International Journal of Fuzzy Logic and Intelligent Systems, vol. 7, no. 4, December 2007 pp. 285-294

Online Parameter Estimation and Convergence Property of Dynamic Bayesian Networks

Hyun Cheol Cho[1], M. Sami Fadali[2] and Kwon Soon Lee[1]

[1] Dept. of Electrical Engineering, Dong-A University, Pusan, 604-714, Korea
[2] Dept. of Electrical Engineering/260, University of Nevada, Reno, NV 89557, USA

Abstract

In this paper, we investigate a novel online estimation algorithm for dynamic Bayesian network (DBN) parameters, given as conditional probabilities. We sequentially update the parameter adjustment rule based on observation data. We apply our algorithm to two well known representations of DBNs: to a first-order Markov Chain (MC) model and to a Hidden Markov Model (HMM). A sliding window allows efficient adaptive computation in real time. We also examine the stochastic convergence and stability of the learning algorithm.

Key words : dynamic Bayesian networks, online parameter estimation, convergence property, sliding window, Markov chain

1. Introduction

A dynamic Bayesian network (DBN) is a graphical model of stochastic causal systems [1]. The model represents conditional probabilities for random variables of system states based on observation data. These probabilities are the DBN parameters that must be optimally estimated for modeling accuracy and reliable inference.

Although, there is no standard procedure for DBN parameter learning, in general, researchers have used Maximum Likelihood (ML) estimation [2] and the Expectation-Maximization (EM) algorithm [3]. The underlying scheme is to maximize likelihood with respect to the parameter vector for given observation data. These algorithms were successfully utilized for acoustic modeling with finite observation data [4]. However, online learning using these algorithms is difficult because of the heavy computational burden of the optimization routine. Moreover, the EM algorithm can settle at a local minimum and yield suboptimal parameters [3].

Applications of DBN include fault detection/diagnosis [5], control systems [6], probability density estimation [7], etc., where stochastic modeling is warranted. Many of these applications involve nonstationary statistics and a large data set. DBN modeling of such systems requires online learning and adaptation.

Surprisingly, few investigators have addressed online iterative learning. In [8], online learning algorithm for HMMs was explored using gradient optimization for likelihood maximization. The authors also incorporated the algorithm into the EM framework for online EM learning of HMMs. In [9], a recursive parameter estimate for HMMs based on extended least squares and recursive state prediction error minimization was developed. The authors proved that their algorithm is locally convergent and more computationally efficient via an ordinary differential equation model. In [10], the authors proposed an online HMM estimation algorithm that re-estimates the 'lifted' parameters using past data history. More recently, in [11], the voting EM algorithm which is an extension of EM method was proposed for adaptive online learning of (static) Bayesian networks. The learning rate of the dynamic estimation rule is adaptively changed against the modeled environment. In [12], EM and voting EM algorithms were compared by application to a flood decision problem. The authors concluded the latter method gave better result for online learning.

Most algorithms for DBN learning involve complex mathematical formulas and are not suitable for real time implementation. In this paper, we explore an efficient learning algorithm for DBN parameter estimation. First, we present an online DBN parameter adjustment rule. Next, we apply our algorithm to a MC model, then to a HMM. Because a long data record is involved, there is a need to use a subset of the data for on-line computation. In an adaptive environment, older data points can be discarded and more recent data is sufficient. We limit data length by introducing a sliding window approach with an adaptive window size for flexible selection of the data sequence. We also study the stochastic convergence and stability properties of the proposed learning algorithm.

Manuscript received Oct. 10, 2007; revised Dec. 11, 2007.
This work was supported by the Korea Science and Engineering Foundation (KOSEF) through the National Research Lab. Program funded by the Ministry of Science and Technology (No. M10300000303O6J000030310).

The remainder of this paper is as follows: In Section 2, we describe our proposed learning algorithm and apply it to a MC and a HMM structure respectively, including computer simulation. In Section 3, we introduce the sliding window learning algorithm. In Section 4, we consider the stochastic convergence and stability of the learning algorithm. Finally, conclusions and suggestions for future are given in Section 5.

2. Learning Algorithm

In this section, we derive the iterative equations of our learning algorithm and apply them to two DBN examples. We begin with the simplest possible DBN, namely a first-order MC. Then we extend our algorithm to the more complex HMM.

2.1 First-order Markov Model

We consider a first-order MC model as a simple DBN, shown in Fig. 1. Let X be a random variable with N distinct states labeled by $\{1,\cdots,N\}$ in Fig. 1. The model parameters are the conditional probabilities expressed by

$$a_{ij} = p(X(k) = j \mid X(k-1) = i), \quad i, j \in \{1,\cdots,N\} \tag{1}$$

where k denotes discrete time. The parameters a_{ij} satisfy the constraint

$$\sum_{j=1}^{N} a_{ij} = 1 \tag{2}$$

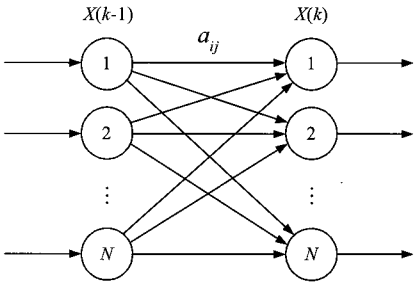

Fig. 1. A simple MC model.

