STATISTICAL SPOKEN LANGUAGE UNDERSTANDING: FROM GENERATIVE MODEL TO CONDITIONAL MODEL

Ye-Yi Wang[1], John Lee[2] , Milind Mahajan[1] and Alex Acero[1]

[1]Speech Technology Group, Microsoft Research
[2]Computer Science and Artificial Intelligence Laboratory, MIT

ABSTRACT

Spoken Language Understanding (SLU) addresses the problem of extracting semantic meaning conveyed in a user's utterance. Traditionally the problem is solved with a knowledge-based approach. In the past decade many data-driven statistical models have been proposed, all of them are in the generative framework. In our previous study, we have introduced a HMM/CFG composite model. It is a generative model that integrates knowledge-based approach in a statistical learning framework. We have investigated similar integration of prior knowledge and statistical learning in the framework of conditional models recently. This extended summary describes our experiences and presents some preliminary results, which shows a 17% slot error rate reduction over the generative model.

1. INTRODUCTION

Spoken Language Understanding (SLU) addresses the problem of extracting semantic meaning conveyed in a user's utterance. Traditionally the problem is solved with a knowledge-based approach. In the past decade many data-driven statistical models have been proposed for the problem. An introduction to these models can be found in [1]. All of the statistical leaning approaches exploit generative models for SLU. Data sparseness is one of the major problems associated with such approaches. In our previous study, we have proposed a HMM/CFG composite model, another generative model that integrates knowledge-based approach in a statistical learning framework. The inclusion of prior knowledge in the model compensates for the dearth of data for model training. The HMM/CFG composite model achieves the understanding accuracy at the same level as the best performing semantic parsing system based on a manually developed grammar in ATIS evaluation [2]. We have recently exploited conditional models for further improving the understanding accuracy. We have tried direct porting of the HMM/CFG composite model to Hidden Conditional Random Fields (HCRFs) [3]. We have failed to obtain any improvement mainly due to the local optimality of the hidden model and vast parameter space. We have then simplified the original model structure and remove the hidden valuables in the HCRF. With the introduction of important non-homogeneous features to the Conditional Random Field (CRF) [4], we have improved the slot insertion-deletion-substitution error rate by 17%. In this extended summary, we will first introduce the HMM/CFG generative model, then discuss the problem of directly porting the model to a HCRF, and finally introduce the CRF and the features we used to obtain the best SLU result on ATIS test data.

2. THE GENERATIVE MODEL

The HMM/CFG composite model [1] adopts a pattern recognition approach to SLU. Given a word sequence W, a SLU component needs to find the semantic representation of the meaning M that has the maximum *a posteriori* probability $\Pr(M \mid W)$:

$$\hat{M} = \arg\max_M \Pr(M \mid W) = \arg\max_M \Pr(W \mid M) \cdot \Pr(M)$$

The composite model integrates domain knowledge by setting the topology of the prior model, $\Pr(M)$, according to the domain semantics; and by using PCFG rules as part of the lexicalization model $\Pr(W \mid M)$.

The domain semantics define an application's semantic structure with s*emantic frames*. Figure 1 shows a simplified example of two semantic frames in the ATIS domain. Figure 2 shows a meaning representation according to the domain semantics.

```
< frame name="ShowFlight" toplevel="true" >
    <slot name="Flight" filler="Flight"/ >
< /frame >
< frame name="GroundTrans" toplevel="true" >
    < slot name="City" filler="City"/ >
< /frame >
< frame name="Flight" >
    <slot name="DCity" filler="City"/ >
    < slot name="ACity" filler="City"/ >
< /frame >
```

Figure 1. Simplified semantic frames in the ATIS domain. The two frames with "toplevel" attribute are also known as commands. The filler specifies the semantic object (covered by the corresponding CFG rule) that can fill a slot. For example, an object that is an instantiation of the **Flight** frame can be the filler for the Flight slot of **ShowFlight** frame, and a string covered by the "**City**" rule in a CFG can be the filler of the ACity (ArriveCity) or the DCity (DepartCity) slot.

```
< ShowFlight" >
   < Flight >
      < DCity type="City" > Seattle < /DCity >
      < ACity type="City" > Boston < /ACity >
   < /Flight >
< /ShowFlight >
```

Figure 2. The semantic representation for "Show me the flights departing from Seattle arriving at Boston" is an instantiation of the semantic frames in Figure 1.

