JIRSS (2018)
Vol. 17, No. 02, pp 205-225
DOI: 10.29252/jirss.17.2.10

Generalized Baum-Welch and Viterbi Algorithms Based on the Direct Dependency among Observations

Vahid Rezaei Tabar [1], **Dariusz Plewczynski**[2,3] **and Hosna Fathipour** [4]

[1]Department of Statistics, Faculty of Mathematics and Computer Sciences, Allameh Tabataba'i University, Tehran, Iran.
[2]Laboratory of Functional and Structural Genomics, Centre of New Technologies, University of Warsaw, Warsaw, Poland
[3]Faculty of Mathematics and Information Science, Warsaw University of Technology, Warsaw, Poland
[4]Financial Mathematics Group, Faculty of Financial Sciences, University of Kharazmi, Tehran, Iran.
Received: 12/06/2017, Revision received: 27/02/2018, Published online: 07/08/2018

Abstract. The parameters of a Hidden Markov Model (HMM) are transition and emission probabilities. Both can be estimated using the Baum-Welch algorithm. The process of discovering the sequence of hidden states, given the sequence of observations, is performed by the Viterbi algorithm. In both Baum-Welch and Viterbi algorithms, it is assumed that, given the states, the observations are independent from each other. In this paper, we first consider the direct dependency between consecutive observations in the HMM, and then use conditional independence relations in the context of a Bayesian network which is a probabilistic graphical model for generalizing the Baum-Welch and Viterbi algorithms. We compare the performance of the generalized algorithms with the commonly used ones in simulation studies for synthetic data. We finally apply these algorithms on real data sets which are related to biological and inflation data. We

Corresponding Author: Vahid Rezaei Tabar (vhrezaei@gmail.com)
Dariusz Plewczynski (dariuszplewczynski@cent.uw.edu.pl)
Hosna Fathipour (hosnafathi@yahoo.com)

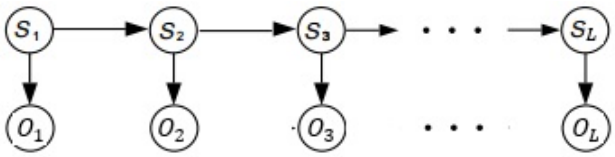

Figure 1: A Bayesian network representing conventional HMM.

show that the generalized Baum-Welch and Viterbi algorithms significantly outperform the conventional ones when sample sizes become larger.

Keywords. Baum-Welch Algorithm, Bayesian Network, Hidden Markov Model, Viterbi Algorithm

MSC: 62-09; 62Hxx.

1 Introduction

The Hidden Markov model (HMM) is a statistical Markov model in which the system being modeled is assumed to be a Markov process with unobserved (hidden) states (Eddy , 1996; Rabiner , 1989). The HMMs are especially known for their applications in temporal pattern recognition such as speech (Bahl *et al.* , 1986; Cooke *et al.* , 1994), handwriting (Nogueiras *et al.* , 2001), gesture recognition (Arrowood , 2003), part-of-speech tagging (Bilmes , 2003), partial discharges (Byrne , 2006; Gales , 2007) and bioinformatics (Audhkhasi *et al.* , 2013; Selvaraj and Ganesan , 2014). The HMM can be presented as the simplest dynamic Bayesian network which is shown in Figure 1 (Ghahramani , 2001; Murphy and Mian , 1999; Pearl , 1998). A Bayesian network is a specific type of graphical model which is a directed acyclic graph (DAG). Bayesian networks encode the dependencies and independences among variables.

A conventional HMM is constructed by three sets of parameters which are: the initial state probabilities (π), transition (A) and emission (E) matrices. In any HMM, there are generally three problems to be considered (Rabiner , 1989):

- Evaluation: Given observation sequence $\mathbf{O} = \{O_1, \ldots, O_L\}$ and a model $\lambda = (A, E, \pi)$, how do we efficiently compute $P(\mathbf{O}|\lambda)$, i.e., the probability of the observation sequence given the model? For evaluation, two algorithms are used: the forward algorithm or the backward algorithm (Rabiner , 1989; Charniak , 1996).

- Recognition: Given an observation sequence $\mathbf{O} = \{O_1, \ldots, O_L\}$ and a model $\lambda = (A, E, \pi)$, how do we choose a corresponding state sequence $\mathbf{S} = \{S_1, \ldots, S_L\}$ which is optimal in some sense, i.e., best explains the observations? For this problem, the Viterbi algorithm is used (Viterbi , 1967; Forney , 1973).

- Training: Given the observation sequence $\mathbf{O} = \{O_1, \ldots, O_L\}$, how do we adjust the model parameters $\lambda = (A, E, \pi)$ to maximize $P(\mathbf{O}|\lambda)$? For this problem, the Baum-Welch algorithm is considered (Baum *et al.* , 1970).

The BaumWelch algorithm is a particular case of a generalized expectation-maximization (GEM) algorithm (Benyacoub *et al.* , 2015). It is used to find the unknown parameters of the HMM. This algorithm is an iterative procedure to estimate the model parameter λ. It works by maximizing the log-likelihood, and updating the current model. Each iteration of Baum-Welch is guaranteed to increase the log-likelihood of the data.

The Viterbi algorithm was introduced by Viterbi (1967) as a decoding algorithm for convolution codes over noisy digital communication links. It is a dynamic programming algorithm to find the most likely sequence of hidden states.

In the conventional HMM, given the current state, the current observation is independent of all other observations. In other words, in the process of Baum-Welch and Viterbi algorithms, it is assumed that, given the states, the observations are independent from each other and only the dependency between hidden states is considered. Some authors (Altman , 2007; Shannon *et al.* , 2013; Stanculescu *et al.* , 2014) introduced approaches in which the dependency among observations is combined with the conventional HMM. This extension of HMM is called the Autoregressive HMM (ARHMM).

