Backoff Model Training using Partially Observed Data: Application to Dialog Act Tagging

Gang Ji and **Jeff Bilmes**
Department of Electrical Engineering
University of Washington
Seattle, WA 98105-2500
{gang,bilmes}@ee.washington.edu

Abstract

Dialog act (DA) tags are useful for many applications in natural language processing and automatic speech recognition. In this work, we introduce *hidden backoff models* (HBMs) where a large generalized backoff model is trained, using an embedded expectation-maximization (EM) procedure, on data that is partially observed. We use HBMs as word models conditioned on both DAs and (hidden) DA-segments. Experimental results on the ICSI meeting recorder dialog act corpus show that our procedure can strictly increase likelihood on training data and can effectively reduce errors on test data. In the best case, test error can be reduced by 6.1% relative to our baseline, an improvement on previously reported models that also use prosody. We also compare with our own prosody-based model, and show that our HBM is competitive even without the use of prosody. We have not yet succeeded, however, in combining the benefits of both prosody and the HBM.

1 Introduction

Discourse patterns in natural conversations and meetings are well known to provide interesting and useful information about human conversational behavior. They thus attract research from many different and beneficial perspectives. Dialog acts (DAs) (Searle, 1969), which reflect the functions that utterances serve in a discourse, are one type of such patterns. Detecting and understanding dialog act patterns can provide benefit to systems such as automatic speech recognition (ASR) (Stolcke et al., 1998), machine dialog translation (Lee et al., 1998), and general natural language processing (NLP) (Jurafsky et al., 1997b; He and Young, 2003). DA pattern recognition is an instance of "tagging." Many different techniques have been quite successful in this endeavor, including hidden Markov models (Jurafsky et al., 1997a; Stolcke et al., 1998), semantic classification trees and polygrams (Mast et al., 1996), maximum entropy models (Ang et al., 2005), and other language models (Reithinger et al., 1996; Reithinger and Klesen, 1997). Like other tagging tasks, DA recognition can also be achieved using conditional random fields (Lafferty et al., 2001; Sutton et al., 2004) and general discriminative modeling on structured outputs (Bartlett et al., 2004). In many sequential data analysis tasks (speech, language, or DNA sequence analysis), standard dynamic Bayesian networks (DBNs) (Murphy, 2002) have shown great flexibility and are widely used. In (Ji and Bilmes, 2005), for example, an analysis of DA tagging using DBNs is performed, where the models avoid label bias by structural changes and avoid data sparseness by using a generalized backoff procedures (Bilmes and Kirchhoff, 2003).

Most DA classification procedures assume that within a sentence of a particular fixed DA type, there is a fixed word distribution over the entire sentence. Similar to (Ma et al., 2000) (and see citations therein), we have found, however, that intra-

Proceedings of the Human Language Technology Conference of the North American Chapter of the ACL, pages 280–287,
New York, June 2006.

sentence discourse patterns are inherently dynamic. Moreover, the patterns are specific to each type of DA, meaning a sentence will go through a DA-specific sequence of sub-DA phases or "states." A generative description of this phenomena is that a DA is first chosen, and then words are generated according to both the DA and to the relative position of the word in that sentence. For example, a "statement" (one type of DA) can consist of a subject (noun phrase), verb phrase, and object (noun phrase). This particular sequence might be different for a different DA (e.g., a "back-channel"). Our belief is that explicitly modeling these internal states can help a DA-classification system in conversational meetings or dialogs.

