CONTENTS

Progress in Biological Sciences
Vol. 6, Number 2, Summer / Autumn 2016/117-127 – DOI: 10.22059/PBS.2016.590014

A generalization of Profile Hidden Markov Model (PHMM) using one-by-one dependency between sequences

Vahid Rezaei Tabar[1,2]*, Hamid Pezeshk[2,3]

[1] *Department of Statistics, Faculty of mathematics and Computer Sciences, Allameh Tabataba'i University, Tehran, Iranz*

[2] *School of Computer Science, Institute for Research in Fundamental Science (IPM), Tehran, Iran*

[3] *School of Mathematics, Statistics and Computer Science, University of Tehran, Iran*

Received: May 12, 2016; Accepted: December 10, 2016

ABSTRACT

The Profile Hidden Markov Model (PHMM) can be poor at capturing dependency between observations because of the statistical assumptions it makes. To overcome this limitation, the dependency between residues in a multiple sequence alignment (MSA) which is the representative of a PHMM can be combined with the PHMM. Based on the fact that sequences appearing in the final MSA are written based on their similarity; the one-by-one dependency between corresponding amino acids of two current sequences can be append to PHMM. This perspective makes it possible to consider a generalization of PHMM. For estimating the parameters of generalized PHMM (emission and transition probabilities), we introduce new forward and backward algorithms. The performance of generalized PHMM is discussed by applying it to the twenty protein families in Pfam database. Results show that the generalized PHMM significantly increases the accuracy of ordinary PHMM.

Keywords: Statistics; Multiple sequence alignment; Amino Acids; Protein families; Pfam database

Introduction

A 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 (1, 2, 3). It is used in almost all current speech recognition system, in numerous applications in computational molecular biology, in data compression, and in other areas of artificial and pattern recognition (4, 5, 6, 7). A Hidden Markov Model (HMM) can be presented as a specific type of graphical model which is a directed acyclic graph (DAG) (Figure 1). Under the casual Markov assumption, the joint probability distribution of a HMM can be written as:

$$P(O_{1:L}, S_{1:L}) = P(S_1) \prod_{t=1}^{L} P(S_t | S_{t-1}) P(O_t | S_t) \quad \{1\}$$

in which $P(S_t|S_{t-1})$ and $P(O_t|S_t)$indicate the transition and emission probabilities.

* **Corresponding author:** vhrezaei@atu.ac.ir

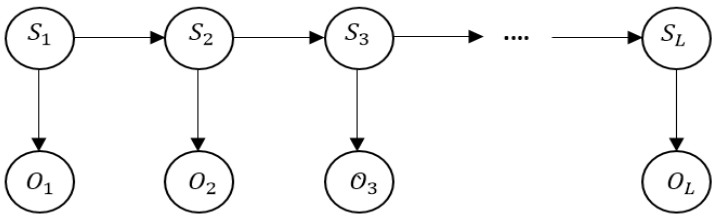

Figure 1. Structure of a HMM

Sonnhammer et al. (8) introduced an HMM architecture that was well suited for representing profiles of multiple sequence alignments (MSA). For each consensus column of the multiple alignment, a "Match" (M) state models the distribution of residues allowed in the column. An "Insert" (I) state and "Delete" (D) state at each column allow for insertion one or more residues between that column and the next, or for deleting the consensus residues. Profile HMMs are strongly linear, left-right models. Figure 2 shows a profile HMM corresponding to the MSA.

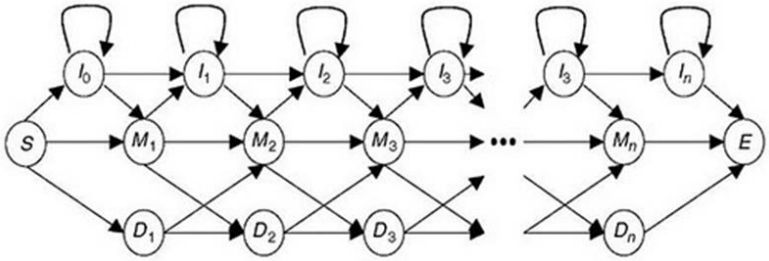

Figure 2. Structure of a profile HMM

The Profile Hidden Markov Model (PHMM) can be poor at capturing dependency between observations because of the statistical assumptions it makes (1). For overcoming this problem, we consider the one-by-one dependency between two current residues. Based on the fact that with doing a MSA, the sequences are biologically related, we can use the MSA to find the areas of similarity between two current sequences. Therefore the one-by-one dependency between a residue and the corresponding residue located above it can be combined with the PHMM (i.e. Figure 3).