In this paper, we consider the ARHMM with the assumption of dependencies among observations as a Bayesian network. This perspective makes it possible to use conditional independence relations for generalizing both Baum-Welch and Viterbi algorithms. In other words, we consider the ARHMM as a Directed Acyclic Graph (DAG) and use conditional independence for extending the conventional HMM. This means that we propose a new algorithm based on the Bayesian network rules. Figure 2 shows a special case of Bayesian network with first-order dependency (first-order Markov) in which the current observation is conditioned on the current state as well as the previous observation. For conditional independence relations in a Bayesian network, Pearl (1998) proposed a concept called D-separation. The D-separation is a graphical property of Bayesian networks and has the following implication: If two sets of nodes X and Y are D-separated in a Bayesian network by a third set Z (excluding X and Y), the corresponding variable sets X and Y are independent given the variables in

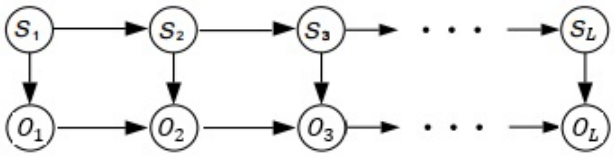

Figure 2: A Bayesian network representing a first-order Markov process between observations.

$O_{1,1}$	$O_{1,2}$	$O_{1,3}$
$O_{2,1}$	$O_{2,2}$	$O_{2,3}$
$O_{3,1}$	$O_{3,2}\uparrow$	$O_{3,3}$
$O_{4,1}$	$O_{4,2}$	$O_{4,3}$

Figure 3: An example of the neighbors of $O_{3,2}$.

Z ($X \perp Y|Z$). This means that nodes X and Y are D-separated in Bayesian networks by a third set Z (excluding X and Y) if and only if every path between X and Y is blocked. According to Figure 2 and the D-separation concept, we have:

$$S_t \perp \{S_1, O_1, \ldots, S_{t-2}, O_{t-2}, O_{t-1}\} | S_{t-1}, \quad O_t \perp \{S_1, O_1, \ldots, S_{t-1}, O_{t-2}\} | \{S_t, O_{t-1}\}, \quad 2 \leq t \leq L.$$

Based on these relations, the Baum-welch and Viterbi algorithms are generalized.

Rezaei *et al.* (2013) generalized the Baum-Welch and Viterbi algorithms in the bioinformatics field. They consider the profile Hidden Markov Model (PHMM) which is widely used to assign the protein sequences to their respective families. They use the multiple sequence alignment (MSA) which is a representative of a PHMM for consideration of the dependency between sequences. In other words, one-by-one dependency between corresponding amino acids of two current sequences (not between consecutive observations) is considered (Figure 3). In this paper, we focus on the direct dependency among consecutive observations (Figure 2).

This paper is organized as follows. Conditional independence relations in a Bayesian network are explained in Section 2. Based on the conditional independence rules, we generalize both Baum-Welch and Viterbi algorithms in Section 3. In Section 4, we compare the performance of the generalized algorithms with the commonly used

ones in a simulation study. We also apply these algorithms on real data sets in Section 5. Conclusions are presented in Section 6.

2 Conditional Independence Relations in Bayesian Networks

Probabilistic graphical models are graphs in which nodes represent random variables, and the (lack of) arcs represent conditional independence assumptions (Friedman *et al.*, 1999; Jensen , 1996; Pearl , 1998). Hence, they provide a compact representation of joint probability distributions. Undirected graphical models, also called Markov Random Fields (MRFs) or Markov networks, have a simple definition of independence: two (sets of) nodes A and B are conditionally independent given a third set, C, if all paths between the nodes in A and B are separated by a node in C. Directed acyclic graphical models called Bayesian networks or Belief networks, have a more complicated notion of independence, which take into account the direction of the arcs. Under the causal Markov assumption, each variable in a Bayesian network is independent of its ancestors given the values of its parents. With the causal Markov assumption, we can check some conditional independence in Bayesian networks. For the general conditional independence in a Bayesian network, Pearl (1998) proposed a concept of D-separation. The definition of D-separation is as follows: two sets of nodes X and Y are D-separated in Bayesian networks by a third set Z (excluding X and Y) if and only if every path between X and Y is blocked, where the term blocked means that there is an intermediate variable V (distinct from X and Y) such that:

- The connection through V is tail-to-tail or tail-to-head and V is instantiated.
- Or, the connection through V is head-to-head and neither V nor any of Vs descendants have received evidence.

The graph patterns of tail-to-tail, tail-to-head and head-to-head are shown in Figure 4. We consider the ARHMM as a Bayesian network and consider the following conditional independence relations:

$$S_t \perp \{S_1, O_1, \ldots, S_{t-2}, O_{t-1}\} | S_{t-1}, \qquad O_t \perp \{S_1, O_1, \ldots, S_{t-1}, O_{t-2}\} | \{S_t, O_{t-1}\}.$$

Based on these relations, the Baum-Welch and Viterbi algorithms are generalized. For this purpose, we need to define the joint probability distribution. The main advantage of Bayesian networks is that, given the representation of conditional independences by its structure and the set of local conditional distributions, the global joint probability

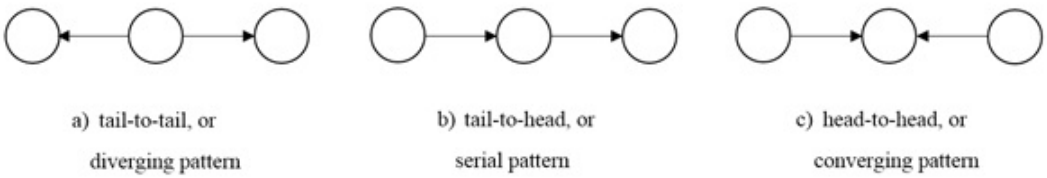

Figure 4: Patterns for paths through a node.

distribution of a Bayesian network is written as

$$P(O_1, \ldots, O_L) = \prod_{i=1}^{L} P(O_i | parent(O_i)), \tag{2.1}$$

where L is the number of variables and $\theta_i = P(O_i|parent(O_i))$ is specified by the parameters. Note that $parent(O_i)$ indicates the parents of O_i.

