Joint Discriminative Decoding of Words and Semantic Tags for Spoken Language Understanding

Anoop Deoras, *Member, IEEE,* Gokhan Tur, *Senior Member, IEEE* Ruhi Sarikaya, *Senior Member, IEEE* Dilek Hakkani-Tür, *Senior Member, IEEE*

Abstract

Most Spoken Language Understanding (SLU) systems today employ a cascade approach, where the best hypothesis from Automatic Speech Recognizer (ASR) is fed into understanding modules such as slot sequence classifiers and intent detectors. The output of these modules is then further fed into downstream components such as interpreter and/or knowledge broker. These statistical models are usually trained individually to optimize the error rate of their respective output. In such approaches, errors from one module irreversibly propagates into other modules causing a serious degradation in the overall performance of the SLU system. Thus it is desirable to jointly optimize all the statistical models 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 sequence (semantic tag sequence) jointly given the input acoustic stream. Furthermore, the improved recognition output is then used for an utterance classification task, specifically, we focus on intent detection task. On a SLU task, we show 1.5% absolute reduction (7.6% relative 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 state-of-the-art large vocabulary ASR followed by conditional random field (CRF) based slot sequence tagger. Similarly, for intent detection, we show 1.2% absolute reduction (12% relative reduction) in classification error rate.

Index Terms

A. Deoras, G. Tur, R. Sarikaya and D. Hakkani-Tür are with Microsoft Corporation, 1065 La Avenida, Mt. View 94041 USA e-mail: (Anoop.Deoras,Ruhi.Sarikaya)@microsoft.com, (gokhan.tur,dilek)@ieee.org.

Joint Decoding, MaxEnt, CRF, SLU, ASR, Lattice Decoding, spoken language processing, speech and dialog understanding

I. Introduction

Spoken language understanding (SLU) systems aim to automatically identify the intent of the user as expressed in natural language, extract associated arguments or slots, and take actions accordingly to satisfy the user's requests [1]. The SLU task is mostly coined by the DARPA (Defense Advanced Research Program Agency) Airline Travel Information System (ATIS) project during early 90s [2]. The ATIS task consisted of spoken queries on flight-related information. An example utterance is *I want to fly to Boston from New York next week*. Understanding was reduced to the problem of extracting task-specific arguments/slots, such as *Destination* and *Departure Date*. Participating systems employed either a data-driven statistical approach [3], [4] or a knowledge-based approach [5], [6], [7].

The state-of-the-art approaches for slot filling [8], [9, among others] use discriminative statistical models, such as conditional random fields, (CRFs) [10], for modeling. More formally, slot filling is framed as a sequence classification problem to obtain the most probable slot sequence:

$$\hat{\mathbf{C}} = \operatorname*{argmax}_{\mathbf{C}} P(\mathbf{C}|\mathbf{W})$$

where $\mathbf{W} = w_1, ..., w_T$ is the input word sequence and $\mathbf{C} = c_1, ..., c_T, c_t \in \mathcal{C}$ is the sequence of associated class labels $\in \mathcal{C}$.

In this study, we follow the popular IOB (in-out-begin) format in representing the data as shown below.

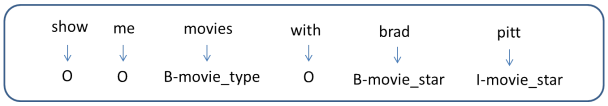


Fig. 1. IOB format for representing slot tags for a typical query in movies domain.

Almost simultaneously with the semantic frame filling-based SLU approaches, a new task emerged motivated by the success of the early commercial interactive voice response (IVR) applications used in call centers. The SLU was framed as *classifying users' utterances into predefined categories* (called as *intents* or *call-types*) [11]. For example, movies domain may include intents such as *get_review*, *find_director*, etc.

For intent determination, early work on discriminative classification algorithms was completed on the AT&T HMIHY system [11] using Boosting [12]. Discriminative call classification systems employing large margin classifiers (e.g., support vector machines – SVMs [13]) include work by Haffner *et al.* [14], who proposed a global optimization process based on an optimal channel communication model that allowed a combination of *heterogeneous* binary classifiers.

In today's state-of-the-art spoken language 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 an Automatic Speech Recognition engine (ASR) with a series of understanding modules such as slot sequence classifiers and intent detectors. The output of these two modules is then fed into an interpreter which interprets the semantics of the spoken query and outputs the desired results depending upon the application. Fig. 2(a) represents such a cascaded pipeline. In this paper, we propose a joint decoding scheme where word recognition and semantic tagging are jointly optimized, resulting in improved word sequence and slot sequence hypotheses. The output of joint decoding is then used for intent detection. Fig. 2(b) represents the proposed joint decoding setup.

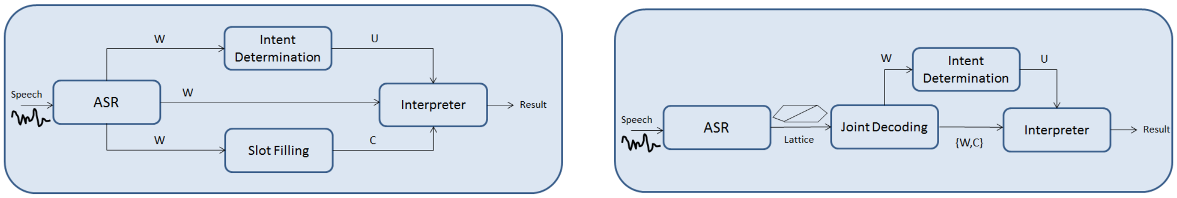
More formally, given an acoustic signal, $\mathbf{A}$, ASR outputs the most likely word sequence, $\mathbf{W}^*$ given by

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

[15]. Typically in cascade systems, this ASR 1-best hypothesis, $\mathbf{W}^*$, is then fed into the SLU system (say targeting slot sequence classification and intent detection) to output the most likely sequence of slots, $\mathbf{C}^*$ and the most likely intent class U^*, given by:

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

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



(a) SLU Cascade Architecture

(b) SLU Proposed Joint Decoding Architecture

Fig. 2. SLU Architectures.