The HMM topology and the state transition probabilities comprise the semantic prior model. The topology is determined by the domain semantics defined by the frames and the transition probabilities can be estimated from training data. Figure 3 shows the topology of the underlying states in the statistical model for the semantic frames in Figure 1.

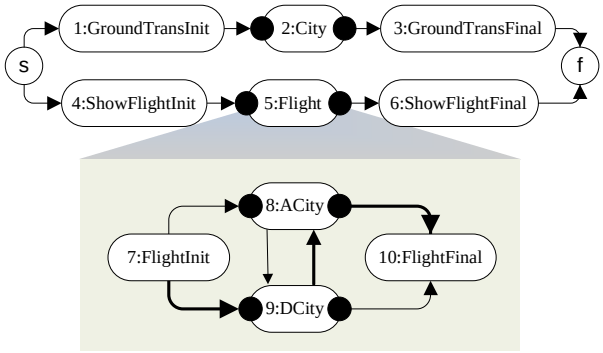

Figure 3. The HMM/CFG model's state topology, as determined by the semantic frames in Figure 1. On the top is the transition network for the two top-level commands. State 1 and state 4 are called *precommands*. State 3 and state 6 are called *postcommands*. States 2, 5, 8 and 9 represent slots. They are actually a three state sequence — each slot is bracketed by a *preamble* and a *postamble* (represented by the dots) that serve as the contextual clue for the slot's identity.

The lexicalization model, $\Pr(W \mid M)$, depicts the process of generating sentences from the topology. It models the distribution for words to emit from the states in the topology. It uses state-dependent n-grams to model the precommands, postcommands, preambles and postambles, and uses CFG to model the fillers of a slot. The use of knowledge-based CFG rules compensate for the dearth of domain-specific data.

Given the semantic representation (training examples) in Figure 2, the state sequence through the topology in Figure 3 is deterministic as show in Figure 4. The alignments of the words to the state in the shaded boxes are not labeled. The parameters in these n-gram models can be estimated with an EM algorithm that treats the alignments as hidden variables.

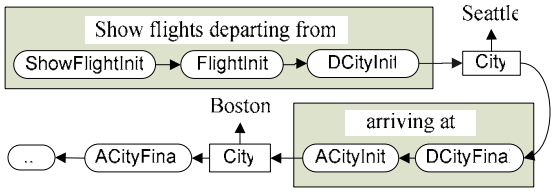

Figure 4. Word/state alignments. The segmentation of the word sequences in the shaded region is hidden. The EM algorithm is applied to train state-specific n-gram models.

We evaluate the HMM/CFG composite model in the ATIS domain [2]. The model is trained with ATIS3 category A training data (~1700 annotated sentences) and tested with the 1993 ATIS3 category A test sentences. Compared to the manually annotated labels, the test set slot ins-del-sub error rate is 5%. This leads to a 5.3% semantic error rate in the standard ATIS evaluation, which is slightly better than the best manually developed system (5.5%).

3. PORTING TO CONDITIONAL MODELS

We investigated the application of conditional models to SLU. The problem can be formulated as assigning a label l to each word in the word sequence $\mathbf{o}_1^\tau$ of observation $\mathbf{o}$. Here an observation $\mathbf{o}$ consists of a word vector $\mathbf{o}_1^\tau$ and CFG non-termimals that cover subsequences of $\mathbf{o}_1^\tau$, as illustrated in Figure 5. The task for the conditional model is to label "two" as the "NumOfTickets" slot of the "ShowFlight" command, and label "Washington D.C." as the ArriveCity slot for the same task. To do so, the model must learn to resolves the following ambiguities: the filler/non-filler ambiguity (e.g. "two" as a NumTickets slot filler vs. as part of the preamble of ArriveCity); CFG coverage ambiguity (e.g. City vs. State for "Washington"); segmentation ambiguity (e.g. [Washington][D.C.] vs. [Washington D.C.]); and semantic label ambiguity (e.g. [$_{\text{ArriveCity}}$ Washington D.C.] vs. [$_{\text{DepartCity}}$ Washington D.C.]).

