LETTER

Dynamic Bayesian Network Inversion for Robust Speech Recognition

Lei XIE[†a)], ***Nonmember and*** **Hongwu YANG**[††], ***Member***

SUMMARY This paper presents an inversion algorithm for dynamic Bayesian networks towards robust speech recognition, namely DBNI, which is a generalization of hidden Markov model inversion (HMMI). As a dual procedure of expectation maximization (EM)-based model reestimation, DBNI finds the 'uncontaminated' speech by moving the input noisy speech to the Gaussian means under the maximum likelihood (ML) sense given the DBN models trained on clean speech. This algorithm can provide both the expressive advantage from DBN and the noise-removal feature from model inversion. Experiments on the Aurora 2.0 database show that the hidden feature model (a typical DBN for speech recognition) with the DBNI algorithm achieves superior performance in terms of word error rate reduction.

key words: *speech recognition, hidden Markov model, dynamic Bayesian network*

1. Introduction

Recently, dynamic Bayesian networks (DBNs) [1] have gained much attention in speech recognition [2] due to their great expressive power and improved capability in inference and learning. Although hidden Markov models (HMMs) have been widely established in speech recognition systems, they have inherent expressive drawbacks in modeling real-world speech phenomena, such as gender and age differences, pronunciation variability, and channel variability. In speech recognition, the widely used HMMs possess a left-to-right structure, allowing only one hidden state in each time frame. Previous attempts have been made to model the complex speech phenomena indirectly by front-end processing and model adaptations. In contrast, DBNs have the ability to express arbitrary sets of variables in each time frame and to interpret causality between variables. They offer a highly systematic and unified means to model the details of human speech.

As a dual procedure to the Baum-Welch HMM *reestimation* algorithm, hidden Markov model *inversion* (HMMI) algorithm [3] has shown superior performance in robust speech recognition [3] and other speech applications [4]. The inversion of an HMM seeks the latent 'clean' speech that optimizes the maximum likelihood (ML) criterion given contaminated noisy speech input and model parameters trained with clean speech. Consequently, the output 'clean' speech is fed into a speech recognizer. The merit of HMMI is *mean-approaching*, i.e., moving the contaminated speech to Gaussian means trained by clean speech, while still retaining the original data distribution [3].

This paper generalizes the *inversion* algorithm (previously proposed for HMM) to the typical DBN topology in speech recognition, namely DBNI, in order to benefit speech recognition from the merits of both *expressive* ability of DBN and *noise-removal* feature of model inversion. Different from HMMI that realizes inversions on a simple model structure, we derive the inversion algorithm on complicated DBN structures that may describe conditional relations among both intra- and inter-time-frames. The effectiveness of the DBNI algorithm is demonstrated on the Aurora 2.0 database [5].

2. DBNs for Speech Recognition

A Bayesian network (BN) specifies a set of variables and a factorization of the joint probability distribution (JPD) among the variable set [2]. The network is represented by a directed acyclic graph (DAG) $\mathcal{G} = \langle X_{1:M}, \mathcal{E} \rangle$, where the nodes $X_{1:M}$ specify a set of M random variables and the edges $\mathcal{E}$ induce a factorization of the joint probability. Specifically, we denote the random variables by capital letters, and their specific values by lowercase letters. Each random variable X_i has an associated conditional probability distribution (CPD) $P(X_i|pa(X_i))$, where $pa(X_i)$ denotes the parents of X_i in $\mathcal{G}$. The probability of a joint assignment of variable values can be expressed in a factored way as

$$P(x_{1:M}) = \prod_{i=1}^{M} P(x_i|pa(x_i)). \tag{1}$$

where $pa(x_i)$ denotes the value set of the parents of X_i. According to the Markov properties, conditioned on its parents, a variable is independent of all the other variables except its descendants.

Dynamic Bayesian networks (DBNs) extend the BN framework by representing multiple collections of random variables as they evolve over time. The HMM commonly used in speech recognition is just a special, simple case of DBNs. Figure 1 illustrates a DBN model recently used in speech recognition, which describes complicated intra- and inter-frame dependencies between nodes. This model is also known as the *hidden feature model* (HFM) [6]. There

Manuscript received October 10, 2006.
Manuscript revised January 12, 2007.