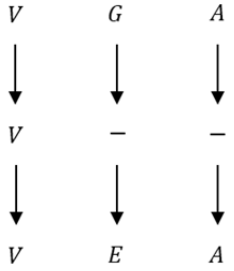

Figure 3. One-by-one dependency between sequences in MSA

This approach in spirit is similar to the works proposed by Holmes (9), Qian and Goldstein (10) and Siepel and Haussler (11) where a PHMM is augmented with phylogenetic trees. In their approach, the evolutionary information is appended to the PHMM. They considered the dependency between sequences based on the fact that all the current sequences are dependent upon their ancestral sequences and there is no dependency between two current sequences. But in our approach, the dependency between two current sequences based on the similarity between them can be appended to the PHMM.

When talking about PHMMs, there are generally three problems to be considered (12):

Evaluation: Given the observation sequence O={O_1,O_2,…,O_L} and a model λ, how do we efficiently compute P(O|λ), i.e., the probability of the observation sequence given the model. For evaluation, two algorithms are used: the forward algorithm or the backwards algorithm (1).

Recognition: Given the observation sequence O={O_1,O_2,…,O_L} and a model λ, how do we choose a corresponding state sequence S={S_1,S_2,…,S_L} which is optimal in some sense, i.e., best explains the observations. For this problem the Viterbi algorithm is used (13).

Training: Given the observation sequence O={O_1,O_2,…,O_L}, how do we adjust the model parameters λ to maximize P(O|λ). For this purpose the Baum Welch (forward-backward) algorithm is considered (14).

In this paper, based on the one-by-one dependency between two current sequences, we introduce the new forward and backward algorithms. As a result, the Baum-Welch and Viterbi algorithms are generalized.

This paper organizes as follows: in section 2, we introduce the PHMM. In section 3, parameter estimation of PHMM is presented. More details of generalized Viterbi algorithm presented in section 4. We finally compare the performance of the generalized PHMM with the common one by applying them on twenty protein families in Pfam database which is a well-known database of protein families (15).

Materials and Methods

The PHMM

Profile hidden Markov model (PHMM) techniques are among the most powerful methods for protein homology detection (16). One of the advantages of using the PHMMs is that they provide a better method for dealing with gaps found in protein families. A profile HMM is a linear state machine consisting of a series of nodes, each of which corresponds roughly to a position (column) in the alignment from which it was built (17-19). In other words, the PHMM is a linear structure of three states Match (M), Delete (D), and Insert (I). The construction of the PHMM is shown in Figure 2. In PHMM, we need to decide how many states exist in a PHMM. In other words, we should determine the length of the PHMM (i.e., how many match states do we have in a profile?). Here we assume that n is the number of Match states (M) in the PHMM. So, the total number of states is 3n+3.

Delete, Start and End states are silent and they emit no symbols. One heuristic method to set M, is to include those columns that have amino acids in at least half of the sequences using MSA (2). It should be noted that in each column, we have 20 amino acids or gap in which 20 amino acids are observed from Match and Insert states. A profile HMM has several types of probabilities associated with it. One type is the transition probability; the probability of transitioning from one state to another. There are also emissions probabilities associated with each Match or Insert state, based on the probability of a given residue existing at that position in the alignment.

Parameter Estimation of the Generalized PHMM

A major limitation of a PHMM is the assumption that given states, the observations, are independent. To overcome this limitation, the dependency between amino acids in a multiple sequence alignment (MSA) which is the representative of a PHMM can be appended to the PHMM. It is very important because we can generalize profile hidden Markov models using the on-by-one dependency. The sequences appearing in the final multiple sequence alignment are written based on their similarity (note that MSA is a representative of PHMM). So, the one-by-one dependency between corresponding amino acids of two current sequences can be combined with PHMM. Regarding MSA, we assume that protein sequences consisting of 21 observations (20 amino acids and one gap) have been placed on a regular lattice. In other words, each observation (residue) is arranged as a site (i.e. Figure 4).