Parameter learning is required to estimate the probabilities in (1) based on sequential observation data. To derive our parameter estimates, we first define the parameters in (1) as

$$a_{ij}(k) = \alpha m_{ij}(k), \qquad i, j \in \{1,\cdots,N\} \tag{3}$$

where α is a normalizing factor and m_{ij} is the average likelihood and is defined as

$$m_{ij}(k) = \frac{1}{k}\sum_{n=1}^{k} \zeta_{ij}(n) = \left(\frac{k-1}{k}\right) m_{ij}(k-1) + \left(\frac{1}{k}\right)\zeta_{ij}(k) \tag{4}$$

where ζ_{ij} are selected as 1 or 0 based on the observation. If we observe that $X = i$ at $k-1$ and $X = j$ at k (or $X(k) = j \mid X(k-1) = i$), then $\zeta_{ij}(k) = 1$, otherwise $\zeta_{ij}(k) = 0$, i.e.

$$\zeta_{ij} = \begin{cases} 1, & \text{if } X = j \text{ given } X = i \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

Clearly, if $\zeta_{ij}(k)=1$, then the corresponding average likelihood $m_{ij}(k)$ increases while other average likelihoods decrease. The likelihood values are then normalized for the parameters a_{ij} to satisfy the constraint (2). This estimation procedure is simple to realize in practice as illustrated in the following Example.

***Example* 1**: We consider a nonstationary MC model with three random states. An observation signal is randomly generated with zero-mean uncorrelated Gaussian distribution, i.e. $X \sim N(0, \sigma^2)$, where σ^2 is uniformly distributed in [0,1]. Fig. 2 shows a realization of the random observations sequence. We discretize the continuous observations to three states:

$$\begin{aligned} x_1 &= \{X : X \in (-\infty, -0.5)\} \\ x_2 &= \{X : X \in [-0.5, 0.5]\} \\ x_3 &= \{X : X \in (0.5, \infty)\} \end{aligned} \tag{6}$$

where x_i, i = 1, 2, 3 is ith state for X. Since the model has three states, the total number of parameters is nine (3^2). We simulate the model and estimate the parameters using our estimation algorithm. Fig. 3 shows the trajectories of the estimated transition probabilities. We observe that after about 300 data points the responses approaches steady-state behavior. We calculate the state probability vector $\hat{p}(X)$ with estimated parameters and prior probability vector, i.e.

$$\hat{p}(x_i(k)) = \sum_{j=1}^{N} a_{ji}\hat{p}(x_j(k-1)), \quad i = 1,\cdots,N \tag{7}$$

We utilize the estimation in (7) to test the validity of the estimates of a_{ij} compared to the result of Parzen estimation [13]. The latter algorithm is well-known for estimation of probability density functions and is reported to have outstanding performance for large data samples [14]. We simulated both methods using the observation data of Fig. 2 and compared their results using the logarithm error

$$E(k) = \log\left\| p(X(k)) - \hat{p}(X(k)) \right\| \tag{8}$$

where $p(X)$ is the Parzen estimate. The error trajectory is plotted in Fig. 4. The error trajectory shows that the logarithmic error progressively decreases. Thus, the two estimates are asymptotically identical.

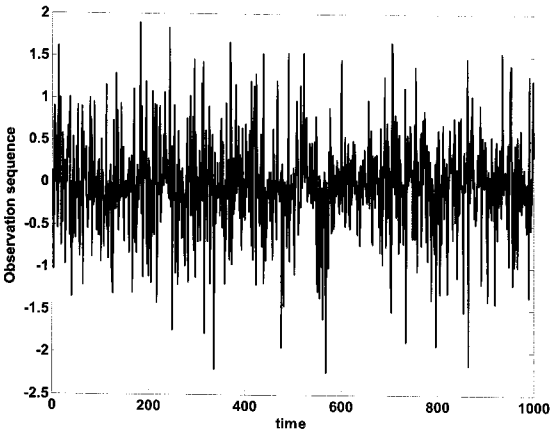

Fig. 2. Observation sequence.

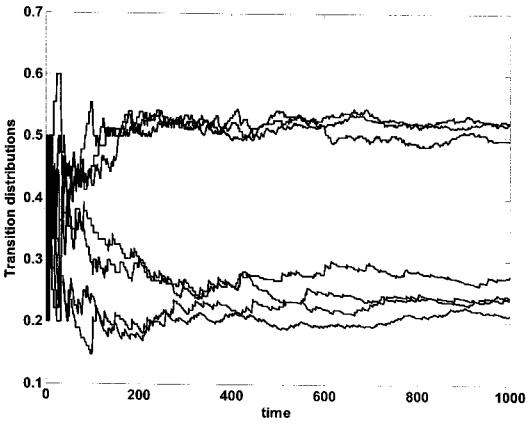

Fig. 3. Estimated transition probabilities.

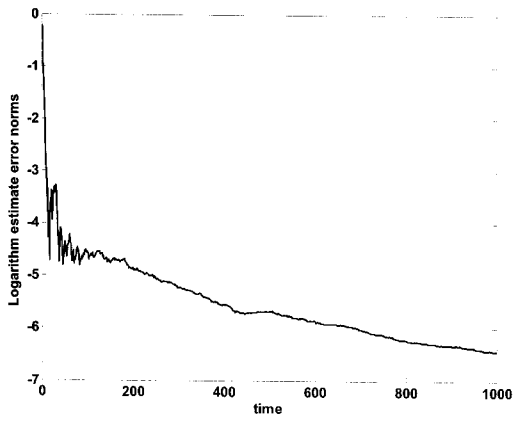

Fig. 4. Estimate error norms.

2.2 Online HMM Learning

We now extend the algorithm of Section 2.1 to a typical HMM model, shown in Fig. 5.