[†]The author is with the School of Computer Science, Northwestern Polytechnical University, China.
[††]The author is with the Dept. of Computer Science & Technology, Tsinghua University, and the College of Physics and Electronics Engineer, Northwest Normal University, China.
a) E-mail: xielei21st@gmail.com

DOI: 10.1093/ietisy/e90–d.7.1117

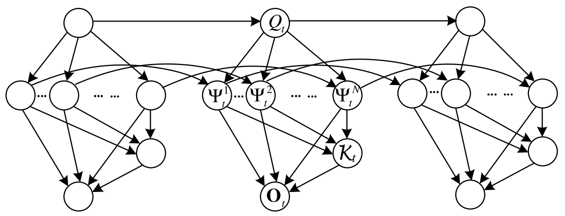

Fig. 1 A typical DBN (HFM) in speech recognition.

are four layers of variables in each time frame: the hidden state variable Q_t, the hidden feature variable Ψ_t, the Gaussian mixture variable $\mathcal{K}_t$ and the observation variable $\mathbf{O}_t$. In each frame, there are N features $\Psi_t^1, \cdots, \Psi_t^N$, each of which depends on the current phonetic state Q_t and on its own value in the previous frame. These features can evolve asynchronously and do not need to line up to form rigid phonetic segments. This reflects the suggestions from nonlinear phonology that speech is considered to be the output of multiple tiers.

3. Dynamic Bayesian Network Inversion

3.1 Inversion Algorithm

Since the inversion algorithm of HMMs has shown promising performance in robust speech recognition [3] and DBNs have superior expressive ability over HMMs, it is worth to derive the generalized inversion algorithm for DBNs and demonstrates its performance in speech recognition.

Figure 2 shows the typical DBN topology in speech recognition, which is composed of four sets of variables, i.e.,$\mathbf{O} = \{\mathbf{O}_1, \cdots, \mathbf{O}_t, \cdots, \mathbf{O}_T\}$ – observation variables; $\mathcal{K} = \{\mathcal{K}_1, \cdots, \mathcal{K}_t, \cdots, \mathcal{K}_T\}$– mixture variables; $\Psi = \{\Psi_1, \cdots, \Psi_t, \cdots, \Psi_T\}$ – hidden parents of observations, where $\Psi_t = \Psi_t^i, i = 1, \cdots, N$; $Q = \{Q_1, \cdots, Q_t, \cdots, Q_T\}$– other hidden variables. Many popular DBNs in speech recognition including the HFM in Fig. 1, can be generalized to this model topology.

According to the Baum-Welch algorithm, optimal data observations $\mathbf{O}^*$ can be found by iteratively maximizing the auxiliary function $Q(\lambda, \lambda; \mathbf{O}, \mathbf{O}')$, i.e.,

$$\mathbf{O}^* = \arg\max_{\mathbf{O}' \in O} Q(\lambda, \lambda; \mathbf{O}, \mathbf{O}'), \tag{2}$$

where $\mathbf{O}$ and $\mathbf{O}'$ denote the old and new observation sequences in the observation space O, respectively.

As described in EM [7], the *incomplete-data* likelihood function is given by $P(\mathbf{O}|\lambda)$, whereas the *complete-data* likelihood function is $P(\mathbf{O}, Q, \Psi, \mathcal{K}|\lambda)$. Given Q, Ψ, $\mathcal{K}$, and a trained model parameter set λ, according to the Markov property of independent relationships between variables, the likelihood function of the complete-data can be formed as

$$P(\mathbf{O}, Q, \Psi, \mathcal{K}|\lambda) = \prod_{t=1}^{T} \left[P(\mathbf{O}_t|\Psi_t, \mathcal{K}_t) \prod_{i=1}^{D} P(X_t^i|pa(X_t^i)) \right], \tag{3}$$

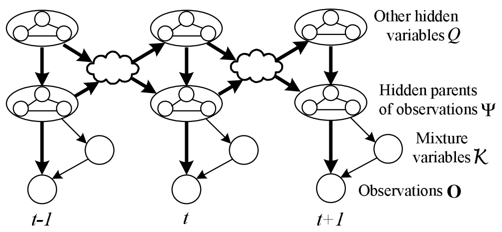
Other hidden variables Q
Hidden parents of observations Ψ
Mixture variables $\mathcal{K}$
Observations $\mathbf{O}$
t-1
t
t+1