Unfortunately, ASR is usually far from perfect and typically ASR's 1-best hypothesis suffers from many word errors leading to errors in the prediction of slot sequence and intent. Previously, researchers have addressed this issue by incorporating more information from ASR search spaces such as confusion networks, N-best lists and word lattices. For instance, recently, Kurata et.al. [16] proposed an approach for named entity recognition on word confusion networks of 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: $\arg\max_{\mathbf{C}\in\mathcal{C}} P(\mathbf{C}|\mathbf{A}) \approx \arg\max_{\mathbf{C}\in\mathcal{C}} P(\mathbf{C}|\phi(\mathbf{A}))$, where $\phi(\mathbf{A})$ can capture ASR confusions easily obtained from confusion networks, N-best lists or even word lattices. Yaman et.al, [17] have also tried to model $P(\mathbf{C}|\mathbf{A})$ explicitly. They 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 classification, they focused only on utterance classification task (similar to the intent determination task).

In the above and related other approaches such as [18], only slot sequence classification performance and/or utterance classification performance is targeted by modeling only slot sequences/intent class, directly given acoustics without worrying about improving the ASR performance. Although improving ASR performance may not be that essential from a minimalistic view of just slot sequence classification or intent detection, spoken language understanding, however, encompasses other important tasks such as interpretation, the performance of which improves with better hypotheses (word sequence, slot sequence and user intent). It thus becomes necessary to improve performance of both ASR and SLU modules. In this paper, we extend our previously proposed 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 they jointly maximize the posterior probability given acoustics [19]. Formally:

$$\{\mathbf{C},\mathbf{W}\}^* = \underset{\mathbf{C}\in\mathcal{C},\mathbf{W}\in\mathcal{W}}{\operatorname{argmax}} P(\mathbf{C},\mathbf{W}|\mathbf{A}) \tag{4}$$

It has been shown that sometimes lower word error rate does not imply better semantic parsing performance [20], [21]. In their work, authors obtained much better semantic performance, but at the expense of poor word recognition accuracies. It is, however, still widely believed that better word accuracies imply better understanding accuracies. In this extended version of our paper, we have performed extrinsic evaluation experiments, by using the improved output of joint decoding and finding the most likely intent, an utterance classification task. We show that joint decoding not only result in better slot and word accuracies, but improved word accuracies further help intent detection performance. The improved slot sequence and predicted intent class are then fed as an input to other SLU modules such as interpreter. Apart from the additional experimentation and analysis, we have added detailed description of our model and search strategy.

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

II. Mathematical Formulation

As discussed above, we wish to find pair of sequences composed of words and slots such that given the acoustics, the posterior probability of this pair is maximized:

$$\begin{aligned}\{\mathbf{C},\mathbf{W}\}^* &= \underset{\mathbf{C}\in\mathcal{C},\mathbf{W}\in\mathcal{W}}{\operatorname{argmax}} P(\mathbf{C},\mathbf{W}|\mathbf{A}) \\ &= \underset{\mathbf{C}\in\mathcal{C},\mathbf{W}\in\mathcal{W}}{\operatorname{argmax}} P(\mathbf{C}|\mathbf{W},\mathbf{A})P(\mathbf{W}|\mathbf{A}) \\ &= \underset{\mathbf{C}\in\mathcal{C},\mathbf{W}\in\mathcal{W}}{\operatorname{argmax}} P(\mathbf{C}|\mathbf{W})P(\mathbf{A}|\mathbf{W})P(\mathbf{W}) \end{aligned} \tag{5}$$

where the first term of the last step follows from the assumption that when conditioned on the word sequences, slot sequences and acoustics are independent. Last two terms of the last step follows from the Bayes' rule and ignoring $P(\mathbf{A})$, since $\mathbf{A}$ is observed and hence fixed. We restrict the space of word sequence hypotheses to those possible in the word lattice obtained after decoding acoustic observation with ASR engine. In practice, we use a scaling parameter on tagging model – $P(\mathbf{C}|\mathbf{W})$, to compensate the differences between the dynamic ranges of ASR scores and tagging score. Similarly, a scaling parameter on language model is used to balance the dynamic range of acoustic model score and language model score. The formulation then becomes:

$$\{\mathbf{C},\mathbf{W}\}^* = \underset{\mathbf{C}\in\mathcal{C},\mathbf{W}\in\mathcal{W}}{\operatorname{argmax}} \underbrace{P(\mathbf{C}|\mathbf{W})^{\gamma}}_{\text{Tagging Model}} \times \underbrace{(P(\mathbf{A}|\mathbf{W}) \times P(\mathbf{W})^{\alpha})}_{\text{ASR Models}} \tag{6}$$

Typically, $P(\mathbf{C}|\mathbf{W})$ is modeled directly either using discriminative models such as Maximum Entropy (MaxEnt) [22] framework[1] or Conditional Random Field (CRF) [10] framework or more traditionally as a generative model by invoking Bayes' rule and putting a prior on the slot sequence i.e. $P(\mathbf{C})$.

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

In the context of language modeling, Erdogan et.al. [23] proposed a joint semantic-lexical language model to improve speech recognition performance. They showed how semantic information captured via shallow semantic parses or full hierarchical parses improved language modeling performance when the lexical and semantic concepts were predicted jointly. In their work, authors assumed that the probability of a word sequence can be approximated by the joint distribution of the word sequence and its corresponding semantic concept sequence $\Big(P(W) \approx P(W, C_W)\Big)$. Such an approach was used for ASR task alone. Joint decoding formulation specifically for spoken language understanding has previously been proposed by Pieraccini et.al. [24] where they modeled the relation between $\mathbf{C}$ and $\mathbf{W}$ conditioned on the acoustics, 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. [25] too formulated the joint decoding problem similarly. They demonstrated reductions in concept error rate sometimes at the expense of increased word error rate. In these works, they decompose the problem as follows:

$$P(\mathbf{W}, \mathbf{C}|\mathbf{A}) = \frac{P(\mathbf{A}|\mathbf{W}, \mathbf{C}) \times P(\mathbf{W}|\mathbf{C}) \times P(\mathbf{C})}{P(\mathbf{A})}$$

where the denominator, being a constant, is ignored during the search for optimal $\{\mathbf{W}, \mathbf{C}\}$ pair. $P(\mathbf{C})$ in the above formulation captures prior on the slot/tag sequence. It is modeled using an n-gram language model trained from the data comprising of sequence of slots. Similarly the conditional model $P(\mathbf{W}|\mathbf{C})$ is decomposed as:

$$P(\mathbf{W}|\mathbf{C}) = \prod_t P(w_t|w_1^{t-1}, \mathbf{C}) \approx \prod_t P(w_t|w_1^{t-n+1}, \mathbf{C}).$$