V	G	A
V	$-$	$-$
V	E	A

Figure 4. Observations on a regular lattice

Regarding the regular lattice, we can introduce the ingredients of the PHMM as follows:

1. Hidden state (S) takes on 3n+3 values
2. Observation (O) takes on 21 values (20 amino acids and gap)
3. Transition probability matrix $\mathbf{A}_{(3n+3)\times(3n+3)}$ with following entries:

$a_i(j) = P(S_t = j | S_{t-1} = i), 1 \le i, j \le 3n+3$

4. Emission probabilities $\mathbf{B}_{(3n+3)\times 20}$ with the following entries:

$b_u(i,j) = P\left(O_{t,k-1} = i | S_t = u, O_{t,k} = j\right),$

$1 \le u \le 3n+1,\ 1 \le i \le 20,\ 1 \le j \le 21$.

This emission probability presents the probability of current observed variable given the current hidden state as well as observed variable located above it. It should be noted that $O_{t,k-1}$ represents the amino acids (20 types) at lattice point and $O_{t,k}$ is the amino acids or one gap (21 types) located above $O_{t,k-1}$.

5. A vector of initial state $\boldsymbol{\pi}$ with elements $\pi(i) = P(S_1 = i)$

With consideration of the one-by-one dependency between residues, a PHMM can also be considered as a graphical model.

According to Figure 3, if we assume that the observations come from three hidden Match states (M_1, M2 and M_3), then the dependency between Match states (from left to right), the dependency between residues (top to bottom), and the dependency between residues and hidden states can be shown in Figure 5.

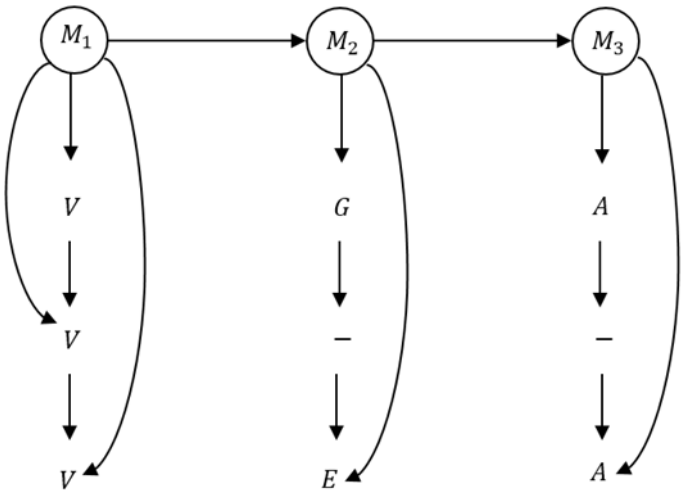

Figure 5. A PHMM with consideration of the one-by-one dependency

Suppose that $\mathbf{O}_t = \begin{bmatrix} O_{t,N} \\ O_{t,N-1} \\ \cdot \\ \cdot \\ \cdot \\ O_{t,1} \end{bmatrix}$, where N is the number of rows in MSA (Note that we arrange the observations on the regular lattice). For instance in Figure 4, we have $\mathbf{O}_1 = \begin{bmatrix} V \\ V \\ V \end{bmatrix}, \mathbf{O}_2 = \begin{bmatrix} G \\ - \\ E \end{bmatrix}, \mathbf{O}_3 = \begin{bmatrix} A \\ - \\ A \end{bmatrix}$.)