In this work, we describe an approach that is motivated by several aspects of the typical DA-classification procedure. First, it is rare to have sub-DAs labeled in training data, and indeed this is true of the corpus (Shriberg et al., 2004) that we use. Therefore, some form of unsupervised clustering or pre-shallow-parsing of sub-DAs must be performed. In such a model, these sub-DAs are essentially unknown hidden variables that ideally could be trained with an expectation-maximization (EM) procedure. Second, when training models of language, it is necessary to employ some form of smoothing methodology since otherwise data-sparseness would render standard maximum-likelihood trained models useless. Third, discrete conditional probability distributions formed using backoff models that have been smoothed (particularly using modified Kneser-Ney (Chen and Goodman, 1998)) have been extremely successful in many language modeling tasks. Training backoff models, however, requires that all data is observed so that data counts can be formed. Indeed, our DA-specific word models (implemented via backoff) will also need to condition on the current sub-DA, which at training time is unknown. We therefore have developed a procedure that allows us to train generalized backoff models (Bilmes and Kirchhoff, 2003), even when some or all of the variables involved in the model are *hidden*. We thus call our models *hidden backoff model*s (HBMs). Our method is indeed a form of embedded EM training (Morgan and Bourlard, 1990), and more generally is a specific form of EM (Neal and Hinton, 1998). Our approach is similar to (Ma et al., 2000), except our underlying language models are backoff-based and thus retain the benefits of advanced smoothing methods, and we utilize both a normal and a backoff EM step as will be seen. We moreover wrap up the above ideas in the framework of dynamic Bayesian networks, which are used to represent and train all of our models.

We evaluate our methods on the ICSI meeting recorder dialog act (MRDA) (Shriberg et al., 2004) corpus, and find that our novel hidden backoff model can significantly improve dialog tagging accuracy. With a different number of hidden states for each DA, a relative reduction in tagging error rate as much as 6.1% can be achieved. Our best HBM result shows an accuracy that improves on the best known (to our knowledge) result on this corpora which is one that uses acoustic prosody as a feature. We have moreover developed our own prosody model and while we have not been able to usefully employ both prosody and the HBM technique together, our HBM is competitive in this case as well. Furthermore, our results show the effectiveness of our embedded EM procedure, as we demonstrate that it increases training log likelihoods, while simultaneously reducing error rate.

Section 2 briefly summarizes our baseline DBN-based models for DA tagging tasks. In Section 3, we introduce our HBMs. Section 4 contains experimental evaluations on the MRDA corpus and finally Section 5 concludes.

2 DBN-based Models for Tagging

Dynamic Bayesian networks (DBNs) (Murphy, 2002) are widely used in sequential data analysis such as automatic speech recognition (ASR) and DNA sequencing analysis (Durbin et al., 1999). A hidden Markov model (HMM) for DA tagging as in (Stolcke et al., 1998) is one such instance.

Figure 1 shows a generative DBN model that will be taken as our baseline. This DBN shows a prologue (the first time slice of the model), an epilogue (the last slice), and a chunk that is repeated sufficiently to fit the entire data stream. In this case, the data stream consists of the words of a meeting conversation, where individuals within the meeting (hopefully) take turns speaking. In our model, the entire meeting conversation, and all turns of all

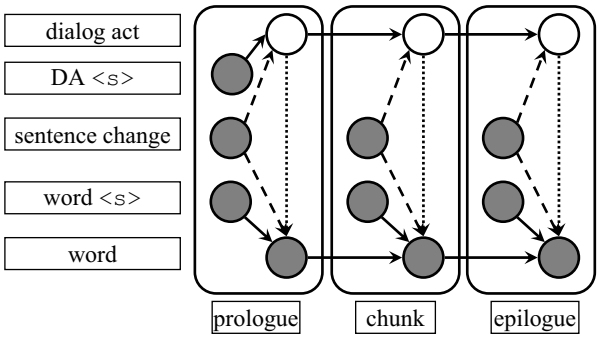

Figure 1: Baseline generative DBN for DA tagging.

speakers, are strung together into a single stream rather than treating each turn in the meeting individually. This approach has the benefit that we are able to integrate a temporal DA-to-DA model (such as a DA bigram).

In all our models, to simplify we assume that the sentence change information is known (as is common with this corpus (Shriberg et al., 2004)). We next describe Figure 1 in detail. Normally, the *sentence change* variable is not set, so that we are within a sentence (or a particular DA). When a sentence change does not occur, the DA stays the same from slice to slice. During this time, we use a DA-specific language model (implemented via a backoff strategy) to score the words within the current DA.