3 Generalized Baum-Welch and Viterbi Algorithms

In order to define the generalized Baum-Welch and Viterbi algorithms in the ARHMM, the following elements should be defined:

- A hidden state takes N values which will be denoted by $\{1, \ldots, N\}$.
- An observation takes M values which will be denoted by $\{1, \ldots, M\}$.
- There are L timestamps in the model, i.e., the set of latent variables of state $\mathbf{S} = \{S_1, \ldots, S_L\}$ and the set of observations $\mathbf{O} = \{O_1, \ldots, O_L\}$.
- A vector of the initial state π with elements $\pi(j) = P(S_1 = j)$, $1 \leq j \leq N$.
- The transition matrix A, in which the element $a_{ij} = P(S_t = j|S_{t-1} = i)$, $1 \leq i, j \leq N$, is the transition probability from state i to state j.
- The emission matrix E in which the element $e_j(k) = P(O_t = k|S_t = j)$, $1 \leq k \leq M$, is the emission probability of the current observed variable given the current hidden state.

- The emission matrix E′ in which the element $e_j(k,k') = P(O_t = k|S_t = j, O_{t-1} = k')$, $1 \leq k,k' \leq M$, is the emission probability of the current observed variable given the current hidden state and the previous observation.

Since the ARHMM is considered as a Bayesian network, the joint probability distribution of a sequence of states ($S_{1:L}$) and observations ($O_{1:L}$) is written as

$$\begin{aligned} P(O_{1:L}, S_{1:L}) &= P(S_1)P(O_1|S_1)\prod_{t=2}^{L} P(S_t|S_{t-1})P(O_t|S_t, O_{t-1}) \\ &= \pi(S_1)P(O_1|S_1)\prod_{t=2}^{L} a_{S_{t-1}S_t}.e_{St}(O_t, O_{t-1}), \end{aligned} \tag{3.1}$$

where $P(O_1|S_1)$ indicates the probability of an initial observation which depends only on the hidden state. Thus, we need to estimate a set of parameters $\lambda' = (A, E', \pi)$. To estimate the parameters λ', we use the Baum-Welch learning process. The Baum-Welch method is indeed an implementation of the general EM (Expectation-Maximization) method (Baum *et al.* , 1970). As indicated by its name, the EM algorithm involves a two-step (E-step and M-step) procedure which will be recursively used (Dempster *et al.* , 1977). Although the quantity $P(O_{1:L}, S_{1:L})$ can be calculated by the use of simple probabilistic arguments, it is not very practical because the calculation involves a number of operations. Fortunately, there is another calculation method with considerably low complexity that uses an auxiliary variable. Therefore, we define the new Forward and Backward algorithms which are to find out a recursive way to represent the variable sequence (Borodovsky and Ekisheva , 2006; Turner , 2008). The Forward algorithm represents the probability of partial observations up to time t and in the state i at time t, given the model λ':

$$\alpha_t(i) = P(O_1, \ldots, O_t, S_t = i|\lambda'),$$

then,

$$P(O_1, \ldots, O_L|\lambda') = \sum_{i=1}^{N} P(O_1, \ldots, O_L, S_L = i|\ \lambda') = \sum_{i=1}^{N} \alpha_L(i).$$

We can solve $\alpha_L(i)$ by using the Bayesian network rules through the equation

$$\begin{aligned}
\alpha_t(i) &= P(O_1,\ldots,O_t,S_t=i|\lambda') = \sum_{j=1}^{N} P(O_1,\ldots,O_t,S_t=i,S_{t-1}=j|\lambda') \\
&= \sum_{j=1}^{N} P(O_1,\ldots,O_{t-1},S_{t-1}=j)P(O_t,S_t=i|O_1,\ldots,O_{t-1},S_{t-1}=j) \\
&= \sum_{j=1}^{N} \alpha_{t-1}(j)P(O_t,S_t=i|O_1,\ldots,O_{t-1},S_{t-1}=j) \\
&= \sum_{j=1}^{N} \alpha_{t-1}(j) \\
&\times \frac{P(S_t=i|S_{t-1}=j)P(S_{t-1}=j)P(O_t|S_t=i,O_{t-1})P(O_{t-1}|O_{t-2},S_{t-1}=j)P(O_{t-2}|O_{t-3})\ldots}{P(S_{t-1}=j)P(O_{t-1}|O_{t-2},S_{t-1}=j)P(O_{t-2}|O_{t-3})\ldots} \\
&= \sum_{j=1}^{N} \alpha_{t-1}(j)P(S_t=i|S_{t-1}=j)P(O_t|S_t=i,O_{t-1}) = \sum_{j=1}^{N} \alpha_{t-1}(j)a_{ji}e_i(O_t,O_{t-1}), \qquad (3.2)
\end{aligned}$$

and $\alpha_1(i) = P(O_1,S_1=i) = \pi_i P(O_1|S_1=i)$. In a very similar manner, we define the new backward variable as follows:

$$\beta_t(i) = P(O_{t+1},O_{t+2},\ldots,O_L|S_t=i,O_t,\lambda'),$$

where $O_{t+1},O_{t+2},\ldots,O_L$ denote the partial time series beyond time t. We can use $\beta_t(i)$ to solve $P(O_{t+1},O_{t+2},\ldots,O_L|\lambda')$ by the following way:

$$\begin{aligned}
\beta_t(i) = P(O_{t+1},O_{t+2},\ldots,O_L|S_t=i,O_t,\lambda') &= \sum_{j=1}^{N} P(O_{t+1},O_{t+2},\ldots,O_L,S_{t+1}=j|S_t=i,O_t\lambda') \\
&= \sum_{j=1}^{N} P(O_{t+2},O_{t+2},\ldots,O_L|S_{t+1}=j,S_t=i,O_{t+1},O_t)P(S_{t+1}=j,O_{t+1}|S_t=i,O_t) \\
&= \sum_{j=1}^{N} P(O_{t+2},O_{t+2},\ldots,O_L|S_{t+1}=j,O_{t+1})P(O_{t+1}|S_{t+1}=j,S_t=i,O_t)P(S_{t+1}=j|S_t=i,O_t) \qquad (3.3) \\
&= \sum_{j=1}^{N} \beta_{t+1}(j)P(O_{t+1}|S_{t+1}=j,S_t=i,O_t)P(S_{t+1}=j|S_t=i,O_t) \\
&= \sum_{j=1}^{N} \beta_{t+1}(j)e_j(O_{t+1},O_t)a_{ij}.
\end{aligned}$$