Invoking Markov assumption: $\mathbf{A} \rightarrow \mathbf{W} \rightarrow \mathbf{C}$, the first term in the numerator is simplified as $P(\mathbf{A}|\mathbf{W})$. The entire model is represented as a finite state automaton by successive compositions and the search for optimal pair is found out by the shortest path search algorithm. The problem with such an approach, as with any other Hidden Markov Model (HMM) system, in contrast with discriminative classifiers such as Maximum Entropy classifiers, is that they do not provide a convenient framework for capturing diverse features. Restricting the conditional model to word and/or concept n-grams limits the ability to capture sentence level features. It has been studied elsewhere in literature before how HMMs under-perform in comparison to MaxEnts and CRFs (see [26]). Due to these reasons, we decompose the joint decoding problem differently, making use of state-of-the-art discriminative models. We also search for the optimal pair of sequences by doing *sequence tagging* on word lattices. This, in our opinion, makes the formulation novel and distinct from previous work.

A. CRFs and MaxEnts

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

$$P(\mathbf{C}|\mathbf{W}) = \frac{1}{Z(\mathbf{W})}\prod_{t=1}^{T}\psi_t\Big(c_t, \phi(c_1^{t-1}), \gamma_t(\mathbf{W})\Big) \tag{7}$$

where, $\phi(c_1^{t-1})$ is an equivalence classification of the slot sub-sequence up to the $(t-1)^{th}$ position. In the first order linear chain CRFs, this function returns c_{t-1}. $\gamma_t(\mathbf{W})$ is a position specific equivalence classification of the entire word sequence. In practice, this function returns w_{t-n+1}^{t+n-1} i.e., n-grams around the t^{th} word. $\psi(\cdot)$ is the local potential function. This is nothing but an exponentiated weighted sum of active features. $Z(\mathbf{W})$ is the partition function that ensures the probability of all state sequences sum to 1. It is simply given as: $Z(\mathbf{W}) = \sum_{\mathbf{C}}\prod_{t=1}^{T}\psi_t\Big(c_t, \phi(c_1^{t-1}), \gamma_t(\mathbf{W})\Big)$.

We will rewrite Eqn.7 as shown below, for reasons that will become clear later:

$$\begin{aligned} P(\mathbf{C}|\mathbf{W}) &= \frac{1}{Z(\mathbf{W})}\prod_{t=1}^{T}\psi_t(c_t, c_{t-1}, \gamma_t(\mathbf{W})) \\ &= \prod_{t=1}^{T}\frac{\psi_t(c_t, c_{t-1}, \gamma_t(\mathbf{W}))}{Z(\mathbf{W})^{1/T}} \end{aligned} \tag{8}$$

Maximum Entropy 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 MaxEnt 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 in practice, $\phi(c_1^{t-1})$ returns c_{t-1}, similar to linear chain CRFs, and $\gamma_t(\mathbf{W})$ returns w_{t-n+1}^{t+n-1}. Each individual distribution is then modeled as shown below:

$$P(c|c', \gamma(\mathbf{W})) = \frac{\psi(c, c', \gamma(\mathbf{W}))}{Z(c', \gamma(\mathbf{W}))} \tag{9}$$

where $\psi(\cdot)$ is the potential function and $Z(\cdot)$ is a normalization constant.

Similar to CRFs, the potential is an exponentiated weighted sum of active features. Partition function is simply given as: $Z(c', \gamma(\mathbf{W})) = \sum_c \psi(c, c', \gamma(\mathbf{W}))$. It ensures that the probability of all states sum to 1.

Plugging the local distribution function in the chain rule, we get:

$$\begin{aligned} P(\mathbf{C}|\mathbf{W}) &= \prod_{t=1}^{T} P(c_t|c_{t-1},\mathbf{W}) \\ &= \prod_{t=1}^{T} \frac{\psi(c_t,c_{t-1},\gamma_t(\mathbf{W}))}{Z(c_{t-1},\gamma_t(\mathbf{W}))} \end{aligned} \tag{10}$$

It should be noted from (8) and (10) that MaxEnt models allow us to linearly decompose the sequence probability into series of *normalized* local potentials. CRF models, on the other hand, decompose the sequence probability into series of local potentials but due to a global normalization constant, these potentials are *unnormalized*. It is thus clear that the search for optimal pair of slot and word sequences on word lattices using the formulation described in Eqn. 5, 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 the local nature of MaxEnt models, we can make use of dynamic programming on word lattices, something much more tractable. Computation of Z for MaxEnt 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). In this paper, we propose to use MaxEnt models in conjunction with ASR acoustic model and language models to solve (5). However, our framework is general enough to accommodate any probabilistic local model. Next, we describe in detail the joint decoding framework using MaxEnt discriminative models.

B. Joint Decoding Framework using MaxEnts

We invoke chain rule on the objective function of (5) to obtain:

$$\begin{aligned} & P(\mathbf{C}|\mathbf{W})P(\mathbf{A}|\mathbf{W})P(\mathbf{W}) \\ & = \prod_{t=1}^{T} P(c_t|c_1^{t-1},\mathbf{W})P(\mathbf{a}_t|w_t)P(w_t|w_1^{t-1}) \\ & \approx \prod_{t=1}^{T} P(c_t|c_{t-1},\gamma_t(\mathbf{W}))P(\mathbf{a}_t|w_t)P(w_t|w_{t-n+1}^{t-1}) \end{aligned} \tag{11}$$

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 substring, $\mathbf{a}_t$, 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 itself. We also work with n-gram language models (LM) and restrict the context at any t^{th} position to contain previous $n-1$ words only. In our work, if

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

C. Joint Decoding Framework using CRFs

In order to solve (5) with CRFs as the tagging model, it is necessary to explicitly compute the partition function for all competing word sequences. It is not clear how exactly we can make use of CRF models for joint decoding on word lattices. Word lattices typically encode huge number of full sentence hypotheses and enumerating them all can be computationally expensive. One way is to extract top N best hypotheses from word lattices and for each hypothesis, extract K best slot sequences using CRF model. In this $N \times K$ space of word and slot sequences, we can then exhaustively search for the best pair, i.e. the pair which maximizes the joint probability given acoustics. Algorithm 1 describes the step:

Algorithm 1 Joint Decoding with CRF Models using N best hypotheses

```
Require: Word Lattice, $\mathcal{L}$
  return Most probable pair of sequences: $\{W, C\}^*$
  $M = -1$
  for $W \in N$-best from $\mathcal{L}$ do
    for $C \in K$-best from CRF tagging of $W$ do
      if $P(W|A) \times P(C|W)^\gamma > M$ then
        $M = P(W|A) \times P(C|W)^\gamma$
        $\{W, C\}^* = \{W, C\}$
      end if
    end for
  end for
```

The other alternative to solve joint decoding problem on word lattices with CRF models would be to assume that the partition function value is same for all word sequences possible in the word lattice, $\mathcal{L}$ i.e. $\forall$ $\mathbf{W}$ and $\mathbf{W}' \in \mathcal{L}$, $Z(\mathbf{W}) = Z(\mathbf{W}')$. This approximation allows us to factorize the sequence probability into a series of potentials, without needing to normalize them. If we represent this common partition function value by simply Z, then using Eqn. 11 and Eqn. 8, the joint probability of word and slot sequence given acoustics can be written as:

$$
\begin{aligned}
&\frac{1}{P(\mathbf{A})} \times P(\mathbf{C}|\mathbf{W})P(\mathbf{A}|\mathbf{W})P(\mathbf{W}) \\
&\propto P(\mathbf{C}|\mathbf{W})P(\mathbf{A}|\mathbf{W})P(\mathbf{W}) \\
&= \prod_{t=1}^{T} \frac{\psi_t(c_t|c_1^{t-1}, \gamma_i(\mathbf{W}))}{Z^{1/T}} \times P(\mathbf{a}_t|w_t) \times P(w_t|w_1^{t-1}) \\
&\propto \prod_{t=1}^{T} \psi_t(c_t|c_1^{t-1}, \gamma_i(\mathbf{W})) \times P(\mathbf{a}_t|w_t) \times P(w_t|w_1^{t-1}) \qquad (12)
\end{aligned}
$$

The above formulation allows us to make use of dynamic programming (Viterbi Decoding) to search for most likely pair of sequences. Details of Viterbi decoding on lattices for outputting pair of word and slot sequence is described in detail in the next section.

III. Lattice Expansion and Viterbi Decoding

A word lattice, $\mathcal{L}$, obtained at the output of the first pass decoding, encodes an exponential number (exponential in the number of states (nodes) present in the lattice) of hypotheses in a very compact data structure. It is a directed acyclic graph $G = (V, E)$, where V and E denote set of vertices (nodes / states) and edges (arcs / links). We assume this graph to have a unique start state. Similarly, we assume that each edge of the lattice is associated with a word label, w, and the corresponding acoustic model and language model probability. In spite of using n-gram language model in the first pass decoding, the lattices typically end up **not** maintaining unique $n-1$ word context at all the edges. This is typically due to back offs in the first pass language model or due to limited cross word context.

In order to assign language model (LM) probability for a word given previous words and probability of some slot type given the context of previous and next words and previous slot type (typical feature selection for discriminative classifiers), it is essential that the lattice maintains at-least as much unique left and right context, at every arc, as is required for the prediction.

Odell [27] and Weng et.al. [28] describe an algorithm to achieve node splitting for trigram LM rescoring on word lattices produced using a bigram LM. They show how to maintain an unambiguous left context of 2 words for any word in the lattice. We propose a node splitting algorithm, which is similar in spirit to the ones mentioned above, but differs in the fact that it is generalized to n-gram (left) 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 is used to either split or merge the nodes, thus avoiding the need to process all outgoing arcs from the current node. We also show how to use this algorithm in conjunction with a certain finite state machine operation to also maintain an unambiguous right context for n-gram processing.

Fig. 3(a) shows a typical toy word lattice produced after determinization and minimization [29]. At any arc, if we were to find out 2 word history for the purpose of LM rescoring, we won't be able to do so unambiguously. For instance, if we look at the edge between node 1 and 2 containing the word c, then since edges containing word a and b both merge at node 1, the left context for the word c is ambiguous. Similarly, if we look at the right context, there too, due to presence of e, g and f, g word sub sequences, the context becomes ambiguous. Thus in order to assign LM scores to c or tag score to any slot label on the arc containing the word c, we won't be able to extract unambiguous left and right context. It is thus necessary to expand the lattice by splitting the nodes. However, exhaustive node splitting will blow up the search space. We thus have to merge back the nodes as soon as they allow for n-gram processing. Fig.3(b) shows the expanded lattice such that at any edge, one is able to find out unambiguous left 2 word context. Similarly, Fig. 3(c) shows the expanded lattice such that at any edge, one is able to find out unambiguous left as well as right 2 word context. Details of the algorithm follows next.

Algorithm 2 illustrates the steps. It iteratively splits and merges the nodes of the input lattice to resolve the left n-gram ambiguities i.e. maintains unambiguous left context of $n-1$ words at every arc. We will introduce some additional 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 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 left 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 (negative log of the combined ASR likelihood), it finds out if the word sequence $\mathbf{v}, w$ exists on any $n-1$ consecutive arcs in G_Q.[2] 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. On the other hand,

[2] 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.

if the word sequence $\mathbf{v}, w$ 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[3] and maintains a unique left context of $n-1$ words thus allowing for unambiguous left n-gram processing.

In our work, we not only need an unambiguous left context, but also an unambiguous right context. In order to do that, we first represent our word lattice as a weighted finite state automaton (W-FSA) [29] and then take the following steps:

1) We use Alg. 2 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 [30].
3) We again use Alg. 2 to expand this 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.