When a sentence change event does occur, a new DA is predicted based on the DA from the previous sentence (using a DA bigram). At the beginning of a sentence, rather than conditioning on the last word of the previous sentence, we condition on the special start of sentence <s> token, as shown in the figure by having a special parent that is used only when *sentence change* is true. Lastly, at the very beginning of a meeting, a special start of DA token is used.

The joint probability under this baseline model is written as follows:

$$P(W, D) = \prod_k P(d_k|d_{k-1}) \cdot \prod_i P(w_{k,i}|w_{k,i-1}, d_k), \quad (1)$$

where $W = \{w_{k,i}\}$ is the word sequence, $D = \{d_k\}$ is the DA sequence, d_k is the DA of the k-th sentence, and $w_{k,i}$ is the i-th word of the k-th sentence in the meeting.

Because all variables are observed when training our baseline, we use the SRILM toolkit (Stolcke, 2002), modified Kneser-Ney smoothing (Chen and Goodman, 1998), and factored extensions (Bilmes and Kirchhoff, 2003). In evaluations, the Viterbi algorithm (Viterbi, 1967) can be used to find the best DA sequence path from the words of the meeting according to the joint distribution in Equation (1).

3 Hidden Backoff Models

When analyzing discourse patterns, it can be seen that sentences with different DAs usually have different internal structures. Accordingly, in this work we do not assume sentences for each dialog act have the same hidden state patterns. For instance (and as mentioned above), a statement can consist of a noun followed by a verb phase.

A problem, however, is that sub-DAs are not annotated in our training corpus. While clustering and annotation of these phrases is already a widely developed research topic (Pieraccini and Levin, 1991; Lee et al., 1997; Gildea and Jurafsky, 2002), in our approach we use an EM algorithm to learn these hidden sub-DAs in a data-driven fashion. Pictorially, we add a layer of hidden states to our baseline DBN as illustrated in Figure 2.

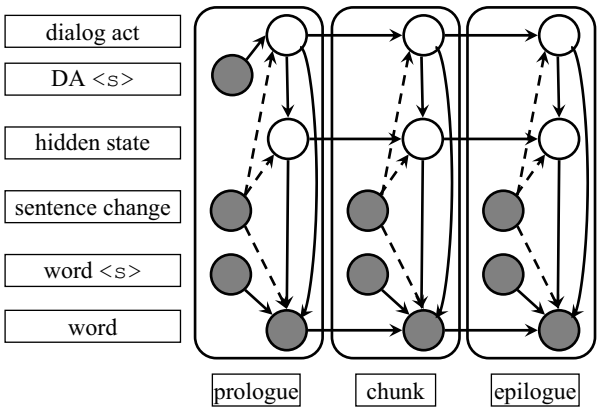

Figure 2: Hidden backoff model for DA tagging.

Under this model, the joint probability is:

$$P(W, S, D) = \prod_k P(d_k|d_{k-1}) \cdot \prod_i [P(s_{k,i}|s_{k,i-1}, d_k) \cdot P(w_{k,i}|w_{k,i-1}, s_{k,i}, d_k)], \quad (2)$$

where $S = \{s_{k,i}\}$ is the hidden state sequence, $s_{k,i}$ is the hidden state at the i-th position of the k-th sentence, and other variables are the same as before.

Similar to our baseline model, the DA bigram $P(d_k|d_{k-1})$ can be modeled using a backoff bigram. Moreover, if the states $\{s_{k,i}\}$ are known during training, the word prediction probability $P(w_{k,i}|w_{k,i-1}, s_{k,i}, d_k)$ can also use backoff and be trained accordingly. The hidden state sequence is unknown, however, and thus cannot be used to produce a standard backoff model. What we desire is an ability to utilize a backoff model (to mitigate data sparseness effects) while simultaneously retaining the state as a hidden (rather than an observed) variable, and also have a procedure that trains the entire model to improve overall model likelihood.