Regarding one-by-one dependency between observations, the likelihood of the parameters $\boldsymbol{\lambda} = (\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{\pi})$ given the observations will be as follows:

$$L(\boldsymbol{\lambda}|\boldsymbol{O}) = P(\boldsymbol{O}|\boldsymbol{\lambda}) = \sum_S P(\boldsymbol{O}|S, \boldsymbol{\lambda}) P(S|\boldsymbol{\lambda})$$
$$= \sum_S \prod_t P(\boldsymbol{O_t}|S_t) P(S_t|S_{t-1})$$

$$= \sum_{S} \prod_{t} \pi(S_t) \prod_{k=2}^{N} b_{S_t}(O_{t,k-1}, O_{t,k}). P(O_{t,N}|S_t). a_{S_{t-1}}(S_t)$$

Note that we consider $\varphi_{S_t}(O_{t,N}) = P(O_{t,N}|S_t)$ as the new parameter. In other words, a vector of initial observation $\boldsymbol{\varphi}$ with elements $\varphi_u(i) = P(O_{t,N} = i|S_t = u)$ is added to the set of parameters $\boldsymbol{\lambda}$. Taken together we have $\boldsymbol{\lambda} = (\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{\pi}, \boldsymbol{\varphi})$. For estimating the set of parameter $\boldsymbol{\lambda}$ in a PHMM, we need to define the new Forward and Backward algorithms which are to find out a recursive way to represent the variable sequence (20).

The Forward algorithm represents the probability of observations up to time t and in state *i* at time *t*, given the model λ;

$$\alpha_t(i) = P(\mathbf{O}_1, \boldsymbol{O}_2, \dots, \boldsymbol{O}_t, S_t = i|\boldsymbol{\lambda})$$

Then we have:

$$P(\boldsymbol{O_1}, \boldsymbol{O_2}, \dots, \boldsymbol{O_L}|\boldsymbol{\lambda}) = \sum_{i=1}^{3n+3} P(\boldsymbol{O_1}, \boldsymbol{O_2}, \dots, \boldsymbol{O_L}, S_L = i\ |\boldsymbol{\lambda}) = \sum_{i=1}^{3n+3} \alpha_L(i)$$

We can solve $\alpha_L(i)$ for inductively through the equation (Note that $\boldsymbol{O_t} \perp \boldsymbol{O_{1:t-1}}$):

$$\alpha_t(i) = P(\boldsymbol{O_1}, \boldsymbol{O_2}, \dots, \boldsymbol{O_t}, S_t = i\ |\boldsymbol{\lambda})$$

$$= \sum_{j=1}^{3n+3} P(\boldsymbol{O_1}, \boldsymbol{O_2}, \dots, \boldsymbol{O_t}, S_t = i, S_{t-1} = j|\boldsymbol{\lambda})$$

$$= \sum_{j=1}^{3n+3} P(\boldsymbol{O_1}, \boldsymbol{O_2}, \dots, \boldsymbol{O_{t-1}}, S_{t-1} = j\ |\boldsymbol{\lambda})\ P(\boldsymbol{O_t}, S_t = i|S_{t-1} = j, \boldsymbol{O_1}, \boldsymbol{O_2}, \dots, \boldsymbol{O_{t-1}}, \boldsymbol{\lambda})$$

$$= \sum_{j=1}^{3n+3} \alpha_{t-1}(j)\ P(\boldsymbol{O_t}, S_t = i|S_{t-1} = j, \boldsymbol{O_{1:t-1}}, \boldsymbol{\lambda})$$

$$= \sum_{j=1}^{3n+3} \alpha_{t-1}(j)\ P(\boldsymbol{O_t}, S_t = i|S_{t-1} = j)$$

$$= \sum_{j=1}^{3n+3} \alpha_{t-1}(j) * P(S_t = i|S_{t-1} = j) P(\boldsymbol{O_t}|S_t = i\ )$$

$$= \sum_{j=1}^{3n+3} \alpha_{t-1}(j) * a_j(i)\ P(O_{t,1}|S_t = i, O_{t,2})\ P(O_{t,2}|S_t = i, O_{t,3})\ P(O_{t,3}|S_t = i, O_{t,4}) \dots P(O_{t,N-1}|S_t = i, O_{t,N}) P(O_{t,N}|S_t)$$

$$= \sum_{j=1}^{3n+3} \alpha_{t-1}(j) * a_j(i)\ P(O_{t,N}|S_t = i) \prod_{k=2}^{N} P(O_{t,k-1}|S_t = i, O_{t,k})$$

$$= \sum_{j=1}^{3n+3} \alpha_{t-1}(j) * a_j(i)\ \varphi_i(O_{t,N}) \prod_{k=2}^{N} b_i(O_{t,k-1}, O_{t,k}) \qquad \{2\}$$