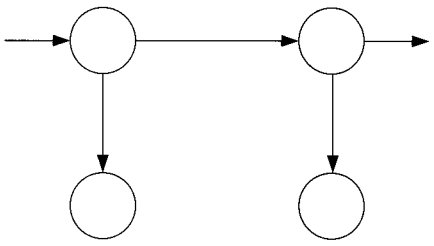

Fig. 5. A typical HMM structure.

This model represents a first-order MC with a hidden state variable X and a discrete observation alphabet $Y = \{y_1,\cdots,y_M\}$ whose symbols denote output values (vector quantization yields a discrete alphabet for continuous systems). In Fig. 5, the transition probability a_{ij} is equally defined as in the case of a MC model. The conditional probability of observation o, when a hidden state i is reached, is expressed by

$$b_{jo} = p(Y(k) = o \mid X(k) = j), \qquad o \in \{1,\cdots,M\} \tag{9}$$

The hidden state probability and observation probability vectors are expressed based on (1) and (9) as

$$X(k) = AX(k-1) \tag{10}$$

$$Y(k) = BX(k) \tag{11}$$

where $A = \{a_{ij}\}$, $i, j = 1,\cdots,N$ and $B = \{b_{jo}\}$, $o = 1, \cdots, M$, are the transition and observation matrices, respectively. Similarly, the transition probability and observation probability are respectively defined as

$$a_{ij}(k) = \alpha\, m_{ij}(k), \qquad i, j \in \{1, \cdots, N\} \tag{12}$$

$$b_{jo}(k) = \alpha\, n_{jo}(k), \qquad o \in \{1, \cdots, M\} \tag{13}$$

where

$$m_{ij}(k) = \left(\frac{k-1}{k}\right) m_{ij}(k-1) + \left(\frac{1}{k}\right) \zeta_{ij}(k) \tag{14}$$

$$n_{jo}(k) = \left(\frac{k-1}{k}\right) n_{jo}(k-1) + \left(\frac{1}{k}\right) \zeta_{jo}(k) \tag{15}$$

and ζ_{ij} and ζ_{jo} are chosen as 0 or 1 as in Section 2.1. If we observe $Y(k) = y_o$ given $X(k) = j$, then $\zeta_{jo} = 1$, otherwise $\zeta_{jo} = 0$. However, the hidden variable $X(k)$ is not observed and we need to determine its (hidden) state. Consider an illustration shown in Fig. 6.

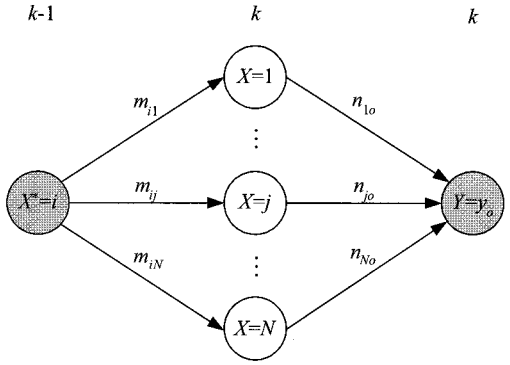

Fig. 6. N paths via hidden HMM variables.

Fig. 6 indicates that the right dark circle is related to observation states at k and the left dark one is assumed to be selected at time $k-1$. Our problem at this point is how to select a suitable hidden state, i.e. $X^*(k)$ under this condition. From Fig. 6, there are N paths between $X^*(k-1)$ and $Y(k)$:

$$\text{Path } l: m_{il} \rightarrow n_{lo}, \qquad l = 1, \cdots, N \tag{16}$$

The likelihood value for each path is easily obtained by multiplying the two likelihoods m_{il} and n_{lo} in Fig. 6. We define this value as a cost function:

$$J_l(k) = m_{il} n_{lo}, \qquad l = 1, \cdots, N \tag{17}$$

We select the hidden state X^* which maximizes the cost functions in (17):

$$Path^*(k) = \arg\max_{X(k)} J_l(k) \tag{18}$$

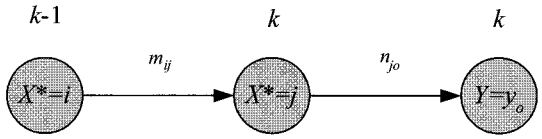

Fig. 7. An optimal path j of a HMM.

This scheme is reasonable since the maximum likelihood stands for the highest probability in the paths between the states in Fig. 6. Following this step, the hidden state is changed from $X(k-1) = i$ to any hidden state $X(k) = j$, $j \in \{1,\cdots,N\}$. Fig. 7 shows illustrations for this procedure.

From Fig. 7, when $\zeta_{ij}=1$ and $\zeta_{jo}=1$, then the optimal path is $X = i \rightarrow X = j \rightarrow Y = y_o$. The likelihoods m_{ij} and n_{jo} are increased while all other parameters are unchanged. Consequentially, after normalizing, the corresponding probabilities, i.e. a_{ij} and b_{jo} in (12) and (13), are increased, but all other probabilities are decreased. The probabilities are sequentially updated by repeating this calculation. The computational load is low and the algorithm is suitable for online learning. The application of the algorithm to a HMM is summarized in Fig. 8. The following *Example* 2 demonstrates the validity of our algorithm for online HMM parameter estimation.