Fig. 2 Typical DBN topology in speech recognition. An elliptical node denotes a group of variables, while a round node denotes a single variable. A wide arc represents multiple dependencies between nodes, and a slim arc represents a single dependency between nodes. The cloud between frames sketches the time dependencies among nodes in two consecutive frames.

where X_t^i denotes a hidden variable in frame t, and $pa(X_t^i)$ its parents. D is the number of hidden variables per frame. The auxiliary function is as follows:

$$\begin{aligned} & Q(\lambda, \lambda; \mathbf{O}, \mathbf{O}') \\ &= \sum_{\mathbf{Q}} \sum_{\Psi} \sum_{\mathcal{K}} P(\mathbf{O}, \mathbf{Q}, \Psi, \mathcal{K}|\lambda) \log P(\mathbf{O}', \mathbf{Q}, \Psi, \mathcal{K}|\lambda) \\ &= \sum_{\mathbf{Q}} \sum_{\Psi} \sum_{\mathcal{K}} P(\mathbf{O}, \mathbf{Q}, \Psi, \mathcal{K}|\lambda) \cdot \\ & \left\{ \sum_{t=1}^{T} \sum_{i=1}^{D} \log P(X_t^i|pa(X_t^i)) + \sum_{t=1}^{T} \log P(\mathbf{O}_t'|\Psi_t, \mathcal{K}_t) \right\} \end{aligned} \tag{4}$$

where Ψ, $\mathbf{Q}$ and $\mathcal{K}$ denote the possible sequences of $\mathbf{O}$'s hidden parents, possible sequences of other hidden variables and possible mixture segmentation sequences, respectively. By taking the derivative of $Q(\lambda, \lambda; \mathbf{O}, \mathbf{O}')$ with respect to $\mathbf{O}_t'$ to zero, we get

$$\begin{aligned} & \frac{\partial Q(\lambda, \lambda; \mathbf{O}, \mathbf{O}')}{\partial \mathbf{O}_t'} \\ &= \frac{\partial}{\partial \mathbf{O}_t'} \left[\sum_{\mathbf{Q}} \sum_{\Psi} \sum_{\mathcal{K}} P(\mathbf{O}, \mathbf{Q}, \Psi, \mathcal{K}|\lambda) \sum_{t=1}^{T} \log P(\mathbf{O}_t'|\Psi_t, \mathcal{K}_t) \right] \\ &= \sum_{\mathbf{Q}} \sum_{\Psi} \sum_{\mathcal{K}} P(\mathbf{O}, \mathbf{Q}, \Psi, \mathcal{K}|\lambda) \frac{\partial}{\partial \mathbf{O}_t'} [\log P(\mathbf{O}_t'|\Psi_t, \mathcal{K}_t)] \\ &= \sum_{\mathbf{Q}} \sum_{\Psi} \sum_{\mathcal{K}} P(\mathbf{O}, \mathbf{Q}, \Psi, \mathcal{K}|\lambda) \frac{1}{b_{\Psi_t \mathcal{K}_t}(\mathbf{O}_t')} \cdot \frac{\partial b_{\Psi_t \mathcal{K}_t}(\mathbf{O}_t')}{\partial \mathbf{O}_t'} \\ &= \sum_{l=1}^{L} \sum_{j=1}^{N} \sum_{k=1}^{K} P(\mathbf{O}, Q_t = l, \Psi_t = j, \mathcal{K}_t = k|\lambda) \cdot \\ & \quad \Sigma_{jk}^{-1}(\mathbf{O}_t' - \mu_{jk}) = 0, \end{aligned} \tag{5}$$

where L denotes the number of possible value sets of other hidden variables, and N denotes the number of possible value sets of hidden parents of observation $\mathbf{O}_t$. For clarity, we define a new quantity $\gamma_t(l, j, k) = P(\mathbf{O}, Q_t = l, \Psi_t = j, \mathcal{K}_t = k|\lambda)$, which is named as state occupation probability.