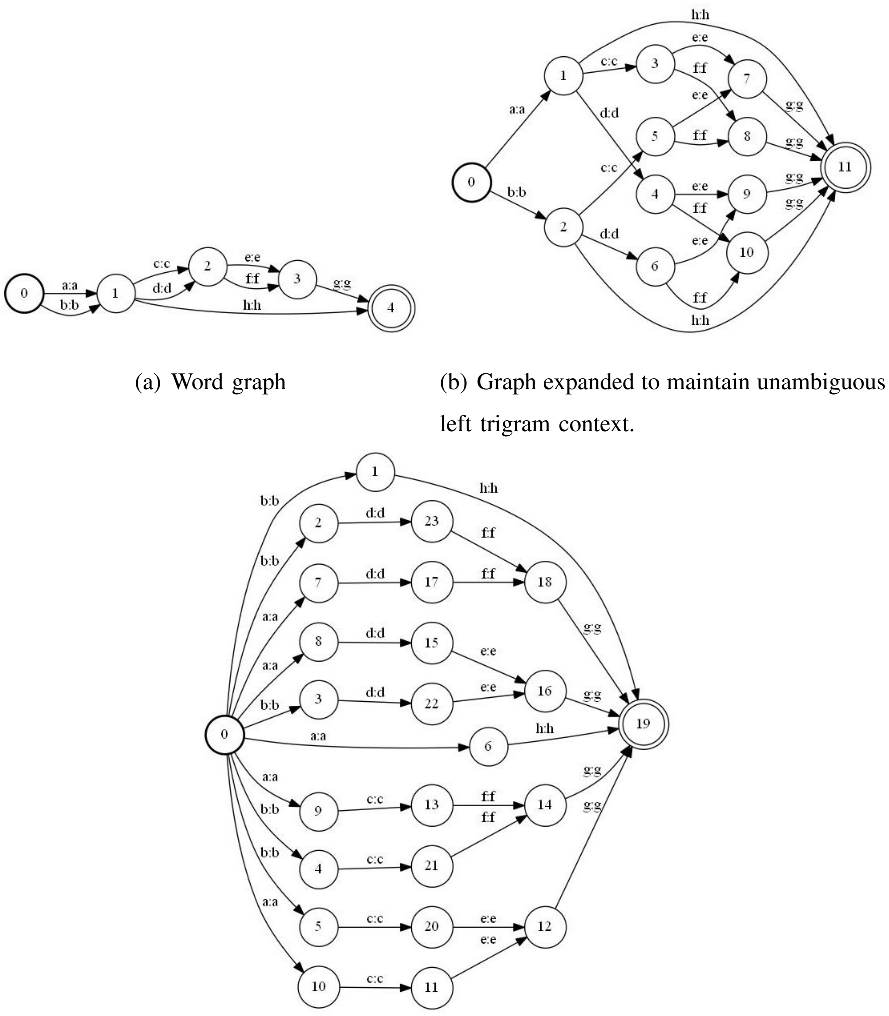
Once we obtain the expanded lattice, we traverse the lattice in a topological order and at every arc, we extract the unambiguous word context (left and right) and for every possible slot type on the previous arc, we obtain a distribution over all slot types on the current arc using the discriminative model (either MaxEnt or CRF with the second approximation discussed in Sec. II-C). 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. We discuss Viterbi Decoding in the next section.

Typical speech lattices contain many duplicate sub paths. We hence determinize and minimize our lattices so that we start off with as compact a lattice as possible. Similarly, 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 the OpenFST toolkit. 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.