---

1) Initialization (k=0)
   $m(i,j) = 0$;
   $n(j,o) = 0$;
   $m_\pi(i) = [0,1]$; randomly unformed distribution
   $\pi(i) = \alpha\, m_\pi(i)$; initial distributions of state variables
   Old_Node_X* = $\arg\max_{i\in[1,N]} \pi_i$

2) $k = 1 \sim T$; $T \rightarrow \infty$
   Observation $Y = y_o$; $o \in [1,M]$
   New_Node_Y* = o

   $J_l = m(i,l) \cdot n(n,o)$
   New_Node_X* = $\arg\max_{i\in\{1,N\}} J_l$; $l = 1, \ldots, N$

   Update of transition probabilities, a_{ij}
   For $i = 1 \sim N$
      For $j = 1 \sim N$
         If i = Old_Node_X* & j = New_Node_X*
            Then $\zeta(i,j) = 1$
            Else $\zeta(i,j) = 0$
         $m(i,j) = (k\text{-}1)/k \cdot m(i,j) + (1/k) \cdot \zeta(i,j)$
         $a(i,j) = \alpha\, m(i,j)$
   Update of observation probabilities, b_{jo}
   For $j = 1 \sim N$
      For $o = 1 \sim M$
         If j = New_Node_X* & k = New_Node_Y*
            Then $\zeta(j,k) = 1$
            Else $\zeta(j,k) = 0$
         $n(j,o) = (k\text{-}1)/k \cdot n(j,o) + (1/k) \cdot \zeta(j,o)$
         $b(j,o) = \alpha\, n(j,o)$

   Old_Node_X* $\leftarrow$ New_Node_X*

Fig. 8. Summary of online HMM parameter learning.

Example 2: We adapt the stochastic system used in [15] to demonstrate our algorithm. The system generates three discrete states with {−1, 0, 1}, but this signal is corrupted with uncorrelated nonstationary Gaussian noise, i.e. $w \sim N(0,\sigma^2)$ where σ^2 is uniformly distributed in [0,0.5]. We observe the output data for 1000 points, plotted in Fig. 9. We need two state variables to construct a HMM based on prior information about the system: the hidden state variable $X \in \{x_1, x_2, x_3\}$ and the observation variable $Y \in \{y_1, y_2, y_3\}$ as

$$x_1 = -1, \quad x_2 = 0, \quad x_3 = 1 \tag{19}$$

and

$$y_1 = \{Y : Y \in (-\infty, -0.5)\}$$
$$y_2 = \{Y : Y \in [-0.5, 0.5]\} \quad (20)$$
$$y_3 = \{Y : Y \in (0.5, \infty)\}$$

We applied the proposed learning algorithm to this simulation scenario. The results for the two conditional probabilities are given in Fig. 10 and Fig. 11, respectively.

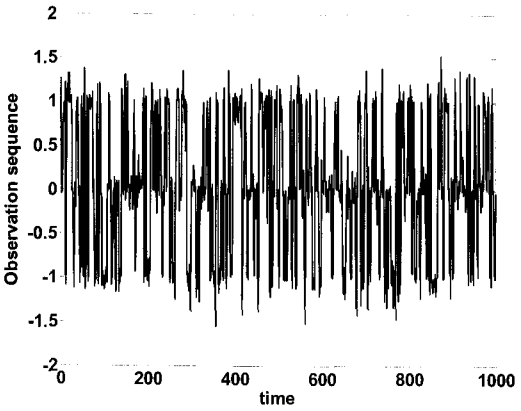

Fig. 9. Noisy observation sequence.

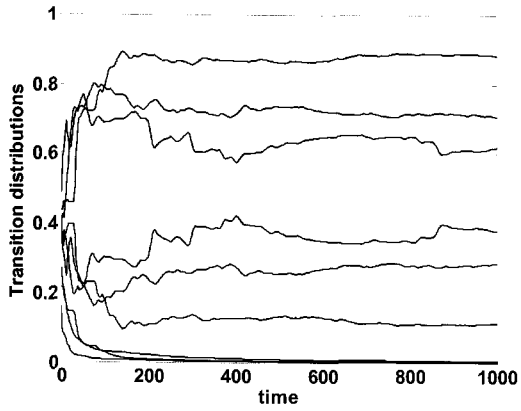

Fig. 10. Estimated transition probabilities.

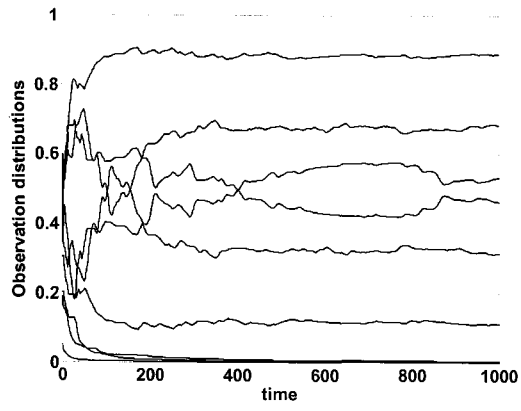

Fig. 11. Estimated observation probabilities.

Since the number of states for the hidden and observation variables is three, the total number of the parameters is nine (3^2) and there are nine curves for each figure. The figures illustrate transient behaviors until about 100 data points. Then the dynamics slowly converge towards steady-state values. However, we observe that two parameters continue to vary throughout the simulation i.e. are nonstationary. Using the estimated probabilities from Figs. 10 and 11, we calculate the system output probability vector as

$$\hat{p}(Y(k+1)) = B(k)\left[A(k)p(X(k))\right] \quad (21)$$

The posterior estimated probability is linearly dependent on the prior probability $p(X)$, and on the entries of the matrices $A(k)$ and $B(k)$ sequentially obtained via the learning procedure. We similarly use the estimator (21) to evaluate our algorithm and compare it to Parzen estimation. We define the logarithmic error between these two estimations as

$$E(k) = \log\left\| p(Y(k)) - \hat{p}(Y(k)) \right\| \quad (22)$$

Fig. 12 shows the error trajectory for (22). From the figure we observe that the logarithm of the deviations remains below −2 after a short transient period then continues dropping. The two estimates are approximately equal after a few hundred points.