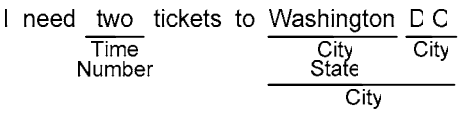

Figure 5. Observation consists of a word sequence and the subsequences covered by CFG non-terminal symbols.

3.1 CRFs and HCRFs

Conditional Random Fileds (CRFs) [4] are undirected conditional graphical models that assign the conditional probability of a state (label) sequence with respect to a vector of the features $\mathbf{f}(s_1^\tau, \mathbf{o}_1^\tau)$. They are of the following form:

$$p(s_1^\tau \mid \mathbf{o}; \lambda) = \frac{1}{z(\mathbf{o}; \lambda)} \exp\left(\lambda \cdot \mathbf{f}(s_1^\tau, \mathbf{o})\right). \quad (1)$$

The parameter vector λ is trained conditionally (discriminatively). If we assume that S_1^T is a Markov chain given $\mathbf{O}$, then

$$p(s_1^\tau \mid \mathbf{o}; \lambda) = \frac{1}{z(\mathbf{o}; \lambda)} \exp\left(\sum_k \lambda_k \sum_{t=1}^{\tau} f_k(s^{(t-1)}, s^{(t)}, \mathbf{o}, t)\right) \quad (2)$$

In some cases, it may be natural to define feature vectors that depend on variables that are not directly observed. For example, the following feature may be defined in terms of an observed words and an unobserved state in the shaded region in Figure 4:

$$f_{\text{FlightInit,flights}}(s^{(t-1)}, s^{(t)}, \mathbf{o}, t) = \begin{cases} 1 & \text{if } s^{(t)}\text{=FlightInit} \wedge \ \mathbf{o}^t = \text{flights;} \\ 0 & \text{otherwise} \end{cases}$$

In this case, the state sequence S_1^T is used in the model, but the sequence is only partially labeled in the observation as $l(S_5) = \text{"DepartCity"} \wedge l(S_8) = \text{"ArriveCity"}$ for the words "Seattle" and "Boston". The state for the remaining words are hidden variables. To obtain the conditional probability of the partially observed label, we need to sum over all possible values of the hidden variables:

$$p(l \mid \mathbf{o}; \lambda) = \frac{1}{z(\mathbf{o}; \lambda)} \sum_{s_1^\tau \in \Gamma(l)} \exp\left(\sum_k \lambda_k \sum_{t=1}^{\tau} f_k(s^{(t-1)}, s^{(t)}, \mathbf{o}, t)\right) \quad (3)$$

Here $\Gamma(l)$ represents the set of all state sequences that satisfy the constraints imposed by the observed label l. CRFs with features depending on hidden variables are called Hidden Conditional Random Fields (HCRFs). They are applied to tasks like Phonetic classification [3].

We train CRFs and HCRFs with gradient-based optimization algorithms that maximize the log conditional likelihood. The gradient of the log conditional likelihood is

$$\nabla_\lambda L(\lambda) = \mathbf{E}_{\tilde{P}_{L,\mathbf{O}} P_{S_1^T|L,\mathbf{O}}} \left[\mathbf{f}(S_1^T, \mathbf{O}); \lambda \right] - \mathbf{E}_{\tilde{P}_{\mathbf{O}} P_{S_1^T|\mathbf{O}}} \left[\mathbf{f}(S_1^T, \mathbf{O}); \lambda \right] \quad (4)$$

which is the difference between the conditional expectation of the feature vector given the observation sequence and label, and its conditional expectation given only the observation sequence. Due to the Markov assumption we made earlier in Eq. (2), these expectations can be computed using forward-backward like dynamic programming algorithm. In the results reported in this summary, we applied stochastic gradient decent (SGD) [5] for model training.