According to (3.2) and (3.3), we can now compute the conditional probability of being in the state i at time t given the observation sequence by

$$P(S_t = i|O_1, \ldots, O_L) = \frac{\alpha_t(i)\beta_t(i)}{\sum_{i=1}^{N} \alpha_L(i)}.$$

Let $\xi_t(i, j)$ be the probability of the HMM being in the state i at time t, and making a transition to state j at time $t + 1$, given the model $\lambda' = (A, E', \pi)$ and observation sequence $(O_{1:L})$. Therefore,

$$\xi_t(i, j) = P(S_t = i, S_{t+1} = j|\mathbf{O}, \lambda').$$

By using Bayes law and the independence assumption, it follows that

$$\begin{aligned}
\xi_t(i, j) &= \frac{P(S_t = i, S_{t+1} = j, \mathbf{O}|\lambda')}{P(\mathbf{O}|\lambda')} \\
&= \frac{P(S_t = i, O_1, \ldots, O_t|\lambda')P(S_{t+1} = j, O_{t+1}, O_{t+2}, \ldots, O_L|S_t = i, O_t, \lambda')}{P(\mathbf{O}|\lambda')} \\
&= \frac{P(S_t = i, O_1, \ldots, O_t|\lambda')P(S_{t+1} = j|S_t = i)P(O_{t+1}, O_{t+2}, \ldots, O_L|S_{t+1} = j, S_t = i, O_t, \lambda')}{P(\mathbf{O}|\lambda')} \\
&= \frac{P(S_t = i, O_1, \ldots, O_t|\lambda')P(S_{t+1} = j|S_t = i)P(O_{t+1}|S_{t+1} = j, O_t, \lambda')P(O_{t+2}, \ldots, O_L|S_{t+1} = j, O_{t+1}, \lambda')}{P(\mathbf{O}|\lambda')} \\
&= \frac{\alpha_t(i)a_{ij}e_j(O_{t+1}, O_t)\beta_{t+1}(j)}{P(O_1)P(O_2|O_1)\ldots P(O_L|O_{L-1})}.
\end{aligned} \tag{3.4}$$

In (3.4), the probability $P(\mathbf{O}|\lambda) = P(O_1)P(O_2|O_1)\ldots P(O_L|O_{L-1})$ goes to 0 very fast with sequence length L. To address this issue, we use the equation

$$P(\mathbf{O}|\lambda) = \sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_t(i)a_{ij}e_j(O_{t+1}, O_t)\beta_{t+1}(j). \tag{3.5}$$

We also define $\gamma_t(i)$ as the probability of being in the state i at time t given the observation sequence $\mathbf{O}$ and model $\lambda' = (A, E', \pi)$, then it can be proven that

$$\begin{aligned}
\gamma_t(i) = P(S_t = i|\mathbf{O}, \lambda') &= \frac{P(S_t = i, \mathbf{O}|\lambda')}{P(\mathbf{O}|\lambda')} \\
&= \frac{P(S_t = i, O_1, \ldots, O_t|\lambda')P(O_{t+1}, O_{t+2}, \ldots, O_L|S_t = i, O_t, \lambda')}{P(\mathbf{O}|\lambda')} \\
&= \frac{\alpha_t(i)\beta_t(i)}{\sum_{i=1}^{N} \alpha_t(i)\beta_t(i)}.
\end{aligned} \tag{3.6}$$

The $\alpha_t(i)$ accounts for the partial observation sequence $O_1, \ldots, O_t$ and state i at t, while $\beta_t(i)$ accounts for the reminder of the observation sequence $O_{t+1}, \ldots, O_L$, given the state i at t. Therefore, the emission and transition probabilities will be as

$$\begin{aligned}\hat{e}_j(k,c) &= \frac{\sum_{\{t:O_t=k,O_{t-1}=c\}} \gamma_t(j)}{\sum_{\{t:O_{t-1}=c\}} \gamma_t(j)} = \frac{\textit{Expected number of times in state } j \textit{ and observing} \{O_t = k, O_{t-1} = c\}}{\textit{Expected number of times in } S_j},\\ \hat{a}_{ij} &= \frac{\sum_t \xi_t(i,j)}{\sum_t \gamma_t(i)} = \frac{\textit{Expected number of transitions from } S_i \textit{ to } S_j}{\textit{Expected number of transitions from } S_i}.\end{aligned} \tag{3.7}$$

We also generalize the Viterbi algorithm. In the generalized Viterbi algorithm, we should find the optimal state sequences which could best explain the given observations in some way according to dependency among observations. The solutions for this problem rely on the optimality criteria that we have chosen. The most widely used criterion is to maximize $P(\mathbf{O}, \mathbf{S}|\lambda')$. It represents the probability (for discrete distribution) or the likelihood (for continuous distribution) of observing a sequence given their joint distribution. The probability of the state path and observation sequence given the model is

$$P(\mathbf{O}, \mathbf{S}|\lambda') = P(\mathbf{O}|\mathbf{S}, \lambda')P(\mathbf{S}|\lambda') = \pi_{S_1} e_{S_1}(O_1) a_{S_1 S_2} e_{S_2}(O_2, O_1) \ldots a_{S_{L-1} S_L} e_{S_L}(O_L, O_{L-1}).$$