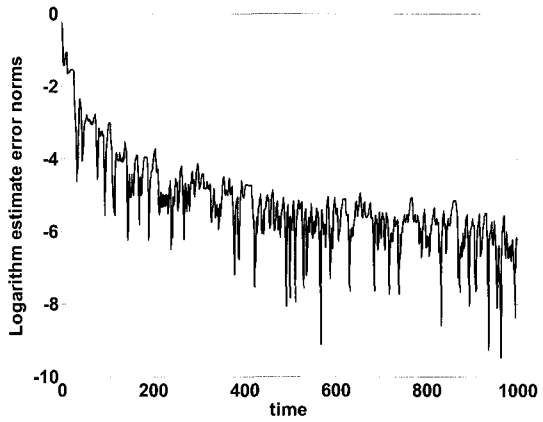

Fig. 12. Estimate error norms.

3. Sliding Window Approach

Online DBN learning generally involves large training data sets with each data point successively arriving for a prolonged time interval. However, a large data sequence is computationally costly and not all data points are necessary for learning. Typically, old data may be neglected and more recent data retained. To embed this scheme in our DBN learning algorithm, a sliding window is introduced in estimation rules (14) and (15).

Update rules for these two likelihoods with a sliding window are respectively given by

$$m_{ij}(k)=\left(\frac{N_w}{N_w+1}\right)m_{ij}(k-1)+\left(\frac{1}{N_w+1}\right)\zeta_{ij}(k) \quad (23)$$

$$n_{jo}(k)=\left(\frac{N_w}{N_w+1}\right)n_{jo}(k-1)+\left(\frac{1}{N_w+1}\right)\zeta_{jo}(k) \quad (24)$$

where N_w is the window size. As shown in Fig. 13, the window slides as k increases and old data are discarded.

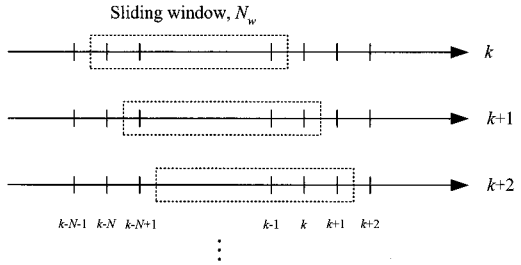

Fig. 13. A sliding window of DBNs.

The update rules of (23) and (24) require the current time k to be larger than the window size N_w i.e. $k > N_w$. Initially, it is possible for the data set to be shorter than the window size $k < N_w$. Therefore, we use the rules in (23) and (24) during the transient state $k < N_w$, and for $k > N_w$, we apply the window based learning.

3.1 Adaptive Window Size

In (23) and (24), the window size N_w is assumed constant for simplicity. In practice, window size selection is a key issue for flexible learning. Intuitively, a narrow window reduces computational load but does not use many data points. By contrast, a wider window uses many data points but increases the computational burden. Thus, a dynamic window whose size is temporally adjusted is often required. Window adjustment is dependent on the degree of data spread. For example, a narrow window is suitable for a data sequence concentrated around a specific value, whereas for widely spread data a wider window is more effective. The degree of data spread is statistically characterized by the variance. We base our adaptive window size on the norm of the variances for the probability. We define the window size adjustment rule as

$$N_w(k+1)=N_w(k)+c\eta \quad (25)$$

where c is arbitrary positive value and η is a coefficient to be selected based on the norm of the data variance. The coefficient η is selected as

$$\eta=\begin{cases}1; & \varepsilon_2 \le \|s\| \\ 0; & \varepsilon_1 \le \|s\| \le \varepsilon_2 \\ -1; & \text{otherwise}\end{cases} \quad (26)$$

where ε_1 and ε_2 are specified bounds. The norm of variance is given by

$$\|s\|=\sum_i\sum_j s_{ij} \quad (27)$$

where s_{ij} is the expectation of the deviation between $m_{ij}(k)$ and $m_{ij}(k-1)$

$$s_{ij}=E\{[m_{ij}(k)-m_{ij}(k-1)]^2\} \quad (28)$$

In summary, the norm of s is obtained by adding the square deviations in (27) and is then used to update window size by the rule (26). Window size is increased if the norm of the variance is larger than an upper bound ε_2, and decreased if the norm is less than a lower bound ε_1. Otherwise, the window size is unchanged.

4. Convergence Analysis

In this Section, we investigate the convergence of the proposed online learning for a large data sample (an infinite number of observations). We apply asymptotic convergence theorems to DBN parameter estimators. In addition, stability theory of dynamic systems is introduced to the learning rule viewed as a discrete time-variant system. First, recall the adjustment rule from Section 2 as

$$m_{ij}(k+1)=a(k)m_{ij}(k)+b(k)\zeta_{ij}(k) \quad (29)$$

$$\hat{\theta}_{ij}(k)=\alpha\, m_{ij}(k) \quad (30)$$

where $a(k)=k^{-1}(k-1)$ and $b(k)=k^{-1}$. Since ζ_{ij} = 1 or 0, we model it as a random variable with Bernoulli distribution, i.e.

$$p(\zeta=\varsigma_{ij})=q^{\varsigma_{ij}}(1-q)^{1-\varsigma_{ij}}, \qquad \varsigma_{ij}=0,1 \quad (31)$$

where $q = p(\zeta_{ij} = 0) \in (0,1)$. Recall that for the Bernoulli distribution, the mean and mean square are

$$E(\zeta)= \quad E(\zeta^2)=q \quad (32)$$

4.1 Stochastic Convergence of DBNs

We first state some useful theorems and definitions for large-sample estimators.