3.2 Porting HMM/CFG Model to HCRF

Our original objective of applying conditional models was to exploit their discriminative training capability. Initially, we used the same state topology and features as the one we used in the HMM/CFG composite model.

Because the state sequence is only partially labeled, a HCRF is used to model the conditional distribution of labels. The following features are included in the model:

1. Command prior features capture the likelihood of observing different top-level commands:

$$f_c^{PR}(s^{(t-1)}, s^{(t)}, \mathbf{o}, t) = \begin{cases} 1 & \text{if } t{=}0 \wedge \mathrm{C}(s^{(t)}) = c \\ 0 & \text{otherwise} \end{cases}, \forall c \in \text{CommandSet}.$$

   Here C(s) stands for the name of the top-level command that corresponding to the transition network containing s.

2. Transition features capture the likelihood of transition from one state to another:

$$f_{s_1,s_2}^{TR}(s^{(t-1)}, s^{(t)}, \mathbf{o}, t) = \begin{cases} 1 & \text{if } s^{(t-1)} = s_1, s^{(t)} = s_2 \\ 0 & \text{otherwise} \end{cases},$$

   $\forall s_1, s_2 |\ s_1 \to s_2$ is a legal trasnition in model topology.

3. Unigram and bigram features capture the words that a state emits:

$$f_{s,w}^{UG}(s^{(t-1)}, s^{(t)}, \mathbf{o}_1^\tau, t) = \begin{cases} 1 & \text{if } s^{(t)} = s \wedge \mathbf{o}^t = w \\ 0 & \text{otherwise} \end{cases},$$

$$f_{s,w_1,w_2}^{BG}(s^{(t-1)}, s^{(t)}, \mathbf{o}_1^\tau, t) = \begin{cases} 1 & \text{if } s^{(t-1)} = s \wedge s^{(t)} = s \wedge \mathbf{o}^{t-1} = w_1 \wedge \mathbf{o}^t = w_2 \\ 0 & \text{otherwise} \end{cases},$$

$$\forall s \,|\, \neg\text{isFiller}(s); \forall w, w_1 w_2 \in \text{TrainingData}$$

The model is trained with SGD with two different ways to initialize the parameters. The flat start initialization set all parameters to 0. The generative model initialization converts the parameters of the HMM/CFG composite model to the conditional model.

Figure 6 shows the test set slot error rates at different training iterations. The flat start initialization (top curve) never catches up the 5% baseline error rate of the HMM/CFG composite model. The generative initialization reduces the error rate to 4.8% at the first two iterations but quickly gets over-trained afterwards.

The failure of the direct porting of the generative model to the conditional model can be attributed to the following reasons:

1. The conditional log-likelihood function is no longer a convex function due to the summation over hidden variables. This makes it highly likely that model training will settle on a local optimum. The fact that the flat start initialization failed to catch up the accuracy of the generative initialization is a clear indication of the problem.
2. The generative model needs to account for the words in test data. For that purpose, the n-grams models are properly smoothed with backed-offs to the uniform distribution over the vocabulary. This results in a huge parameter space, and many of the parameters cannot be estimated reliably in the conditional model, given that model regularization is not as well studied as in the n-gram generation model.
3. The hidden variables also contribute to the unreliable estimate of parameters with a small amount of training data.

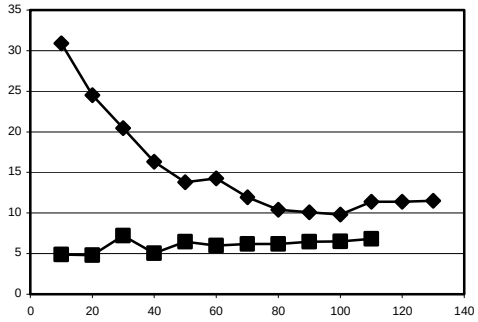

Figure 6. Test set slot error rates (in %) at different training iterations. The top curve is for the flat start initialization. The bottom curve is for the generative initialization.