To convert the products into summations, we define a term $U(S)$ as

$$U(\mathbf{S}) = -\ln(P(\mathbf{O}, \mathbf{S}|\lambda') = -\Big[\ln(\pi_{S_1} e_{S_1}(O_1)) + \sum_{t=2}^{L} \ln\Big(a_{S_{t-1} S_t} e_{S_t}(O_t, O_{t-1})\Big)\Big].$$

Consequently,

$$\max_S P(\mathbf{O}, \mathbf{S}|\lambda') \leftrightarrow \min_S U(\mathbf{S}). \tag{3.8}$$

This reformation now enables us to view the terms $-\ln\Big(a_{S_{t-1} S_t} e_{S_t}(O_t, O_{t-1})\Big)$ as the cost (or distance) associated with the transition from state S_{t-1} to S_t, i.e., Let $U_t(S_1, \ldots, S_t)$ be the first t terms of $U(S)$ and $\delta_t(i)$ be the minimal accumulated cost when we are in the state i at time t, i.e.,

$$\begin{aligned}U_t(S_1, \ldots, S_t) &= -\Big[\ln(\pi_{S_1} e_{S_1}(O_1)) + \sum_{i=2}^{t} \ln\Big(a_{S_{i-1} S_i} e_{S_i}(O_i, O_{i-1})\Big)\Big].\\ \delta_t(i) &= \min_{S_1, S_2, \ldots, S_t = i} U_t(S_1, \ldots, S_{t-1}, S_t = i).\end{aligned} \tag{3.9}$$

Therefore, the generalized Viterbi algorithm can then be implemented in four steps:

- Initialize the $\delta_1(i)$ for all $1 \le i \le N$: $\delta_1(i) = -\ln\left(\pi_{S_i} e_{S_i}(O_i)\right)$;
- Inductively calculate the $\delta_1(i)$ for all $1 \le i \le N$ from time $t = 2$ to $t = L$: $\delta_t(i) = \min_{1 \le i \le N}\left[\delta_{t-1}(j) - \ln\left(a_{S_{i-1}S_i} e_{S_i}(O_i, O_{i-1})\right)\right]$;
- Then, we obtain the minimal vale of $U(S)$: $\min_S U(S) = \min_{1 \le i \le N}[\delta_L(i)]$;
- Finally, we trace back the calculation to find the optimal state path $\mathbf{S} = \{S_1, \ldots, S_L\}$.

It should be noted that the Viterbi algorithm is similar in implementation to the forward calculation (3.2).

4 Simulation Study

For the simulations, we have generated four different synthetic data sets of sizes 25×4 (i.e., 25 subjects over 4 days), 100×4, 200×4 and 500×4 of randomly sampled discrete data with 2 hidden states and 3 observation values using the following matrices:

$$P(S_t|S_{t-1}) = \begin{bmatrix} 0.7 & 0.3 \\ 0.9 & 0.1 \end{bmatrix}, \quad P(O_t|S_t) = \begin{bmatrix} 0.5 & 0.3 & 0.2 \\ 0.1 & 0.3 & 0.6 \end{bmatrix}, \quad \pi(S_1) = \pi(S_2) = 0.5.$$

For evaluation of the performance of the generalized and conventional Baum-Welch algorithms, we compute the Akaike information criterion (AIC) values at each iteration. The model which gives the minimum AIC is selected as the best model. The AIC rewards goodness of fit, but it also includes a penalty that is an increasing function of the number of estimated parameters as

$$AIC = 2K - 2\log(\mathit{Likelihood}). \tag{4.1}$$

Here, the likelihood is the probability of the data given a model and K is the number of estimated parameters. Convergence is generally detected by computing the value of the AIC after each iteration and halting when it appears not changing in a significant manner from one iteration to the next. The estimation process starts by giving initial values to the estimates. Good starting values are needed to find the optimal solution in a reasonable time. In order to reduce the risk of being trapped in a poor local maximum, a large number of initial values should be tested. Although we consider different initial values, this selection of values does not affect the estimates. Thus, we use the following initial values:

$$P(S_t|S_{t-1}) = \begin{bmatrix} 0.7 & 0.3 \\ 0.9 & 0.1 \end{bmatrix}, \quad P(O_t|S_t) = \begin{bmatrix} 0.5 & 0.3 & 0.2 \\ 0.1 & 0.3 & 0.6 \end{bmatrix}, \quad \pi(S_1) = \pi(S_2) = 0.5.$$

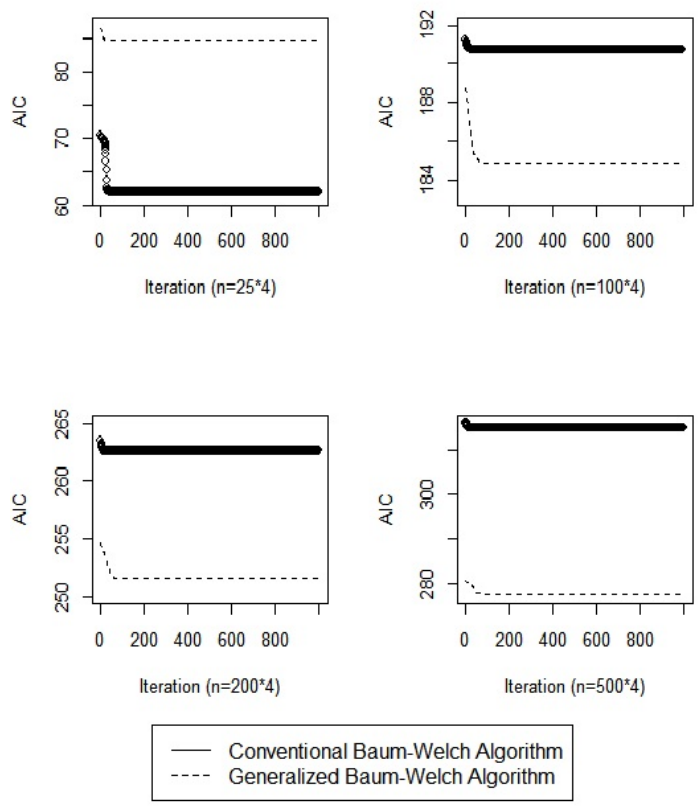

Figure 5: Comparing the performance of generalized and conventional Baum-Welch algorithms.