***Definition* 1** *For a sequence of random variables X on some probability space, $X(k)$ converges in probability to the random variable X, if, for $\varepsilon>0$, $\lim_{k\to\infty}P\{|X(k)-X^*|>\varepsilon\}=0$. This formula is simply denoted by $X(k)\xrightarrow{p}X^*$.* ■

***Definition* 2** *We say that a sequence of random variables X(k) converges to X^* in a mean-squared sense, if* $\lim_{k\to\infty} E\{(X(k)-X^*)^2\}=0$. *This is a specific definition of convergence in rth mean (see page 93 in* [2]). ■

***Theorem* 1** *If a sequence of random variables X(k) converges to X^* in mean square, then it converges to X^* in probability.*

Proof *See page 97 in* [2]. ■

Because a direct proof of *convergence in probability* is difficult, we typically prove *convergence in mean square*, which by *Theorem* 1 converges in probability. The next theorem governs the convergence of a function of a random variable.

***Theorem* 2** *Assume X(k) converges to X^* in probability or* $X(k)\xrightarrow{p}X^*$ *and let g(·) be a continuous function, then* $g(X(k))\xrightarrow{p}g(X^*)$ *as* $k\to\infty$.

Proof See page 24 in [16]. ■

In a MC, the posterior probability of the state variable is given by

$$p_i(k+1)=a_j{}^T p(k) \tag{33}$$

where $p_i(k+1)$ is *i*th element of the stochastic vector, $p(k)$ is a prior probability vector, and the parameter vector $a_j = col\{a_{1j},\cdots,a_{ij}\}$, $i,j=1,\cdots,N$. Substituting (30) in (33), we have

$$p_i(k+1)=(\alpha m_j^T)p(k) \tag{34}$$

where $m_j = col\{m_{1j},\cdots, m_{ij}\}$, $i, j = 1,\cdots,N$. The expression shows that the posterior probability is a linear function of the estimate $m \in \{m_{1j},\cdots, m_{ij}\}$, $i,j=1,\cdots,N$. By *Theorem* 2, we can therefore examine the asymptotic behavior of the posterior probability through the asymptotic behavior of m. We prove mean square convergence of m, which by *Theorem* 1 is sufficient to conclude convergence in probability. Thus, we show that $p_i(k+1)$ converges in probability.

Lemma 1 *The sequence of the random variable m_{ij} in (29) asymptotically converges in mean-square to q.*

Proof *We seek to prove that*

$$\lim_{k\to\infty} E\left\{\left(m_{ij}(k+1)-q\right)^2\right\}=0 \tag{35}$$

where $q = E\{\zeta\}$. Using (4), we expand the limit as

$$\begin{aligned}&\lim_{k\to\infty} E\left\{\left(\frac{1}{k^2}\left[\sum_{n=1}^{k}\varsigma_n\right]^2-\frac{2q}{k}\sum_{n=1}^{k}\varsigma_n+q^2\right)\right\}\\ &=\lim_{k\to\infty} E\left\{\left(\frac{1}{k^2}\left[\sum_{n=1}^{k}\varsigma_n^2+\sum_{n=1}^{k}\sum_{\substack{l=1\\ l\neq n}}^{k}\varsigma_n\varsigma_l\right]-\frac{2q}{k}\sum_{n=1}^{k}\varsigma_n+q^2\right)\right\}\end{aligned} \tag{36}$$

For i.i.d. Bernoulli trials, the expression becomes

$$\lim_{k\to\infty}\left(\frac{q}{k^2}+\frac{k(k-1)}{k^2}q^2-2q^2+q^2\right)=0 \ ■ \tag{37}$$

Remark From Lemma 1 and *Theorem* 1, *we conclude that $p_i(k+1)$ in* (33) *converges in probability.* Therefore, the estimators in (30) stochastically converge such that the state probability vector is correspondingly asymptotic for a MC model.

4.2 HMM Convergence

We extend the MC convergence properties of Section 4.1 to a HMM. From (10) and (11), we rewrite the observation probability vector with respect to the two stochastic matrices and the prior state probability vector as

$$p(Y(k))=B(k)A(k)p(X(k-1)) \tag{38}$$

Note the entries a_{ij}, $i,j=1,\cdots,N$ and b_{jo}, $o=1,\cdots,M$ of A and B are parameters of a HMM which are estimated via a learning procedure. These estimators are computed using the same learning rule as a MC and similarly converge in probability according to *Lemma* 1, i.e. $a_{ij}(k)\xrightarrow{p}a_{ij}^*$ and $b_{jo}(k)\xrightarrow{p}b_{jo}^*$. Assuming that all entries of A and B converge in probability, we have

$$A(k)\xrightarrow{p}A^*,\quad B(k)\xrightarrow{p}B^* \tag{39}$$

Based on (39), we conclude the convergence of the random observation vector of the HMM in (38).