3. CRFS FOR SPOKEN LANGUAGE UNDERSTANDING

An important lesson we have learned from the previous experiment is that we should not think generatively when we apply conditional models. We only need to find the important cues that help identify slots. There is no need to accurately estimate the distribution of generating every word in a sentence. Hence the separation of precommands, preambles, postcommands and postambles is no longer necessary. Every word that appears between two slots is labeled as the preamble state of the second slot, as illustrated in Figure 7. This effectively removes the hidden variables and greatly simplifies the model to a CRF. This not only improves the speed of model training, but also avoids settling at a local optimum because the log conditional likelihood is a convex function in CRF.

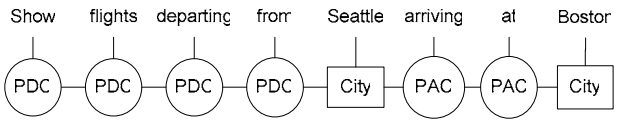

Figure 7. The state sequence is deterministic once the slots are marked in the simplified model topology. The fully marked state sequence leaves no hidden variables and results in a CRF model.

The same command prior and state transition features (with fewer states) are used as in the HCRF model. For unigram and bigram features, only the unigrams and bigrams that occur in front of a CFG non-terminal that can be the filler of a slot are included as the features for the preamble state of that slot:

$$f^{UG}_{s,w}(s^{(t-1)},s^{(t)},\mathbf{o}^{\tau}_1,t)=\begin{cases}1 & \text{if } s^{(t)}=s\wedge\mathbf{o}^{t}=w\\ 0 & \text{otherwise}\end{cases},$$

$$f^{BG}_{s,w_1,w_2}(s^{(t-1)},s^{(t)},\mathbf{o}^{\tau}_1,t)=\begin{cases}1 & \text{if } s^{(t-1)}=s^{(t)}=s\wedge\mathbf{o}^{t-1}=w_1\wedge\mathbf{o}^{t}=w_2\\ 0 & \text{otherwise}\end{cases},$$

$$\forall s\,|\,\neg\text{isFiller}(s);$$

$$\forall w,w_1w_2\,|\,\text{in the training data},w,w_1w_2 \text{ appears in front of sequence covered by a CFG rule that is the filler of the slot preambled by } s.$$

One advantage of CRFs over generative models is that we can introduce more non-independent, non-homogeneous features to the model. The first additional feature set we introduce to the model addresses a side effect of not modeling the generation of every word in a sentence. If a preamble state has never occurred in a position that is confusable with a filler of a slot, and a word in the filler has never occurred as part of the preamble, then the unigram feature of the word for that preamble has weight 0. In such case, there is not penalty for mislabeling the word as the preamble. This is one of the most common errors we observed in the development set. The chunk coverage features are introduced for the model to learn the likelihood of a word covered by a CFG non-terminals being labeled as a preamble:

$$f^{CC}_{c,NT}(s^{(t-1)},s^{(t)},\mathbf{o},t)=\begin{cases}1 & \text{if } \mathrm{C}(s^{(t)})=c\wedge\text{covers}(NT,\mathbf{o}^{t})\wedge\ \text{isPre}(s^{(t)})\\ 0 & \text{otherwise}\end{cases}$$

Here isPre(s) indicates that s is a preamble state.

In many cases the identity of a slot depends on the preambles of the slot in front of it. For examples, "at two PM" is a DepartTime in "flight from Seattle to Boston at two PM" but an ArriveTime in "flight departing from Seattle arriving in Boston at two PM." In both cases, its previous slot is ArriveCity, so the transition features will not be helpful for slot identity disambiguation. The identity of the time slot depends on the preamble of the ArriveCity slot. The previous slot's context features introduce this dependency to the model:

$$f^{PC}_{s_1,s_2,w}(s^{(t-1)},s^{(t)},\mathbf{o},t)=\begin{cases}1 & \text{if } s^{(t-1)}=s_1\wedge s^{(t)}=s_2\wedge w\in\Theta(s_1,\mathbf{o},t-1)\\ & \wedge\ \text{isFiller}(s_1)\wedge\ \text{Slot}(s_1)\neq\text{Slot}(s_2)\\ 0 & \text{otherwise}\end{cases}$$