Thus we can find the reestimated inputs $\mathbf{O}_t'$ by

$$\mathbf{O}'_t = \frac{\sum_{l=1}^{L}\sum_{j=1}^{N}\sum_{k=1}^{K}\gamma_t(l,j,k)\Sigma_{jk}^{-1}\mu_{jk}}{\sum_{l=1}^{L}\sum_{j=1}^{N}\sum_{k=1}^{K}\gamma_t(l,j,k)\Sigma_{jk}^{-1}}, \tag{6}$$

where $\gamma_t(l,j,k)$ can be computed using the extension of the frontier algorithm.

3.2 Frontier Algorithm

The frontier algorithm was first proposed in [8] for exact DBN inference. We extend this algorithm to compute $\gamma_t(l,j,k)$ in DBNI. The basic idea of the frontier algorithm is to "sweep" a Markov blanket across the DBN, first forward and then backward. The nodes in the Markov blanket constitute the *frontier set* denoted as $\mathcal{F}$; the nodes to the left and right of the frontier are denoted as $\mathcal{L}$ and $\mathcal{R}$, respectively. At every step of the algorithm, $\mathcal{F}$ *d-separates* [1] $\mathcal{L}$ (past) and $\mathcal{R}$ (future). Thus $\mathcal{L}$, $\mathcal{F}$ and $\mathcal{R}$ constitute a chain structure. Generalized version of the forward-backward (FB) algorithm is applied to the chain. Without lose of generality, we give the implementation of the frontier algorithm for the HFM (shown in Fig. 1) which is a typical case of the DBN topology in Fig. 2.

For clarity in derivation, we denote the hidden variables $\{Q_t, \Psi_t, \mathcal{K}_t\}$ as X_t^i, where $i = 1, 2, \cdots, D$. Model parameters λ are omitted. We define

$$\alpha_t = P(X_t^{1:D}|\mathbf{O}_{1:t}), \tag{7}$$
$$\beta_t = P(\mathbf{O}_{t+1:T}|X_t^{1:D}), \tag{8}$$

and thus we get

$$\gamma_t = P(\mathbf{O}, X_t^{1:D}) \propto P(X_t^{1:D}|\mathbf{O}_{1:t}) \cdot P(\mathbf{O}_{t+1:T}|X_t^{1:D}) = \alpha_t \cdot \beta_t. \tag{9}$$

The αs and βs can be efficiently calculated by the FB recursions. Here we provide the calculation of the forward pass (α). The frontier initially contains all the nodes in frame $t-1$ (Fig. 3 (a)):

$$F_{t,0} \overset{\text{def}}{=} \alpha_{t-1} = P(X_{t-1}^{1:D}|\mathbf{O}_{1:t-1}). \tag{10}$$

First we add node X_t^1 to the frontier (move it from $\mathcal{R}$ to $\mathcal{F}$) since all its parents are already in the frontier (Fig. 3 (b)). So we multiply in its CPD $P(X_t^1|X_{t-1}^1)$:

$$F_{t,1} = P(X_t^1, X_{t-1}^{1:D}|\mathbf{O}_{1:t-1}) = P(X_t^1|X_{t-1}^1) \times F_{t,0}. \tag{11}$$

Since all the children of X_{t-1}^1 are in the frontier, we then remove it (move it from $\mathcal{F}$ to $\mathcal{L}$) by marginalizing X_{t-1}^1 out (Fig. 3 (c)):

$$F_{t,2} = P(X_t^1, X_{t-1}^{2:D}|\mathbf{O}_{1:t-1}) = \sum_{X_{t-1}^1} F_{t,1}. \tag{12}$$

Next we add X_t^2 since all its parents are in the frontier (Fig. 3 (d)):

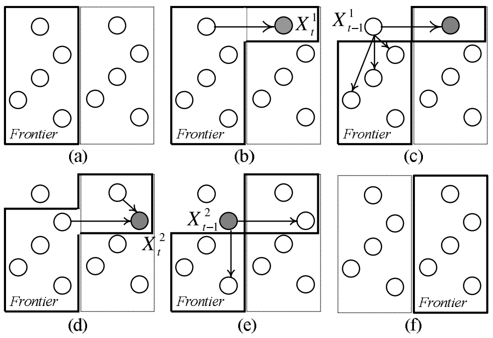

Fig. 3 Forward pass of the frontier algorithm applied to the HFM given in Fig. 1. Observation nodes are omitted for clarity. The nodes inside the black box constitute the frontier. The nodes being operated on are shown shaded; only connections with its parents or children are shown; other arcs are also omitted.