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

Algorithm 2 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 $n \in V_L$ in topological order do
    for $s \in V_L : \{s, n\} \in E_L$ do
      for $p_0 \in V_Q : p_0 \in \mathcal{S}_s$ do
        $\mathbf{v} = \text{word}(\{p_{k-2}, p_{k-3}\}) \cdot \ldots \cdot \text{word}(\{p_2, p_1\})$
            $\cdot\text{word}(\{p_1, p_0\})$
            $\forall i \in \{0, \ldots, k-3\} : \{p_{i+1}, p_i\} \in E_Q$
        for $w \in \text{word}(\{s, n\})$ do
          $c = \text{cost}_w(\{s, n\})$
          {// 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
            $\mathcal{H}(\{\mathbf{v}, w\}) = m$
            $\mathcal{S}_n = \{\mathcal{S}_n, m\}$
            $V_Q = \{V_Q, m\}$
            $E_Q = \{E_Q, \{p_0, m\}\}$
            $\text{word}(\{p_0, m\})$=$w$
            $\text{cost}(\{p_0, m\})$=$c$
            $m$++
          end if
        end for
      end for
      Reset $\mathcal{H}$
    end for
  end for
```



(a) Word graph

(b) Graph expanded to maintain unambiguous left trigram context.

(c) Graph expanded to maintain unambiguous left as well right trigram context.

Fig. 3. Toy word graph and its expanded representations

A. Viterbi Decoding

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. Before expanding them, we first append our determinized and minimized speech word lattices with a begin of sentence token : BOS. Similarly, we attach arcs from all the end states to a new and unique end state. We associate end of sentence marker : EOS on all these arcs. In the initialization step of Viterbi decoding, we assign most likely joint probability value, denoted by $\gamma_{c,e}$, to each slot type, c, for the BOS arc, e. The vocabulary of our slots also contain a special token : O which can be thought of as a label not belonging to any of the defined slot types. We assign $\gamma_{c,e} = 1$ for c = `O` and $w(e)$ = `BOS`, where $w(e)$ denotes the word on the arc e. For all the remaining other slot labels, c', we assign $\gamma_{c',e} = 0$. With these initializations, we then progress through the lattice in a topological order and for any possible word-tag pair, find out all possible incoming transitions and remember the one that maximizes the joint probability up to that time.

We will define some additional notation: Given an edge, e, we will define the associated acoustic sub string by $\mathbf{a}(e)$. We will define the set of incoming edges (i.e. transitioning into e) by $\mathcal{N}(e)$. $\mathbf{w}(L(e))$ will denote the unambiguous context of words appearing in the left context of the edge, e. Similarly, $\mathbf{w}(R(e))$ will denote the unambiguous context of words appearing in the right context of the edge, e. Since the lattice is assumed to maintain an unambiguous left and right context at every edge, $\mathbf{w}(L(e))$ (and $\mathbf{w}(R(e))$) will return the same set of words irrespective of which set of arcs we traverse from e in the left (and right) direction. However, we have to be careful about the ASR scores (combined AM and LM scores). Although the edge, e, maintains unambiguous context of some finite set of words in left and right direction, their individual acoustic model score could be different leading to different ASR scores. It is hence important to evaluate all possible combinations of word-slot pairs available in the left context of the edge under consideration. Thus for any arc, e, we iterate over all slot types ($c' \in \mathcal{C}$) and all incoming edges ($e' \in \mathcal{N}(e)$) and find out the best forward probability transition among these and remember it using back pointers. We traverse the lattice in a topological order and use the following recursion:

$$\begin{aligned} \gamma_{c,e} = & \max_{c' \in \mathcal{C}, e' \in N(e)} \gamma_{c',e'} \times P\Big(c|c', \mathbf{w}(L(e)), w(e), \mathbf{w}(R(e))\Big) \\ & \times P\Big(\mathbf{a}(e)|w(e)\Big) \times P\Big(w(e)|\mathbf{w}(L(e))\Big) \end{aligned} \tag{13}$$

With the aid of back pointers, we traverse back from the last edge, containing the special word token `EOS` and the special slot token `O`, following the path of most likely incoming transition until we reach the starting edge containing the word `BOS` and tag `O`.

As discussed earlier, $P\Big(c|c', \mathbf{w}(L(e)), w(e), \mathbf{w}(R(e))\Big)$ is modeled as a MaxEnt model. In order to use CRF models in this setting, we assume that the normalization constant is irrelevant and replace $P\Big(c|c', \mathbf{w}(L(e)), w(e), \mathbf{w}(R(e))\Big)$ with $\psi\Big(c, c', \mathbf{w}(L(e)), w(e), \mathbf{w}(R(e))\Big)$, i.e. the local potential, everything else being the same.

IV. Experiments and Results

We have focused on slot sequence tagging and intent detection tasks for spoken language understanding on Microsoft's SLU system. Our SLU 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. 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 acoustic models (AM) is trained with most state-of-the-art technologies, such as discriminative training, speaker adaption etc. Our n-gram 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 slot sequence tagger was trained with 2 kinds of discriminative models – Maximum Entropy (MaxEnt) and Conditional Random Fields (CRF). We made use of Wapiti tool [31] to train these models.

A. Data

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 MaxEnt 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 MaxEnt 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.

B. Metrics

We use word error rate (WER) as the metric to measure the quality of word sequence hypotheses output by either ASR or by the proposed joint decoding model. For sequence tagging tasks, conventionally, researchers have resorted to F measure metrics. We follow suit for our slot sequence tagging performance measurement.

For scoring the hypothesized slot sequence, $\hat{\mathbf{C}}$, for any word sequence, $\hat{\mathbf{W}}$ (typically output of ASR or joint decoding setup), we need to form sequence of reference slots to be compared against. To do this, we first propagate the true slot sequence, $\mathbf{C}$, (attached to manual transcriptions, $\mathbf{W}$, of underlying spoken utterance) to the word sequence under consideration i.e. $\hat{\mathbf{W}}$. We do this by first aligning $\mathbf{W}$ and $\hat{\mathbf{W}}$ using string edit distance (which we also have to do for WER computation) and then simply propagate the tags associated with the aligned reference words. Wherever there are false insertions in $\hat{\mathbf{W}}$ with respect to $\mathbf{W}$, we insert the tag $\texttt{O}$ in the aligned reference slot sequence in the appropriate location. This way, we penalize any falsely inserted words and hence falsely tagged slot in the hypothesized slot sequence. We also penalize false deletion of words in $\hat{\mathbf{W}}$ with respect to $\mathbf{W}$ by inserting dummy tokens in $\hat{\mathbf{W}}$ and attaching the corresponding true tags associated with $\mathbf{W}$ in the aligned reference slot sequence in the appropriate location. Since falsely deleted words are never processed during slot tagging, we hypothesize tag $\texttt{O}$ for all these words. This way, we penalize false insertions and false deletions.

C. Results