Here the condition isFiller(s_1) restricts that s_1 is a slot filler (not a slot preamble.) Slot(s) stands for the semantic slot associated with the state s, which can be the slot's filler or its preamble. $\Theta(s_1,\mathbf{o},t-1)$ is a set that contains the two words in front of the longest sequence that ends at position $t-1$ and that is covered by the filler non-terminal for Slot(s_1).

The next set of features helps prevent the model from making mistakes like segmenting "Washington D.C." into two different cities. The slot boundary chunk coverage feature is activated when a slot boundary within a task is covered by a CFG non-terminal *NT*:

$$f^{SB}_{c,NT}(s^{(t-1)},s^{(t)},\mathbf{o},t)=\begin{cases}1 & \text{if } \mathrm{C}(s^{(t)})=c\wedge\text{covers}(NT,\mathbf{o}^{t}_{t-1})\\ & \wedge\ \text{isFiller}(s^{(t-1)})\wedge\ \text{isFiller}(s^{(t)})\wedge s^{(t-1)}\neq s^{(t)}\\ 0 & \text{otherwise}\end{cases}$$

This feature shares its weight with $f^{CC}_{t,NT}(s^{(t-1)},s^{(t)},\mathbf{o}^{\tau}_1,t)$. So no extra model parameters are introduced.

Table 1 shows the Number of new parameters and the slot ins-del-sub error rate after each new feature set is introduced into the model, taken from the training iteration that obtains the best accuracy on the development set. The inclusion of new features makes the model more accurate in predicting slot identity and reduces the error rate by 17% relatively over the generative HMM/CFG composite model.

Features	# of Parameters	Slot Error Rate
Task prior	6	
+Slot Transition	+1377	
+Unigrams	+14433	8.40%
+Bigrams	+58191	7.87%
+ChunkCoverForWords	+156	6.87%
+PrevSlotContext	+290	5.46%
+ChunkCoverSlotBoundaries	+0	4.17%

Table 1. Number of additional parameters and the slot ins-del-sub error rate after each new feature set is introduced into the model.

It is important to note that features similar to f^{CC}, f^{SB} and f^{PC} could not be easily introduced in the generative model. The capability of incorporating non-homogeneous features is the key benefit of CRFs. This is consistent with the findings in that work that used conditional model for POS tagging [Lafferty, 2001 #141].

4. DISCUSSIONS AND CONCLUSIONS

We have shown that conditional model reduces SLU slot error rate by 17% over the generative HMM/CFG composite model. The improvement was mostly due to the introduction of the new features into the model. We have also discussed about our experience in direct porting a generative model to a conditional model, and demonstrated that it may not helpful at all if we still think generatively in conditionally modeling --- more specifically, using the same feature set as a generative model in a conditional model may not help much. The key benefit that the conditional models bring is their capability of incorporating non-independent and non-homogeneous features.

REFERENCES

[1] Y.-Y. Wang, L. Deng, and A. Acero, "Spoken Language Understanding --- An Introduction to the Statistical Framework." *IEEE Signal Processing Magazine*, vol. 22, 2005.

[2] P. Price, "Evaluation of Spoken Language System: the ATIS domain." In the Proceedings of DARPA Speech and Natural Language Workshop, Hidden Valley, PA, 1990.

[3] A. Gunawardana, M. Mahajan, A. Acero, and J. C. Platt, "Hidden Conditional Random Fields for Phone Classification." In the Proceedings of Eurospeech, Lisbon, Portugal, 2005.

[4] J. Lafferty, A. McCallum, and F. Pereira, "Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data." In the Proceedings of ICML, 2001.

[5] H. J. Kushner and G. G. Yin, *Stochastic Approximation Algorithms and Applications*: Springer-Verlag, 1997.