$$P(O_t|S_t, O_{t-1} = 1) = P(O_t|S_t, O_{t-1} = 2) = P(O_t|S_t, O_{t-1} = 3) = \begin{bmatrix} 0.4 & 0.2 & 0.4 \\ 0.1 & 0.4 & 0.5 \end{bmatrix}.$$

The results of using the generalized and conventional Baum-Welch algorithms are shown in Figure 5. For the first data set (25×4), the AIC values of the conventional Baum-Welch algorithm are less than the AIC values of the generalized Baum-Welch algorithm. When the sample size increases, the generalized Baum-Welch algorithm has more power than the conventional Baum-Welch algorithm. In other words, the generalized Baum-Welch algorithm is preferred to the conventional Baum-Welch algorithm when the sample is becoming larger.

It seems that the generalized Baum-Welch algorithm is a more complex model

than the conventional Baum-Welch algorithm. For both algorithms, we compute the time complexity. Using time complexity makes it easy to estimate the running time of an algorithm. The time complexity for both the generalized and the conventional Baum-Welch algorithm is reported in Table 1. As it is shown, the difference of consumed times in the process of the generalized and conventional Baum-Welch algorithms are not statistically significant. It should be noted that the algorithms have been implemented in Matlab.

Table 1: Comparing Time Complexity (in seconds).

Size of Datasets	Conventional Baum-Welch Algorithm	Generalized Baum-Welch Algorithm
25×4	11.23s	11.43s
100×4	36.01s	37.38s
200×4	100.20s	125.35s
500×4	192.25s	200.57s

The generalized and conventional Viterbi algorithms require knowledge of the parameters of the ARHMM and HMM. Thus, after estimating the emission and transition probabilities using the generalized and conventional Baum-Welch algorithms, we compare the results of the generalized and conventional Viterbi algorithms. For this purpose, for each sequence of simulated data sets, we obtain the most probable path under the generalized and conventional Viterbi algorithms. This is done by finding a maximum over all possible state sequences. We compute the logarithm of probability of most probable path for each sequence using the following equation:

$$\log\Big(\prod_{t=1}^{L} P(S_t|S_{t-1})\Big) = \sum_{t=1}^{L} \log\Big(P(S_t|S_{t-1})\Big). \tag{4.2}$$

The percentages of higher log-likelihood values and consumed times using the generalized Viterbi and conventional Viterbi algorithms are shown in Table 2. The results confirm that the generalized Viterbi algorithm has higher log-likelihood values than the conventional one. We also conclude that the difference of consumed times in the process of the generalized and conventional Viterbi algorithms are not statistically significant.

Table 2: The Percentage of Higher log-likelihood values (and time complexity) under generalized and conventional Viterbi algorithms.

Number of sequences	Conventional Viterbi Algorithm	Generalized Viterbi Algorithm
25	0.40 (8.1s)	0.60 (9s)
100	0.22 (25.2s)	0.78 (30.3s)
200	0.10 (90.4s)	0.90 (95.9s)
500	0.09 (186.7s)	0.91 (191.3s)

5 Real Data

In this section, the generalized Baum-Welch and Viterbi algorithms are applied to biological and inflation data sets.

5.1 Biological Data

In this section, we use the Pfam database which is a well-known data set of protein families (Finn *et al.* , 2016). It is widely used to align new protein sequences to the known proteins of a given family. There are two components in Pfam: Pfam-A and Pfam-B. The entries of Pfam-A have high quality. As shown in Table 3, we use the top twenty protein families of Pfam-A for assigning the protein sequences to protein families using the generalized and conventional Baum-Welch algorithms. We have three hidden states named; Match (M), Delete (D), and Insert (I), and 20 amino acids as observations $\{O_1, O_2, \ldots, O_{20}\}$.

To assess the performance of the generalized Baum-Welch algorithm, ten sequences from each of the top twenty families are randomly removed. Totally, we have 200 removed sequences which are used as test sequences, while the others form the training set. We repeat this procedure 10 times and based on the generalized and conventional Baum-Welch algorithms, we estimate the emission and transition matrices for training sets of each protein family. We then use the AIC to assign each test sequence to families. The means of the numbers of correctly assigned proteins into the top twenty protein families are shown in Table 4.

Based on the results, the assignment of sequences to the protein families using the generalized Baum-Welch algorithm is considerably improved. For all protein families, more than 70 percent of the removed sequences are assigned correctly using the generalized Baum-Welch algorithm.

Table 3: Top twenty protein families in pfam database.

profile	Number of sequence	
	Seed	Full
WD40	1465	378719
ABC_tran	55	369723
zf-C2H2	159	340711
pkinase	38	236455
MFS_1	192	214283
Response_reg	52	176760
Ank_2	203	172686
BPD_transp_1	81	148125
HATPase	658	133923
LRR_8	63	133230
RRM_1	72	131391
Helicase	422	119885
PPR_2	226	98670
Mito-carr	161	89340
fn_3	98	88510
AMP	145	87704
I-set	48	87027
adh	44	86592
PPR	459	85615
HisKA_1	265	85578

5.2 Inflation Data

In this section, the generalized Baum-Welch and Viterbi algorithms are applied to another data set which is related to the monthly change percentage of inflation (from 2001 to 2015) in Iran. The source of this data is the Central Bank of Iran. The rates of the monthly change percentage are between 0 to 6; so, we divide them into 3 intervals, each one with length two as follows: $[0, 2), [2, 4), [4, 6)$. We assign labels 1 to 3 to these intervals, respectively (O_1, O_2, O_3) . We also consider the hidden states for this data set as follows:

- Reducing bank interest rate (S_1);
- Increasing cash volume (S_2);
- Demand pressure (disrupting demand and supply in market (S_3).