Expectation-maximization (EM) algorithms are well-known to be able to train models with hidden states. Furthermore, standard advanced smoothing methods such as modified Kneser-Ney smoothing (Chen and Goodman, 1998) utilize integer counts (rather than fractional ones), and they moreover need "meta" counts (or counts of counts). Therefore, in order to train this model, we propose an embedded training algorithm that cycles between a standard EM training procedure (to train the hidden state distribution), and a stage where the most likely hidden states (and their counts and meta counts) are used externally to train a backoff model. This procedure can be described in detail as follows:

Input : W — meeting word sequence
Input : D — DA sequence
Output : $P(s_{k,i}|s_{k,i-1})$ - state transition CPT
Output : $P(w_{k,i}|w_{k,i-1}, s_{k,i}, d_k)$ - word model

randomly generate a sequence S;
backoff train $P(w_{k,i}|w_{k,i-1}, s_{k,i}, d_k)$;
while *not "converged"* **do**
  EM train $P(s_{k,i}|s_{k,i-1})$;
  calculate best $\bar{S}$ sequence by Viterbi;
  backoff train $P(w_{k,i}|w_{k,i-1}, \bar{s}_{k,i}, d_k)$;
end

Algorithm 1: Embedded training for HBMs

In the algorithm, the input contains words and a DA for each sentence in the meeting. The output is the corresponding conditional probability table (CPT) for hidden state transitions, and a backoff model for word prediction. Because we train the backoff model when some of the variables are hidden, we call the result a *hidden backoff model*. While we have seen embedded Viterbi training used in the past for simultaneously training heterogeneous models (e.g., Markov chains and Neural Networks (Morgan and Bourlard, 1990)), this is the first instance of training backoff-models that involve hidden variables that we are aware of.

While embedded Viterbi estimation is not guaranteed to have the same convergence (or fixed-point under convergence) as normal EM (Lember and Koloydenko, 2004), we find empirically this to be the case (see examples below). Moreover, our algorithm can easily be modified so that instead of taking a Viterbi alignment in step 5, we instead use a set of random samples generated under the current model. In this case, it can be shown using a law-of-large numbers argument that having sufficient samples guarantees the algorithm will converge (we will investigate this modification in future work).

Of course, when decoding with such a model, a conventional Viterbi algorithm can still be used to calculate the best DA sequence.

4 Experimental Results

We evaluated our hidden backoff model on the ICSI meeting recorder dialog act (MRDA) corpus (Shriberg et al., 2004). MRDA is a rich data set that contains 75 natural meetings on different topics with each meeting involving about 6 participants. DA annotations from ICSI were based on a previous approach in (Jurafsky et al., 1997b) with some adaptation for meetings in a number of ways described in (Bhagat et al., 2003). Each DA contains a main tag, several optional special tags and an optional "disruption" form. The total number of distinct DAs in the corpus is as large as 1260. In order to make the problem comparable to other work (Ang et al., 2005), a DA tag sub-set is used in our experiments that contains back channels (b), place holders (h), questions (q), statements (s), and disruptions (x). In our evaluations, among the entire 75 conversations, 51 are used as the training set, 11 are used as the development set, 11 are used as test set, and the remaining 3 are not used. For each experiment, we used a genetic algorithm to search for the best factored language model structure on the development set and

we report the best results.

Our baseline system is the generative model shown in Figure 1 and uses a backoff implementation of the word model, and is optimized on the development set. We use the SRILM toolkit with extensions (Bilmes and Kirchhoff, 2003) to train, and use GMTK (Bilmes and Zweig, 2002) for decoding. Our baseline system has an error rate of 19.7% on the test set, which is comparable to other approaches on the same task (Ang et al., 2005).

4.1 Same number of states for all DAs

To compare against our baseline, we use HBMs in the model shown in Figure 2. To train, we followed Algorithm 1 as described before and as is here detailed in Figure 3.