***Theorem* 3** *If random variables X_1 and X_2 converge to X_1^* and X_2^* in probability, then $(X_1 \cdot X_2)$ also converges to $(X_1^* \cdot X_2^*)$ in probability.*

Proof See page 26 in [16]. ■

Theorem 3 governs the convergence of the product of two convergent random sequences. By *Theorem* 3, the matrix product $B(k)A(k)$ in (38) converges in probability since A and B respectively converge (see (39)). Moreover, from (38), $p(Y(k))$ is a function of the product $B(k)A(k)$ and by *Theorem* 2, $p(Y)$ converges in probability.

4.3 Stability of the Online Learning

Next, we discuss the stability of the time-variant dynamic systems to our learning algorithm. The estimation rules of (23) and (24) are rewritten in vector form as

$$m(k+1) = F(k)m(k) + G(k)\zeta(k) \tag{40}$$

$$\theta(k) = C\,m(k) \tag{41}$$

where m, ζ, $\theta \in \Re^{N^2}$ are nonnegative vectors and F, G, $C \in \Re^{N^2 \times N^2}$ are the corresponding nonnegative matrices. Note that $F(k) = (k-1/k)I_N$ and $G(k) = (1/k)I_N$, are time-variant and diagonal with elements less than unity. Similarly, C in (41) is a diagonal and nonnegative matrix whose elements are less than unity. We first state stability results for both the zero-input response and the zero-state response.

***Theorem* 4** [17]. *Consider an unforced linear discrete time-varying system as $x(k+1) = F(k)x(k)$. Its solution vector is $x(k) = \phi(k,k_0)x(k_0)$, $k_0 > k$, where the state-transition matrix $\phi(k,k_0) = F(k)F(k-1)\cdots F(k_0)$. If a norm of the solution $\|x(k)\| \to 0$ as $k \to \infty$ for any initial state $x(k_0)$, this system is asymptotically stable. This is equivalent to the condition $\|\phi(k,k_0)\| \to 0$ as $k \to \infty$.* ■

***Lemma* 2** The dynamic equation (40) is asymptotically stable for any initial state $m(k_0)$ at initial time k_0.

Proof We expand the state-transition matrix for the equation in (40) as

$$\phi(k,k_0) = \left(\prod_{i=k_0}^{k}\left(\frac{i-1}{i}\right)\right)I_N = \left(\frac{k_0}{k}\right)I_N, \quad k_0 < k \tag{42}$$

The limit as $k \to \infty$ is

$$\lim_{k\to\infty}\phi(k,k_0) = \left(\lim_{k\to\infty}\left(\frac{k_0}{k}\right)\right)I_N = \mathbf{0} \tag{43}$$

From *Theorem* 4, we conclude that the recursion (40) is asymptotically stable. ■

Next, we consider Bounded Input Bounded Output (BIBO) stability. A system is said to be BIBO stable if and only if for any bounded system input, the system output is bounded. A necessary and sufficient condition for BIBO stability is given in the following theorem.

***Theorem* 5** [17]. *For a time-varying dynamic system $x(k+1) = F(k)x(k) + G(k)u(k)$ and $y(k) = C(k)x(k)$, system is BIBO stable if and only if there exists a positive constant $d < \infty$, such that*

$$\sum_{i=k_0}^{k}\left\| C(k)\phi(k,i)G(i)\right\| \le d \ \blacksquare \tag{44}$$

We apply *Theorem* 5 to equations (40) and (41). By the triangle inequality, in (44), the norm is rewritten by

$$\| C(k)\phi(k,i)G(i) \| \le \| C(k) \|\| \phi(k,i)G(i) \| \tag{45}$$

where $\|C(k)\|$ is less than one for all k and

$$\phi(k,i)G(i) = \left(\frac{1}{i}\prod_{j=i}^{k}\frac{1}{j}\right)I_N = \left(\frac{1}{i^2}\prod_{j\neq i}^{k}\frac{1}{j}\right)I_N \tag{46}$$

such that $\|\phi(k,i)G(i)\| < 1$ for all k and $\|\phi(k,i)G(i)\| \to 0$ as $k \to \infty$. Summing both sides of (46), we obtain

$$\sum_{i=k_0}^{k}\left\| C(k)\phi(k,i)G(i)\right\| \le \sum_{i=k_0}^{k}\left\|C(k)\right\|\left\|\phi(k,i)G(i)\right\| < (k-k_0) \tag{47}$$

Designate the second term in (44) as

$$d(k) \equiv \sum_{i=k_0}^{k}\left\|C(k)\right\|\left\|\phi(k,i)G(i)\right\| \tag{48}$$

As $k \to \infty$, then $d(k)$ converges to a finite value $d < \infty$. Thus, we finally have

$$\sum_{i=k_0}^{k}\left\| C(k)\phi(k,i)G(i)\right\| \le d < \infty \tag{49}$$

We conclude that our system is BIBO stable.

5. Conclusions

DBN applications, other than off-line signal processing, require an efficient online learning algorithm. We propose a novel online approach to estimate DBN parameters using an average likelihood for each parameter. The average likelihood is sequentially calculated based on observation data. A windowing approach is utilized for feasible implementation with large data sets. In addition, we explore the use of an adaptive window size that is dynamically adjusted according to data variance. We evaluated its performance through numerical analysis of a HMM with three states and satisfactory results were observed. Output distributions computed using the HMM compared favorably to Parzen pdf estimation.

We also studied the stochastic convergence and stability of DBNs. We applied well known theorems of stochastic convergences to demonstrate the asymptotic behavior for the proposed learning algorithms. Our analytical results show that the learning estimator asymptotically converges to an equilibrium state. Similarly, by utilizing stability criteria we analyzed the dynamic behavior of the parameter estimator.

