability of words and tags given acoustic signal (Jnt.Dec.). Setups (a), (b) and (c) correspond to cascade system. The slot sequence tagging performance on the lattice oracle hypotheses possibly represent an upper bound on the tagging accuracy. Setup (d) corresponds to the proposed joint decoding approach.

From the results, we can see that if we use cascade model i.e. use the top hypothesis from ASR and then do slot sequence tagging on it using discriminative models, then we end up getting an F measure of 72.2%, 74.8% and 78.0% using ME-L, ME-LR and CRF-LR models respectively. As against that, our proposed method obtains an F measure of 75.8% and 79.2% using ME-L and ME-LR models respectively. F measure of 79.2% using ME-LR joint decoding setup is a 4.4% and 1.2% absolute improvement over ME-LR and CRF-LR cascade baselines. Use of CRF models for joint decoding is left as part of our future work.

Another very interesting result is in terms of improvement in transcription accuracy. While the word error rate (WER) in cascade setup (irrespective of slot sequence model used) is same as that obtained from ASR lattice 1-best decoding, it is **1.3**% absolute lower in joint decoding setup. Thus our proposed method not only improves tagging accuracy of slots, but also reduces the WER of the hypothesis.

(a) Development data set

Setup	ME-L		ME-LR		CRF-LR	
	WER	F	WER	F	WER	F
(a) Manual	0	87.7	0	90.3	0	91.2
(b) Lat OB	10.4	80.8	10.4	83.3	10.4	84.3
(c) Lat 1B	18.7	74.0	18.7	76.8	18.7	77.8
(d) Jnt.Dec.	17.5	77.0	**17.8**	**79.7**	-	-

(b) Evaluation data set

Setup	ME-L		ME-LR		CRF-LR	
	WER	F	WER	F	WER	F
(a) Manual	0	85.4	0	88.2	0	90.6
(b) Lat OB	10.3	78.6	10.3	81.4	10.3	83.9
(c) Lat 1B	19.7	72.2	19.7	74.8	19.7	78.0
(d) Jnt.Dec.	18.7	75.8	**18.4**	**79.2**	-	-

Table 1: *Performance (WER (%) and F measure) of cascade and proposed joint decoding technique. Use of CRF models for joint decoding is left as part of our future work.*

5. Conclusions

In this paper, we presented a novel joint decoding framework for joint recognition and tagging of hypotheses. We demonstrated significant improvements in both recognition and semantic tagging accuracy, over cascade approach. As part of future directions, we plan to use CRF model, with possibly additional features such as pre- and suffix-features etc. [11], for joint decoding and use the improved word sequence hypotheses for other SLU tasks such as intent determination, domain classification etc.

6. References

[1] G. Kurata, N. Itoh, M. Nishimura, A. Sethy, and B. Ramabhadran, "Named entity recognition from Conversational Telephone Speech leveraging Word Confusion Networks for training and recognition," in *Proc. of IEEE ICASSP*, 2011.

[2] S. Yaman, L. Deng, D. Yu, Y.-Y. Wang, and A. Acero, "An Integrative and Discriminative Technique for Spoken Utterance Classification," *IEEE TASL*, vol. 16, no. 6, pp. 1207 –1214, Aug. 2008.

[3] D. Hakkani-Tur, F. Bechet, G. Riccardi, and G. Tur, "Beyond ASR 1-Best: Using Word Confusion Networks in Spoken Language Understanding," *Computer Speech and Language*, 2006.

[4] A. Ratnaparkhi, "Maximum Entropy Models for Natural Language Ambiguity Resolution," Ph.D. dissertation, University of Pennsylvania, 1998.

[5] J. Lafferty, A. McCallum, and F. Pereira, "Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data," in *Proc.of the International Conference on Machine Learning*, 2001.

[6] R. Pieraccini and E. Levin, "Stochastic Representation of Semantic Structure for Speech Understanding," *Speech Communication*, 1992.

[7] C. Servan, C. Raymond, F. Bechet, and P. Nocera, "Conceptual decoding from word lattices: application to the spoken dialogue corpus MEDIA," in *Proc. of INTERSPEECH*, 2006.

[8] F. Weng, A. Stolcke, and A. Sankar, "Efficient Lattice Representation and Generation," in *Proc. of the ICSLP*, 1998.

[9] C. Allauzen, M. Riley, J. Schalkwyk, W. Skut, and M. Mohri, "OpenFst: A General and Efficient Weighted Finite-State Transducer Library," in *Proc. of the International Conference on Implementation and Application of Automata*, 2007.

[10] T. Lavergne, O. Cappé, and F. Yvon, "Practical very large scale CRFs," in *Proc. of ACL*, 2010.

[11] S. Hahn, et.al., "Comparing Stochastic Approaches to Spoken Language Understanding in Multiple Languages," *IEEE TASLP*, 2011.