These three states have been known as three effective factors on inflation. The inflation data set is a matrix with 30 rows and 6 columns. The first row represents the

Table 4: The mean of the numbers of correctly assigned sequences.

profile	Mean	
	Conventional Baum-Welch algorithm	Generalized Baum-Welch algorithm
WD40	7.5	8.8
ABC _tran	6.1	9.2
zf-C2H2	2.1	9.3
pkinase	3.3	8.6
MFS 1	8.4	8.9
Response _reg	7.9	8.9
Ank 2	6.7	9.7
BPD transp 1	7.9	8.1
HATPase	7.2	9.9
LRR_8	8.2	8.9
RRM_1	6.1	7.9
Helicase	8.4	8.9
PPR_2	8.1	8.8
Mito-carr	2.1	7.9
fn_3	8.2	8.5
AMP	6.8	8.5
I-set	5.9	7.5
adh	6.9	8.8
PPR	7.2	7.8
HisKA _1	7.5	8.6

rate of inflation in the first 6-months of 2001, the 2nd row the rate of inflation in the second 6-months of 2001 and so on. This means that the length of observations is $L = 6$. Starting with the initial value of the parameters as below, the generalized Baum-Welch algorithm is used: $A = P(S_t|S_{t-1}) = \begin{bmatrix} 0.2 & 0.3 & 0.5 \\ 0.4 & 0.3 & 0.3 \\ 0.5 & 0.2 & 0.3 \end{bmatrix}$, $P(O_t|S_t) = \begin{bmatrix} 0.3 & 0.3 & 0.4 \\ 0.3 & 0.4 & 0.3 \\ 0.4 & 0.3 & 0.3 \end{bmatrix}$, $\pi(S_1) = 0.3, P(O_1|S_1) = 0.3, P(O_t|S_t, O_{t-1} = 1) = P(O_t|S_t, O_{t-1} = 2) = P(O_t|S_t, O_{t-1} = 3) =$

$\begin{bmatrix} 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \end{bmatrix}$. The estimated transitions, and emission matrices are as follows:

$$\hat{A} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 01 \\ 0.9105 & 0 & 0.0895 \end{bmatrix}, \quad \hat{P}(O_t|S_t, O_{t-1} = 1) = \begin{bmatrix} 0.8387 & 0.1613 & 0 \\ 0.9373 & 0.0627 & 0 \\ 0.7861 & 0.2139 & 0 \end{bmatrix},$$

$$\hat{P}(O_t|S_t, O_{t-1} = 2) = \begin{bmatrix} 0.8387 & 0.1613 & 0 \\ 0.9373 & 0.0627 & 0 \\ 0.7861 & 0.2139 & 0 \end{bmatrix}, \quad \hat{P}(O_t|S_t, O_{t-1} = 3) = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \end{bmatrix}.$$

Also, the estimated transition and emission matrices using the conventional Baum-Welch algorithm are as follows:

$$\hat{A} = \begin{bmatrix} 0 & 0.3284 & 0.6716 \\ 0.3839 & 0.6161 & 0 \\ 0.4332 & 0 & 0.5668 \end{bmatrix}, \quad \hat{P}(O_t|S_t) = \begin{bmatrix} 0.9034 & 0.0966 & 0 \\ 0 & 0.8761 & 0.1239 \\ 1 & 0 & 0 \end{bmatrix}.$$

The AIC values in 1000 iterations, for the generalized Baum-Welch algorithm and the conventional Baum-Welch algorithm are shown in Figure 6. The AIC of the proposed algorithm is less than the conventional algorithm. Thus, the generalized Baum-Welch algorithm outperforms the conventional Baum-Welch algorithm.

After estimating the emission and transition probabilities using the generalized and the conventional Baum-Welch algorithms, we compare the generalized and conventional Viterbi algorithms by applying them to the inflation data set. Results show that the generalized Viterbi algorithm gives more accurate probable paths. For this purpose, we computed the log-likelihood values. The log-likelihood values obtained by the generalized Viterbi algorithm (4.2) in all sequences of observations are greater than the conventional Viterbi algorithm. This shows that the generalized Viterbi algorithm is more efficient.

6 Conclusion

The results presented in this paper show that considering a HMM which incorporates the dependency among consecutive observations will result in a notable improvement and has greater modeling power than the conventional HMM. We consider the ARHMM as a directed acyclic graph and use the conditional independence relations for generalizing the Baum-Welch and Viterbi algorithms. This model can be used for

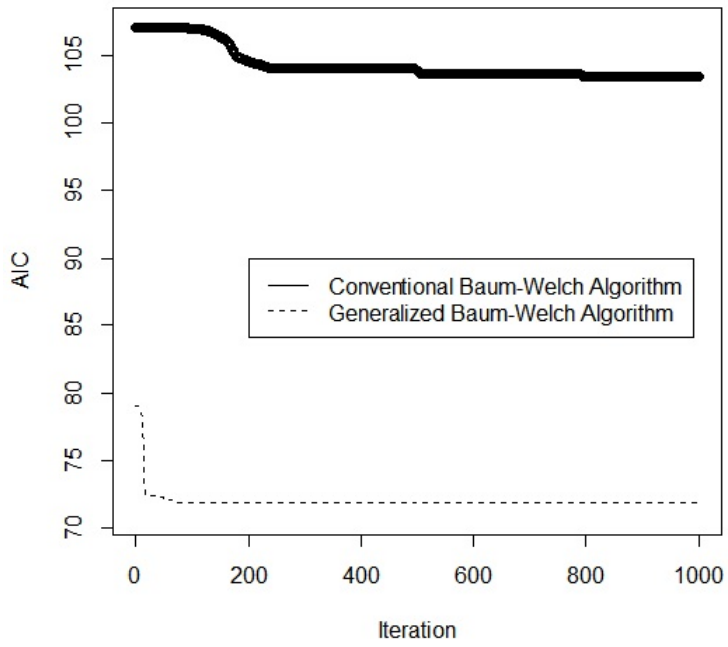

Figure 6: The iteration vs. AIC values.

sequence labeling problems such as part-of-speech (POS) tagging, text chunking in natural language processing, and structures of handwriting recognition. As an example, consider the situation where we tend to annotate a three-word sentence with part-of-speech tags. Therefore, the possible way of decomposition from left-to-right is likely to find the highest probability sequence of the desired three-word sentence.