Potential applications for the proposed learning algorithm include transmission control protocol (TCP) networks, and the detection of a pulse position in pulse-position modulation (PPM).

References

[1] K. Murphy, "Dynamic Bayesian networks: Representation, Inference and Learning." *Ph. D. Dissertation*, UC Berkeley, 2002.

[2] J. M. Mendel, Lessons in estimation theory for signal processing, communications, and control, Prentice Hall, 1995.

[3] T. K. Moon, "The expectation-maximization algorithm," *IEEE Signal Processing Magazine*, vol. 13, no. 6, pp. 47-60, 1996.

[4] X. Huang, A. Acero, and H.-W. Hon, *Spoken language processing*, Prentice Hall, 2001.

[5] J. Ying, T. Kirubarajan, K. R. Pattipati, A. Patterson-Hine, "A hidden Markov model-based algorithm for fault diagnosis with partial and imperfect tests," *IEEE Trans. on Systems, Man and Cybernetics*, vol. 30, no. 4, pp. 463-473, 2000.

[6] D. Hernandez-Hernandez, S. I. Marcus, and P. J. Fard, "Analysis of a risk-sensitive control problem for hidden Markov chains," *IEEE Trans. on Automatic Control*, vol. 44, no. 5, pp. 1093-1100, 1999.

[7] H. C. Cho, S. M. Fadali, and K. S. Lee, "Estimation of non-Gaussian probability density by dynamic Bayesian networks," *Int. Conf. on Control, Automation, and Systems*, pp. 408-413, 2005.

[8] P. Baldi and Y. Chauvin, "Smooth on-line learning algorithm for hidden Markov models," *Neural Computation*, vol. 6, no. 2, pp. 307-318, 1994.

[9] J. J. Ford and J. B. Moore, "Adaptive estimation of HMM transition probabilities," *IEEE Trans. on Signal Processing*, vol. 46, no. 5, pp. 1374-1385, 1998.

[10] J. C. Stiller and G. Radons, "Online estimation of hidden Markov models," *IEEE Signal Processing Letters*, vol. 6, no. 8, pp. 213-215, 1999.

[11] I. Cohen and A. Bronstein, "Adaptive online learning of Bayesian network parameters," *http://www.hpl.hp.com /techreports /2001/HPL-2001-156.pdf*, 2001.

[12] S.-Z. Zhang, H. Yu, N-H. Yang, and X.-K Wang, "An application of online learning algorithm for Bayesian network parameter," *Proc. of the Second International Conf. on Machine Learning and Cybernetics*, pp. 153-156, 2003.

[13] E. Parzen, "On estimation of a probability density function and mode," *Annals of Mathematical Statistics*, vol. 33, no. 3, pp. 1065-1076, 1962.

[14] B. W. Silverman, Density Estimation for Statistics and Data Analysis, Chapman & Hall/CRC, 1986.

[15] V. Krishnamurthy, J. B. Moore, and S.-H. Chung, "Hidden Markov model signal processing in presence of unknown deterministic inferences," *IEEE Transaction on Automatic Control*, vol. 38, no. 1, pp. 146-152, 1993.

[16] R. J. Serfling, Approximation theorems of mathematical statistics, Wiley, 1980.

[17] W. J. Rugh, *Linear system theory*, Prentice Hall, 1996. 2004.

Hyun Cheol Cho received a B.S. from the Pukyong National University in 1997, a M.S. from the Dong-A University, Korea in 1999, and a Ph.D. from University of Nevada-Reno, USA in 2006. He is currently a post-doc. researcher in the Dept. of Electrical Engineering, Dong-A University. His research interests are in the areas of control systems, neural networks, signal processing, and embedded systems.

E-mail : hyunccho@gmail.com

M. Sami Fadali earned a BS in Electrical Engineering from Cairo University in 1974, an MS from the Control Systems Center, UMIST, England, in 1977 and a Ph. D. from the University of Wyoming in 1980. He was an Assistant Professor of Electrical Engineering at the University of King Abdul Aziz in Jeddah , Saudi Arabia 1981-1983. From 1983-85, he was a Post Doctoral Fellow at Colorado State University. In 1985, he joined the Electrical Engineering Dept. at the University of Nevada, Reno, where he is currently Professor of Electrical Engineering. In 1994 he was a visiting professor at Oakland University and GM Research and Development Labs. He spent the summer of 2000 as a Senior Engineer at TRW, San Bernardino. His research interests are in the areas of robust control, robust stability, fault detection, Bayesian networks and fuzzy logic control.

E-mail : fadali@unr.edu

Kwon Soon Lee received the B.S. degree from Chungnam National University, Daejeon, and M.S. degree in electrical engineering from Seoul National University, Seoul, Korea, in 1977 and 1981, respectively,

and the Ph. D. degree in electrical and computer engineering from Oregon State University, USA, in 1990. He joined the Department of Electrical Engineering, Dong-A University, Busan, Korea, as an Assistant Professor from 1982 to 1994. Since October 1, 1994, he has been with Division of Electrical, Electronic, and Computer in Dong-A University, Busan, Korea, where he is currently a Professor. His research interests include all aspects of port automation systems, intelligent control theory, and application of immune algorithm. He is a responsible person of National Research Laboratory nominated by the Korean Ministry of Science & Technology and a team leader of New University for Regional Innovation (NURI) in Dong-A University.

E-mail : kslee@dau.ac.kr