Table I shows performance of various methods under various configurations. ME-L and ME-LR correspond to MaxEnt 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 best 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 output, which maximizes the joint probability of words and tags given acoustic signal (Jnt.Dec.). Setups (a), (b) and (c) correspond to cascade system where (b) is with possibly the best word sequence one could hope to obtain out of ASR and defines the upper bound for possible performance improvements. Setup (d) corresponds to the proposed joint decoding approach, where both word & slot sequence is output given acoustics.

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 77.0% 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 2.2% absolute improvement over ME-LR and CRF-LR cascade baselines.

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 $\mathbf{1.3}\%$ ($19.7\% \rightarrow 18.4\%$) absolute lower in joint decoding setup. Thus our proposed method not only improves tagging accuracy of slots, but also reduces the WER of the hypotheses.

As noted earlier, joint decoding with CRF models is intractable due to explicit need to compute partition

(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**	**17.6**	**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	77.05
(d) Jnt.Dec.	18.7	75.8	**18.4**	**79.2**	**18.3**	**79.0**

TABLE I

Performance (WER (%) and F measure) of cascade and proposed joint decoding technique. While we make use of word lattices for joint decoding with MaxEnt models, we restrict our search space to N best lists when CRF models are used.

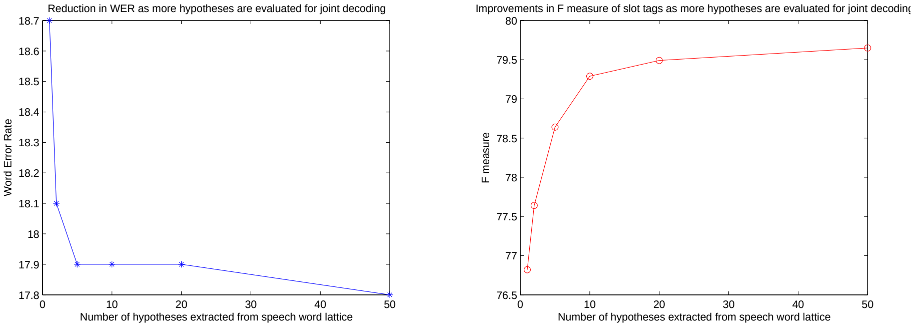
Setup	Dev		Eval	
	WER	F	WER	F
(a) Jnt. Dec Lat.	18.1	78.5	19.1	77.9
(b) Jnt. Dec. NBest.	17.6	79.7	18.3	79.0

TABLE II

Performance (WER (%) and F measure) of joint decoding technique using CRF-LR model. In (a), partition function value is assumed to be constant while processing word lattices, while in (b), partition function value is explicitly computed for every hypothesis in N best list extracted from the word lattice.

function for every word sequence. In Table II, we compare the performance of joint decoding with CRF models when either the partition function value is assumed constant or when it is explicitly computed but the search space is restricted to N best lists instead of the entire word lattice. We can see from the results that ignoring the partition function value hurts the overall performance. Search on N best lists by explicitly computing the partition function for every word sequence hypothesis gives the best overall results, both in terms of WER and F measure. We restricted our search space to top 50 word sequence hypotheses and for every word sequence extracted from speech word lattice, we extracted just top 2 slot sequences. Joint decoding performance with CRF on N best lists is almost the same as that with ME-LR model on word lattices (Table I).

Fig. 4 shows how the speech recognition and slot tagging accuracy improves when more hypotheses are used during joint decoding with ME-LR model. For every word sequence hypotheses, we extracted top 2 slot sequences. The joint decoding performance with ME-LR model on lattices is matched when about 50 unique hypotheses are extracted from word lattice and for each hypotheses, top 2 slot sequences are analyzed.



(a) Reduction in WER as more hypotheses are evaluated for joint decoding

(b) Improvements in F measure as more hypotheses are evaluated for joint decoding

Fig. 4. Recognition and Slot Tagging performance improves when more hypotheses are evaluated for joint decoding.

D. Analysis

In situations where both acoustic model and language model are incapable of discriminating the lexical alternatives, use of tagging model turns out to be very helpful. For instance, in the real example from evaluation corpus, the user says “*show me westerns with john wayne*”. ASR mis-recognizes the word ‘westerns’ and instead substitutes it with the word ‘western’ and also falsely deletes the word ‘with’. This in turn makes the tagging model to put a huge probability mass on the slot type ‘movie-name’ for the lexical tokens ‘western john wayne‘, probably because some movies have the lexical token ‘western’ in their names and also probably because ‘show me’ lexical tokens are usually followed by some movie name, rather than an actor name (which otherwise would have seemed more likely due to presence of ‘john wayne’). In the joint decoding approach, the presence of ‘john wayne’ as the actor name makes ‘westerns with’ as more likely lexical candidates than ‘western <epsilon>’, more so because the slot type of ‘westerns’, which is ‘movie-genre’ goes very naturally with the lexical tokens such as ‘with’ or ’featuring’ etc, which in turn goes naturally with some movie actor name.

Error Type	Lat 1B	Jnt.Dec.
(a) WER	19.7	18.4
(b) Sub.E.	11.3	9.7
(c) Ins. E.	1.9	1.7
(d) Del.E.	6.5	7.0

TABLE III

Different string edit errors for hypotheses output by ASR (Lat 1B) and the proposed joint decoding setup (Jnt.Dec.).

By doing the analysis more quantitatively, we observed that the recognition output from joint decoding had considerably fewer substitution errors when compared to the baseline i.e. one best output from ASR decoding. False insertion errors were slightly lower, however, we observed slightly higher false deletion errors. Table III shows the percentage distribution of various string edit errors. In about one third of the instances where joint decoding outperformed cascade setup involved cases where ASR 1bests were missing prepositions such as *with*, *by* etc. Majority of the remaining instances where joint decoding outperformed cascade involved errors in 1best due to either false substitution of a (compound) word with 2 or more words (*gallipeli* → *the lovely*, *freejack* → *free jack*) or vice versa where 2 or more words were falsely substituted with one (compound) word (*play now* → *playhouse*, *start over* → *server*). There were no instances where cascade outperformed joint decoding suggesting that joint decoding was able to only correct the mistakes rather than introducing them.

E. Extrinsic Evaluation on Intent Determination

Achieving significantly better results for optimizing recognition for slot filling may not necessarily propagate to other semantic parsing tasks as shown before [20]. To study this effect, we performed an extrinsic evaluation using the sister semantic modeling task of intent determination. To this end, we trained utterance classifiers using two state-of-the-art algorithms – Boosting [12] and SVMs [13] (we used icsiboost [32] and libsvm [33] respectively). We used word trigram features for training these models. We used the same training, dev and eval corpus. Our setup had a total of 31 intents.

From Table IV, we can see that we improve the intent determination performance by as much as 1.2% absolute ($10.83\% \rightarrow 9.57\%$). By carrying out experiments with state-of-the-art classifiers, we have also made sure that the improvements are consistent across classifiers. It is very rare to obtain same level of improvements in intent detection given an improvement in ASR. Joint decoding systematically fixes more important errors leading to improvements in intent detection tasks.

(a) Development data set

Setup	Boost	SVM
(a) Manual	4.02	3.58
(b) Lat OB	6.00	5.11
(c) Lat 1B	9.07	8.05
(d) Jnt.Dec.	**8.22**	**7.15**

(b) Evaluation data set

Setup	Boost	SVM
(a) Manual	6.00	5.79
(b) Lat OB	8.00	7.55
(c) Lat 1B	11.34	10.83
(d) Jnt.Dec.	**9.78**	**9.57**

TABLE IV

Performance on intent determination task (Error Rate (%)) of cascade and proposed joint decoding technique.

V. Conclusions