Acknowledgements

We would like to thank two anonymous referees and an associate editor for their constructive comments and suggestions. This work has been supported by the Polish National Science Centre (2014/15/B/ST6/05082), Foundation for Polish Science (TEAM to DP) and by the grant from the Department of Science and Technology, India under Indo-

Polish/Polish-Indo project No.: DST/INT/POL/P-36/2016. The work was co-supported by grant 1U54DK107967-01 Nucleome Positioning System for Spatiotemporal Genome Organization and Regulation within 4DNucleome NIH program.

References

Altman, R .M. (2007), Mixed hidden Markov models: an extension of the hidden Markov model to the longitudinal data setting. *Journal of the American Statistical Association*, **102(477)**, 201–210.

Arrowood, Jon A. (2003), *Using Observation Uncertainty for Robust Speech Recognition*. (Doctoral dissertation, Georgia Institute of Technology).

Audhkhasi, K., Osoba, O. and Kosko, B. (2013), Noisy hidden Markov models for speech recognition. *IEEE International Joint Conference on Neural Networks (IJCNN)*, 1–6.

Bahl, L., Brown, P., De Souza, P. and Mercer, R. (1986), Maximum mutual information estimation of hidden Markov model parameters for speech recognition. *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP86*, **11**, 49–52.

Baum, L. E., Petrie, T., Soules, G. and Weiss, N. (1970), A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains. *The Annals of Mathematical Statistics*, **41(1)**, 164–171.

Benyacoub, B., ElMoudden, I., ElBernoussi, S., Zoglat, A. and Ouzineb, M. (2015), Initial model selection for the Baum-Welch algorithm applied to credit scoring. *Computation and Optimization in Information Systems and Management Sciences* (359–368). Springer, Cham.

Bilmes, J. A. (1998), A gentle tutorial of the EM algorithm and its application to parameter estimation for Gaussian mixture and hidden Markov models. *International Computer Science Institute*, **4(510)**, 126.

Bilmes, J. A. (2003), Buried Markov models: A graphical-modeling approach to automatic speech recognition. *Computer Speech and Language*, **17(2)**, 213–231.

Borodovsky, M. and Ekisheva, S. (2006), *Problems and Solutions in Biological Sequence Analysis*. Cambridge University Press.

Byrne, W. (2006), Minimum Bayes risk estimation and decoding in large vocabulary continuous speech recognition. *IEICE Transactions on Information and Systems*, **89(3)**, 900–907.

Charniak, E. (1996), *Statistical Language Learning*. MIT press.

Cooke, M., Green, P. D. and Crawford, M. (1994), Handling missing data in speech recognition. *In ICSLP*.

Dempster, A. P., Laird, N. M. and Rubin, D. B. (1977), Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, **39(1)**, 1–38.

Eddy, S. R. (1996), Hidden Markov models. *Current Opinion in Structural Biology*, **6(3)**, 361–365.

Finn, R. D., Coggill, P., Eberhardt, R. Y., Eddy, S. R., Mistry, J., Mitchell, A. L. and Salazar, G. A. (2016), The Pfam protein families database: towards a more sustainable future. *Nucleic Acids Research*, **44(D1)**, D279–D285.

Forney, G. D. (1973), The viterbi algorithm. *Proceedings of the IEEE*, **61(3)**, 268–278.

Friedman, N., Nachman, I. and Peér, D. (1999), Learning Bayesian network structure from massive data sets: the sparse candidate algorithm. *In Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence*, 206–215. Morgan Kaufmann Publishers Inc.

Gales, M. J. F. (2007), Discriminative models for speech recognition. *In 2007 Information Theory and Applications Workshop*, 170–176. IEEE.

Ghahramani, Z. (2001), An introduction to hidden Markov models and Bayesian networks. *International Journal of Pattern Recognition and Artificial Intelligence*, **15(01)**, 9–42.

Jensen, Finn V. (1996), *An Introduction to Bayesian Networks*. **Vol. 210.** London: UCL press.

Murphy, K. and Mian, S. (1999), *Modeling Gene Expression Data using Dynamic Bayesian Networks*, **(Vol. 104)**. Technical report, Computer Science Division, University of California, Berkeley, CA.

Nogueiras, A., Moreno, A., Bonafonte, A. and Mariño, J. B. (2001), Speech emotion recognition using hidden Markov models. *In INTERSPEECH*, 2679–2682.

Pearl, J. (1998), *Bayesian Networks*. Department of Statistics, UCLA.

Rabiner, L. and Juang, B. (1986), An introduction to hidden Markov models. *IEEE Magazine*, **3(1)**, 4–16.

Rabiner, L. R. (1989), A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, **77(2)**, 257–286.

Rezaei, V., Pezeshk, H. and Pérez-Sa'nchez, H. (2013), Generalized Baum-Welch algorithm based on the similarity between sequences. *PloS one*, **8(12)**, e80565.

Selvaraj, L. and Ganesan, B. (2014), Enhancing speech recognition using improved particle swarm optimization based hidden Markov model. *The Scientific World Journal*.

Shannon, M., Zen, H. and Byrne, W. (2013), Autoregressive models for statistical parametric speech synthesis. *IEEE Transactions on Audio, Speech, and Language Processing*, **21(3)**, 587–597.

Stanculescu, I., Williams, C. K. and Freer, Y. (2014), Autoregressive hidden Markov models for the early detection of neonatal sepsis. *IEEE Journal of Biomedical and Health Informatics*, **18(5)**, 1560–1570.

Turner, R. (2008), Direct maximization of the likelihood of a hidden Markov model. *Computational Statistics and Data Analysis*, **52(9)**, 4147–4160.

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