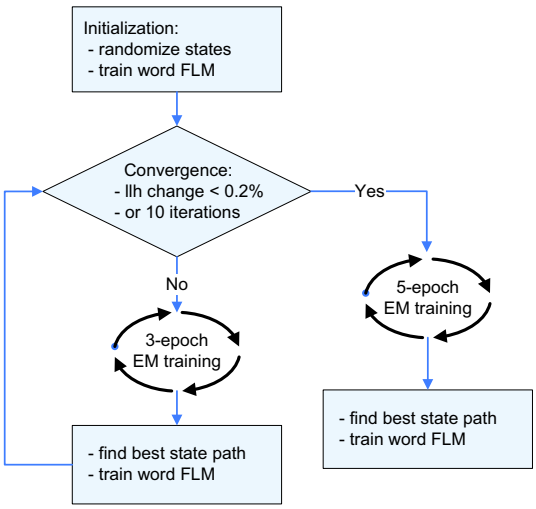

Figure 3: Embedded training: llh = log likelihood

In this implementation, an upper triangular matrix (with self-transitions along the diagonal) is used for the hidden state transition probability table so that sub-DA states only propagate in one direction. When initializing the hidden state sequence of a DA, we expanded the states uniformly along the sentence. This initial alignment is then used for HBM training. In the word models used in our experiments, the backoff path first drops previous words, then does a parallel backoff to hidden state and DA using a mean combination strategy.

The HBM thus obtained was then fed into the main loop of our embedded EM algorithm. The training was considered to have "converged" if either it exceeded 10 iterations (which never happened) or the relative log likelihood change was less than 0.2%. Within each embedded iteration, three EM epochs were used. After each EM iteration, a Viterbi alignment was performed thus obtaining what we expect to be a better hidden state alignment. This updated alignment, was then used to train a new HBM. The newly generated model was then fed back into the embedded training loop until it converged. After the procedure met our convergence criteria, an additional five EM epochs were carried out in order to provide a good hidden state transition probability table. Finally, after Viterbi alignment and text generation was performed, the word HBM was trained from the best state sequence.

To evaluate our hidden backoff model, the Viterbi algorithm was used to find the best DA sequence according to test data, and the tagging error rates were calculated. In our first experiment, an equal number of hidden states for all DAs were used in each model. The effect of this number on the accuracy of DA tagging is shown in Table 1.

Table 1: HBMs, different numbers of hidden states.

# states	error	improvement
baseline	19.7%	–
2-state	18.7%	5.1%
3-state	19.5%	1.0%

For the baseline system, the backoff path first drops `dialog act`, and for the HBMs, all backoff paths drop `hidden state` first and drop `DA` second. From Table 1 we see that with two hidden states for every DA the system can reduce the tagging error rate by more than 5% relative. As a comparison, in (Ang et al., 2005), where conditional maximum entropy models (which are conditionally trained) are used, the error rate is 18.8% when using both word and acoustic prosody features, and and 20.5% without prosody. When the number of hidden states increases to 3, the improvement decreases even though it is still (very slightly) better than the baseline. We believe the reasons are as follows: First, assuming different DAs have the same number of hidden states may not be appropriate. For example, back channels usually have shorter sentences and are constant in discourse pattern over a DA. On the other hand,

questions and statements typically have longer, and more complex, discourse structures. Second, even under the same DA, the structure and inherent length of sentence can vary. For example, "yes" can also be a statement even though it has only one word. Therefore, one-word statements need completely different hidden state patterns than those in subject-verb-object like statements — having one monolithic 3-state model for statements might be inappropriate. This issue is discussed further in Section 4.4.

4.2 Different states for different DAs

In order to mitigate the first problem described above, we allow different numbers of hidden states for each DA. This, however, leads to a combinatorial explosion of possibilities if done in a naïve fashion. Therefore, we attempted only a small number of combinations based on the statistics of numbers of words in each DA given in Table 2.

Table 2: Length statistics of different DAs.

DA	mean	median	std	p
(b)	1.0423	1	0.2361	0.4954
(h)	1.3145	1	0.7759	0.4660
(q)	6.5032	5	6.3323	0.3377
(s)	8.6011	7	7.8380	0.3013
(x)	1.7201	1	1.1308	0.4257

Table 2 shows the mean and median number of words per sentence for each DA as well as the standard deviation. Also, the last column provides the p value according to fitting the length histogram to a geometric distribution $(1-p)^n p$. As we expected, back channels (b) and place holders (h) tend to have shorter sentences while questions (q) and statements (s) have longer ones. From this analysis, we use fewer states for (b) and (h) and more states for (q) and (s). For disruptions (x), the standard deviation of number of words histogram is relatively high compared with (b) and (h), so we also used more hidden states in this case. In our experimental results below, we used one state for (b) and (h), and various numbers of hidden states for other DAs. Tagging error rates are shown in Table 3.