In this paper, we presented a novel decoding framework for joint recognition and semantic tagging of hypotheses. We demonstrated significant improvements in both recognition and semantic tagging accuracy over cascade approach. Joint recognition further helped to improve the intent detection performance.

As part of our future work, we plan to do joint decoding of not only words and tags, but also other SLU attributes such as intents and domains. Apart from using more sophisticated discriminative classifiers such as the ones based on neural networks, we plan on investigating various other approximation strategies for incorporating global models such as CRFs in the joint decoding setup. From the feature engineering point of view, we plan to use features derived from named entity gazetteers and some more such as pre- and suffix-features etc.

References

[1] G. Tur and R. D. Mori, Eds., *Spoken Language Understanding: Systems for Extracting Semantic Information from Speech*. New York, NY: John Wiley and Sons, 2011.

[2] P. J. Price, “Evaluation of spoken language systems: The ATIS domain,” in *Proceedings of the DARPA Workshop on Speech and Natural Language*, Hidden Valley, PA, June 1990.

[3] R. Pieraccini, E. Tzoukermann, Z. Gorelov, J.-L. Gauvain, E. Levin, C.-H. Lee, and J. G. Wilpon, “A speech understanding system based on statistical representation of semantics,” in *Proceedings of the ICASSP*, San Francisco, CA, March 1992.

[4] S. Miller, R. Bobrow, R. Ingria, and R. Schwartz, “Hidden understanding models of natural language,” in *Proceedings of the ACL*, 1994.

[5] W. Ward and S.Issar, “Recent improvements in the CMU spoken language understanding system,” in *Proc. of the ARPA HLT Workshop*, 1994.

[6] S. Seneff, “TINA: A natural language system for spoken language applications,” *Computational Linguistics*, vol. 18, no. 1, pp. 61–86, 1992.

[7] J. Dowding, J. M. Gawron, D. Appelt, J. Bear, L. Cherny, R. Moore, and D. Moran, “Gemini: A natural language system for spoken language understanding,” in *Proceedings of the ARPA Workshop on Human Language Technology*, Princeton, NJ, March 1993.

[8] Y.-Y. Wang and A. Acero, “Discriminative models for spoken language understanding,” in *Proceedings of the ICSLP*, Pittsburgh, PA, 2006.

[9] C. Raymond and G. Riccardi, “Generative and discriminative algorithms for spoken language understanding,” in *Proceedings of the Interspeech*, Antwerp, Belgium, 2007.

[10] J. Lafferty, A. McCallum, and F. Pereira, “Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data.”

[11] A. L. Gorin, G. Riccardi, and J. H. Wright, “How May I Help You?” *Speech Communication*, vol. 23, pp. 113–127, 1997.

[12] R. E. Schapire and Y. Singer, “Boostexter: A boosting-based system for text categorization,” *Machine Learning*, vol. 39, pp. 135–168, 2000.

[13] V. N. Vapnik, *The nature of statistical learning theory*. New York, NY, USA: Springer-Verlag New York, Inc., 1995.

[14] P. Haffner, G. Tur, and J. Wright, “Optimizing SVMs for complex call classification,” in *Proceedings of the ICASSP*, Hong Kong, April 2003.

[15] F. Jelinek, *Statistical methods for speech recognition*. Cambridge, MA, USA: MIT Press, 1997.

[16] 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.

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

[18] 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*, vol. 20, no. 4, pp. 495–514, 2006.

[19] A. Deoras, R. Sarikaya, G. Tur, and D. Hakkani-Tür, “Joint Decoding for Speech Recognition and Semantic Tagging,” in *Proc. of ISCA INTERSPEECH*, Portland, Oregon, US, 2012.

[20] Y.-Y. Wang, A. Acero, and C. Chelba, “Is Word Error Rate a Good Indicator for Spoken Language Understanding Accuracy,” in *Proc. of the ASRU*, 2003.

[21] C. Chelba, M. Mahajan, and A. Acero, “Speech Utterance Classification,” in *Proc. of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2003.

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

[23] H. Erdogan, R. Sarikaya, S. F. Chen, Y. Gao, and M. Picheny, “Using semantic analysis to improve speech recognition performance,” *Computer Speech and Language*, vol. 219, pp. 321–343, 2005.

[24] R. Pieraccini and E. Levin, “Stochastic Representation of Semantic Structure for Speech Understanding,” *Speech Communication*, vol. 11, no. 2-3, pp. 283–288, Jun. 1992.

[25] 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.

[26] Y.-Y. Wang, A. Acero, M. Mahajan, and J. Lee, “Combining Statistical and Knowledge based Spoken Language Understanding in Conditional Models,” in *Proc. of COLING/ACL*, 2006.

[27] J. Odell, “The Use of Context in Large Vocabulary Speech Recognition,” Ph.D. dissertation, Cambridge University Engineering Department, 1995.

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

[29] M. Mohri, F. Pereira, and M. Riley, "The design principles of a weighted finite-state transducer library," *Theoretical Computer Science, 231:17-32*, 2000.

[30] 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 Ninth International Conference on Implementation and Application of Automata, (CIAA 2007)*, 2007.

[31] T. Lavergne, O. Cappé, and F. Yvon, "Practical very large scale CRFs," in *the Proceedings of ACL*, July 2010, pp. 504–513.

[32] B. Favre, D. Hakkani-Tür, and S. Cuendet, "Icsiboost," http://code.google.come/p/icsiboost, 2007.

[33] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, pp. 27:1–27:27, 2011, software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm.

Anoop Deoras is a research scientist at Microsoft. He received the B.E. degree in Electronics and Telecommunication Engineering from College of Engineering, Pune India in 2003, M.S. degree in Applied Math. & Statistics in 2010 and a M.S. and Ph.D in Electrical & Computer Engineering from Johns Hopkins University in 2011. He is interested in applying machine learning techniques to speech recognition and understanding. He is a member of IEEE, ISCA and ACL.

Gokhan Tur received his Ph.D. degree in Computer Science from Bilkent University, Turkey in 2000. He worked at AT&T Labs - Research, NJ from 2001 to 2006 and at the Speech Technology and Research Lab of SRI International, CA from 2006 to 2010. He is currently with Microsoft Research. His research interests include aspects of spoken language understanding (SLU) and speech & language processing. He is a senior member of IEEE, ACL, and ISCA. He is currently an associate editor for the IEEE TASL-P journal.

Ruhi Sarikaya is a principal scientist and the manager at Microsoft. He was a research staff member at IBM Research for ten years and prior to that a researcher at the University of Colorado at Boulder for two years. He received the B.S. degree from Bilkent University, Turkey in 1995, M.S. degree from Clemson University, SC in 1997 and the Ph.D. degree from Duke University, NC in 2001 all in Electrical and Computer Engineering. His past and present research interests span all aspects of speech and language processing. Dr. Sarikaya is a member of IEEE (senior member), ACL and ISCA.

Dilek Hakkani-Tür is a senior researcher at Microsoft Research. Prior to joining Microsoft, she was a senior researcher at ICSI speech group (2006-2010) and a senior technical staff member at AT&T Labs-Research in Florham Park, NJ (2001-2005). She received her BSc degree from Middle East Technical University, in 1994, and MSc and PhD degrees from Bilkent University, in Computer Engineering, in 1996 and 2000, respectively. Her research interests include natural language and speech processing, spoken dialog systems, and active and unsupervised learning for language processing. She is a member of IEEE (senior member), ISCA and ACL.