$$F_{t,3} = P(X_t^{1:2}, X_{t-1}^{2:D}|\mathbf{O}_{1:t-1}) = P(X_t^2|X_t^1, X_{t-1}^2) \times F_{t,2}. \tag{13}$$

Since all the children of X_{t-1}^2 are in the frontier, we then remove it by marginalizing it out (Fig. 3 (e)):

$$F_{t,4} = P(X_t^{1:2}, X_{t-1}^{3:D}|\mathbf{O}_{1:t-1}) = \sum_{X_{t-1}^2} F_{t,3}. \tag{14}$$

The process continues in this way and ends at (Fig. 3 (f)):

$$F_{t,D} = P(X_t^{1:D}|\mathbf{O}_{1:t-1}). \tag{15}$$

Finally we get

$$F_{t+1,0} = \alpha_t = P(X_t^{1:D}|\mathbf{O}_{1:t}) \propto P(\mathbf{O}_t|X_t^{1:D}) \times F_{t,D} = P(\mathbf{O}_t|X_t^{2:D}) \times F_{t,D} = b_t(\mathbf{O}_t) \times F_{t,D}. \tag{16}$$

The βs can be computed using the backward pass in a similar way as for αs, while the frontier moves in the opposite direction of the forward pass.

4. Experiments

We have carried out experiments on the Aurora 2.0 database [5] to demonstrate the effectiveness of the proposed DBNI algorithm. We adopted the ETSI standard front-end processing (WI007), resulting in a 39-dimensional feature vector. The feature vector contains 12 Mel-frequency cepstral coefficients (MFCCs), the energy term, as well as their velocity and acceleration derivatives.

We used the hidden feature model (HFM) shown in Fig. 1 as our DBN speech model, where *articulators* were adopted as the hidden features since our previous work has shown articulatory modeling of speech presents superior performance [9]. We used 9 articulator variables [9], i.e., *voicing*, *velum*, *manner*, *tongueBodyLH*, *tongueBodyBF*, *lipRounding*, *lipSeparation*, *tongueShow* and *teechShow*, all in discrete values.

In the experiments, we compared the proposed DBNI with the HMMI [3] in terms of word accuracy. The experiments involved two HMM systems, i.e., HMMs without

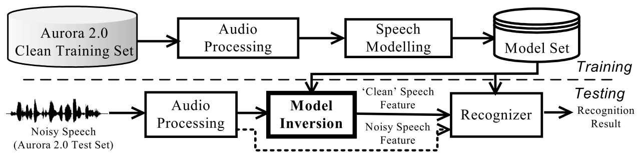

Fig. 4 Block diagram of the speech recognition system. The dotted line in the testing is for the two baseline systems only.

HMMI (HMM-*baseline*, as in [5]) and HMMs with HMMI (HMM-HMMI). The digits were modeled as whole word, left-to-right HMMs with 16 states and 3 Gaussians per state. HTK 3.1 toolkit was modified to implement the HMMI algorithm.

Correspondingly, we built two DBN systems, i.e., HFMs without DBNI (HFM-*baseline*) and HFMs with DBNI (HFM-DBNI). To make a comparative study, we designated a set of 176 values to the state variable of the HFMs, corresponding to the 11 sixteen-state digit HMMs. The CPDs $P(\mathbf{O}_t|\Psi_t, \mathcal{K}_t)$ were trained using the standard EM algorithm [7] via a home-grown DBN toolkit. According to the model structure in Fig. 1, a set of 3 Gaussian mixtures was trained for each possible combination of articulatory feature values. Other CPDs were trained using a supervised learning method described in [9]. A parameter reduction scheme was adopted based on the physical constraints among the articulators. Finally we trained 128 sets of Gaussian mixtures.

Figure 4 shows the block diagram of the speech recognition systems used in the experiments. In the training process, speech models (HMMs or HFMs) were trained using the Aurora 2.0 clean training set. In the testing process, noisy speech from Set A, B, and C were first converted to 'clean' speech using the inversion algorithm (HMMI or DBNI), and then were fed into the recognizer. We did not perform the multi-conditional training that involved both clean and noisy data, since our objective is to see the noise-removal merit of the model inversion algorithms.