From Table 3, we see that using different numbers of hidden states for different DAs can produce better models. Among all the experiments we performed, the best case is given by three states for (q), two states for (s) and (x), and one state for (b) and (h). This combination gives 6.1% relative reduction of error rate from the baseline.

Table 3: Number of hidden states for different DAs.

b	h	q	s	x	error	improvement
1	1	4	4	1	18.9%	4.1%
1	1	3	3	2	18.9%	4.1%
1	1	2	2	2	18.7%	5.1%
1	1	3	2	2	18.6%	5.6%
1	1	3	2	2	18.5%	6.1%

4.3 Effect of embedded EM training

Incorporating backoff smoothing procedures into Bayesian networks (and hidden variable training in particular) can show benefits for any data domain where smoothing is necessary. To understand the properties of our algorithm a bit better, after each training iteration using a partially trained model, we calculated both the log likelihood of the training set and the tagging error rate of the test data. Figure 4 shows these results using the best configuration from the previous section (three states for (q), two for (s)/(x) and one for (b)/(h)). This example is typical of the convergence we see of Algorithm 1, which empirically suggests that our procedure may be similar to a generalized EM (Neal and Hinton, 1998).

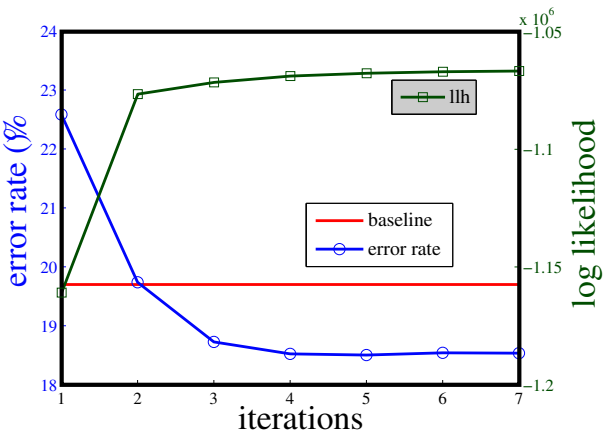

Figure 4: Embedded EM training performance.

We find that the log likelihood after each EM training is strictly increasing, suggesting that our embedded EM algorithm for hidden backoff models

is improving the overall joint likelihood of the training data according to the model. This strict increase of likelihood combined with the fact that Viterbi training does not have the same theoretical convergence guarantees as does normal EM indicates that more detailed theoretical analysis of this algorithm used with these particular models is desirable.

From the figure we also see that both the log likelihood and tagging error rate "converge" after around four iterations of embedded training. This quick convergence indicates that our embedded training procedure is effective. The leveling of the error rates after several iterations shows that model over-fitting appears not to be an issue presumably due to the smoothed embedded backoff models.

4.4 Discussion and Error Analysis

A large portion of our tagging errors are due to confusing the DA of short sentences such as "yeah", and "right". The sentence, "yeah" can either be a back channel or an affirmative statement. There are also cases where "yeah?" is a question. These types of confusions are difficult to remove in the prosody-less framework but there are several possibilities. First, we can allow the use of a "fork and join" transition matrix, where we fork to each DA-specific condition (e.g., short or long) and join thereafter. Alternatively, hidden Markov chain structuring algorithms or context (i.e., conditioning the number of sub-DAs on the previous DA) might be helpful.

Finding a proper number of hidden states for each DA is also challenging. In our preliminary work, we simply explored different combinations using simple statistics of the data. A systematic procedure would be more beneficial. In this work, we also did not perform any hidden state tying within different DAs. In practice, some states in statements should be able to be beneficially tied with other states within questions. Our results show that having three states for all DAs is not as good as two states for all. But with tying, more states might be more successfully used.