Experimental results for Set A, B, C, and babble noise (in Set A) are summarized in Table 1 and Table 2. As can be seen, compared with the HMM-*baseline*, the HMM-HMMI system relatively improves the overall performance by 21.61%, and the HFM-*baseline* system relatively improves the overall performance by 7.89%. The HFM-DBNI system that adopts the proposed DBNI algorithm, achieves the superior performance with a relative word error rate reduction of 38.08% as compared to the HMM-*baseline*. Specifically, for the babble noise that is speech-like, the HFM-DBNI system achieves a relative improvement as high as 44.79%. The promising improvement may be attributed to the facts that: (1) the DBN-based hidden feature model can model the speech production more precisely than the simple linguistic unit model; and (2) the ML-based inversion algorithm can move the noisy speech closer to the Gaussian means trained on the clean speech data.

In the experiments, the average number of iterations performed for the HMMI and DBNI are 6.7 and 6.9, respectively, which shows that the inversion algorithms converge within a very few iterations. The relative time spent for recognition for the four systems are: 1.0 (HMM-*baseline*), 1.07 (HMM-HMMI), 1.18 (HFM-*baseline*) and 1.32 (HFM-DBNI).

Table 1 Aurora 2.0 word recognition accuracy (%) trained on *clean* data and tested on Set A, B, C and babble noise.

System	Set A	Set B	Set C	Overall	babble
HMM-*baseline*	61.34	55.74	66.14	60.06	49.88
HMM-HMMI	69.49	65.83	71.95	68.69	60.17
HFM-*baseline*	63.12	58.27	68.20	63.21	53.36
HFM-DBNI	75.49	74.11	76.57	75.27	72.33

Table 2 Relative word error rate reduction (%) over the HMM-*baseline* system.

System	Set A	Set B	Set C	Overall	babble
HMM-HMMI	21.08	22.80	17.16	21.61	20.53
HFM-*baseline*	4.60	5.72	6.08	7.89	6.94
HFM-DBNI	36.60	41.50	30.80	38.08	44.79

5. Conclusions

In this paper, we present an inversion algorithm for typical dynamic Bayesian networks used in speech recognition, namely DBNI, which is a generalization of the HMMI algorithm [3]. The DBNI can provide both the expressive advantage from DBN and the noise-removal feature from model inversion, intuitively leading to more superior performance. Experiments on Aurora 2.0 database have shown the effectiveness of the DBNI algorithm in performance improvement. The preliminary results indicate that the noise-removal feature of the dynamic Bayesian network inversion algorithm is quite promising for robust speech recognition.

References

[1] F.V. Jensen, Bayesian networks and decision graphs, Springer-Verlag, 2001.
[2] G.G. Zweig, "Bayesian network structures and inference techniques for automatic speech recognition," Comput. Speech Lang., vol.17, pp.173–193, 2003.
[3] S.Y. Moon and J.N. Hwang, "Noisy speech recognition using robust inversion of hidden Markov models," ICASSP'95, pp.145–148, 1995.
[4] K. Choi, Y. Luo, and J. N. Hwang, "Hidden Markov model inversion for audio-to-visual conversion in an MPEG-4 facial animation system," Journal of VLSI Signal Processing, vol.29, no.1-2, pp.51–61, 2001.
[5] D. Pearce and H.-G. Hirsch, "The Aurora experimental framework for the performance evaluation of speech recognition systems under noisy conditions," ICSLP, vol.4, pp.29–32, 2000.
[6] K. Livescu, J. Glass, and J. Bilmes, "Hidden feature models for speech recognition using dynamic Bayesian networks," Eurospeech'03, pp.2529–2532, 2003.
[7] A. Dempster, A.N. Laird, and D. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," J. of Royal Stat. Soc. (Series B), vol.39, pp.89–111, 1977.
[8] G. Zweig, A forward-backward algorithm for inference in bayesian networks and an empirical comparison with HMMs, Master Thesis, UC Berkeley, 1996.
[9] L. Xie and Z.Q. Liu, "Multi-stream articulator model with adaptive reliability measure for audio visual speech recognition," ICMLC'06, LNCS 3930, pp.994–1004, 2006.