4.5 Influence of Prosody Cues

It has been shown that prosody cues provide useful information in DA tagging tasks (Shriberg et al., 1998; Ang et al., 2005). We also incorporated prosody features in our models. We used ESPS `get_f0` based on RAPT algorithm (Talkin, 1995) to get F_0 values. For each speaker, mean and variance normalization is performed. For each word, a linear regression is carried on the normalized F_0 values. We quantize the slope values into 20 bins and treat those as prosody features associated with each word. After adding the prosody features, the simple generative model as shown in Figure 5 gives 18.4% error rate, which is 6.6% improvement over our baseline. There is no statistical difference between the best performance of this prosody model and the earlier best HBM. This implies that the HBM can obtain as good performance as a prosody-based model but without using prosody.

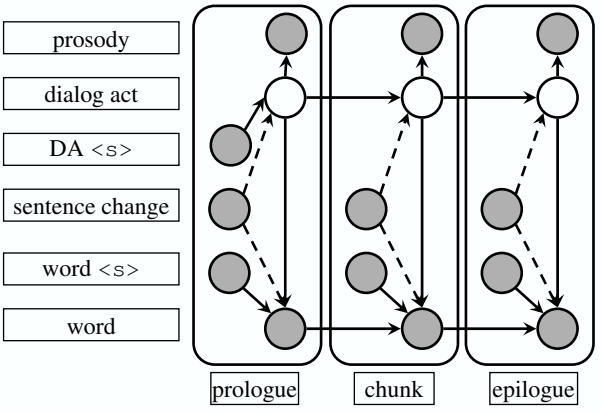

Figure 5: Generative prosody model for DA tagging.

The next obvious step is to combine an HBM with the prosody information. Strangely, even after experimenting with many different models (including ones where prosody depends on DA; prosody depends on DA and the hidden state; prosody depends on DA, hidden state, and word; and many variations thereof), we were unsuccessful in obtaining a complementary benefit when using both prosody and an HBM. One hypothesis is that our prosody features are at the word-level (rather than at the DA level). Another problem might be the small size of the MRDA corpus relative to the model complexity. Yet a third hypothesis is that the errors corrected by both methods are the same — indeed, we have verified that the corrected errors overlap by more than 50%. We plan further investigations in future work.

5 Conclusions

In this work, we introduced a training method for *hidden backoff models* (HBMs) to solve a problem in DA tagging where smoothed backoff models involving training-time hidden variables are useful. We tested this procedure in the context of dynamic Bayesian networks. Different hidden states were used to model different positions in a DA. According to empirical evaluations, our embedded EM algorithm effectively increases log likelihood on training data and reduces DA tagging error rate on test data. If different numbers of hidden states are used for different DAs, we find that our prosody-independent HBM reduces the tagging error rate by 6.1% relative to the baseline, a result that improves upon previously reported work that uses prosody, and that is comparable to our own new result that also incorporates prosody. We have not yet been able to combine the benefits of both an HBM and prosody information. This material is based upon work supported by the National Science Foundation under Grant No. IIS-0121396.

References

J. Ang et al. 2005. Automatic dialog act segmentation and classification in multiparty meetings. In *ICASSP*.

P. Bartlett et al. 2004. Exponentiated gradient algorithms for large-margin structured classification. In *NIPS*.

S. Bhagat et al. 2003. Labeling guide for dialog act tags in the meeting recordering meetings. Technical Report 2, International Computer Science Insititute.

J. Bilmes and K. Kirchhoff. 2003. Factored language models and generalized parallel backoff. In *Human Lang. Tech., North American Chapter of Asssoc. Comp. Ling.*, Edmonton, Alberta, May/June.

J. Bilmes and G. Zweig. 2002. The Graphical Models Toolkit: An open source software system for speech and time-series processing. *Proc. IEEE Intl. Conf. on Acoustics, Speech, and Signal Processing*.

S. Chen and J. Goodman. 1998. An empirical study of smoothing techniques for language modeling. Technical report, Computer Science Group, Harvard University.

R. Durbin et al. 1999. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge University Press.

D. Gildea and D. Jurafsky. 2002. Automatic labeling of semantic roles. *Computational Linguistics*, 28(3):245–288.

Y. He and S. Young. 2003. A data-driven spoken language understanding system. In *Proc. IEEE Workshop on Automatic Speech Recognition and Understanding*, pages 583–588.

G. Ji and J. Bilmes. 2005. Dialog act tagging using graphical models. In *Proc. IEEE Intl. Conf. on Acoustics, Speech, and Signal Processing*, Philadelphia, PA, March.

D. Jurafsky et al. 1997a. Automatic detection of discourse structure for speech recognition and understanding. In *Proc. IEEE Workshop on Speech Recognition and Understanding*.

D. Jurafsky et al. 1997b. Switchboard SWBD-DAMSL shallow-discourse-function annotation coders manual. Technical Report 97-02, Institute of Cognitive Science, University of Colorado.

J. Lafferty et al. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*.

K. Lee et al. 1997. Restricted representation of phrase structure grammar for building a tree annotated corpus of Korean. *Natural Language Engineering*, 3(2-3):215–230.

H. Lee et al. 1998. Speech act analysis model of Korean utterances for automatic dialog translation. *J. KISS(B) (Software and Applications)*, 25(10):1443–1452.

J. Lember and A. Koloydenko. 2004. Adjusted viterbi training. a proof of concept. *In Submission*.

K. Ma et al. 2000. Bi-modal sentence structure for language modeling. *Speech Communication*, 31(1):51–67.

M. Mast et al. 1996. Automatic classification of dialog acts with semantic classification trees and polygrams. *Connectionist, Statistical and Symbolic Approaches to Learning for Natural Language Processing*, pages 217–229.

N. Morgan and H. Bourlard. 1990. Continuous speech recognition using multilayer perceptrons with hidden Markov models. In *ICASSP*, pages 413–416.

K. Murphy. 2002. *Dynamic Bayesian Networks, Representation, Inference, and Learning*. Ph.D. thesis, MIT, Dept. Computer Science.

R. Neal and G. Hinton. 1998. A view of the EM algorithm that justifies incremental, sparse, and other variants. In *Learning in Graphical Models*, pages 355–368. Dordrecht: Kluwer Academic Publishers.

R. Pieraccini and E. Levin. 1991. Stochastic representation of semantic structure for speech understanding. In *Eurospeech*, volume 2, pages 383–386.

N. Reithinger and M. Klesen. 1997. Dialogue act classification using language models. In *Eurospeech*.

N. Reithinger et al. 1996. Predicting dialogue acts for a speech-to-speech translation system. In *ICLSP*, pages 654–657.

J. Searle. 1969. *Speech Acts: An Essay in the Philosophy of Language*. Cambridge University Press.

E. Shriberg et al. 1998. Can prosody aid the automatic classification of dialog acts in conversational speech? *Language and Speech*, 41(3–4):439–487.

E. Shriberg et al. 2004. The ICSI meeting recorder dialog act (MRDA) corpus. In *Proc. of the 5th SIGdial Workshop on Discourse and Dialogue*, pages 97–100.

A. Stolcke et al. 1998. Dialog act modeling for conversational speech. In *Proc. AAAI Spring Symp. on Appl. Machine Learning to Discourse Processing*, pages 98–105.

A. Stolcke. 2002. SRILM – an extensible language modeling toolkit. In *ICLSP*, volume 2, pages 901–904.

C. Sutton et al. 2004. Dynamic conditional random fields: factorized probabilistic models for labeling and segmenting sequence data. In *ICML*.

D. Talkin. 1995. A robust algorithm for pitch tracking (rapt). In W. B. Kleijn and K.K. Paliwal, editors, *Speech Coding and Synthesis*, pages 495–518. Elsevier Science.

A. Viterbi. 1967. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Trans. on Information Theory*, 13(2):260–269.