In a very similar manner, we define the backward variable as follows:

$$\beta_t(i) = P(\boldsymbol{O_{t+1}}, \boldsymbol{O_{t+2}}, \dots, \boldsymbol{O_L}|\ S_t = i, \boldsymbol{\lambda})$$

where $\boldsymbol{O_{t+1}}, \boldsymbol{O_{t+2}}, \dots, \boldsymbol{O_L}$ denote the partial time series beyond time *t* in a PHMM. Then we can use $\beta_t(i)$ to

solve $P(\boldsymbol{O_{t+1}}, \boldsymbol{O_{t+2}}, \dots, \boldsymbol{O_L}|\boldsymbol{\lambda})$ by the following way:

$$\beta_t(i) = P(\boldsymbol{O_{t+1}}, \boldsymbol{O_{t+2}}, \dots, \boldsymbol{O_L}|\, S_t = i, \boldsymbol{\lambda})$$

$$= \sum_{j=1}^{3n+3} P(\boldsymbol{O_{t+1}}, \boldsymbol{O_{t+2}}, \dots, \boldsymbol{O_L}, S_{t+1} = j|\, S_t = i, \boldsymbol{\lambda})$$

$$= \sum_{j=1}^{3n+3} P(\boldsymbol{O_{t+2}}, \boldsymbol{O_{t+3}}, \dots, \boldsymbol{O_L}|S_{t+1} = j, S_t = i\,, \boldsymbol{O_{t+1}}, \boldsymbol{\lambda})\, P(S_{t+1} = j\,, \boldsymbol{O_{t+1}}|S_t = i\,, \boldsymbol{\lambda})$$

$$= \sum_{j=1}^{3n+3} P(\boldsymbol{O_{t+2}}, \dots, \boldsymbol{O_L}|S_{t+1} = j\,)\, \frac{P(S_{t+1} = j, S_t = i\,, \boldsymbol{O_{t+1}})}{P(S_t = i,)}$$

$$= \sum_{j=1}^{3n+3} \beta_{t+1}(j)\, \frac{P(S_{t+1} = j|S_t = i)P(S_t = i)P(\,\boldsymbol{O_{t+1}}|\, S_{t+1} = j\,)}{P(S_t = i)}$$

$$= \sum_{j=1}^{3n+3} \beta_{t+1}(j)\, a_i(j)\; P\big(O_{t+1,N}\big|S_{t+1} = j\big) \prod_{k=2}^{N} P\big(O_{t+1,k-1}\big|S_{t+1} = j, O_{t+1,k}\big)$$

$$= \sum_{j=1}^{3n+3} \beta_{t+1}(j)\, a_i(j)\; \varphi_j\big(O_{t+1,N}\big) \prod_{k=2}^{N} b_j\big(O_{t+1,k-1}, O_{t+1,k}\big) \qquad \{3\}$$

Let $\xi_t(i,j)$ be the probability of the PHMM being in state *i* at time *t* and making a transition to state *j* at time *t* + 1, given the model $\boldsymbol{\lambda} = (\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{\pi}, \boldsymbol{\varphi})$ and observation sequence **O**:

$$\xi_t(i,j) = P(S_t = i, S_{t+1} = j|\boldsymbol{O}, \boldsymbol{\lambda})$$

Using Bayes law and the independency assumption, it follows:

$$\xi_t(i,j) = \frac{P(S_t = i, S_{t+1} = j, \boldsymbol{O}|\boldsymbol{\lambda})}{P(\boldsymbol{O}|\lambda)}$$

$$= \frac{P(S_t = i, \boldsymbol{O_1}, \boldsymbol{O_2}, \dots, \boldsymbol{O_t}\,|\boldsymbol{\lambda})P(S_{t+1} = j,\; \boldsymbol{O_{t+1}}, \boldsymbol{O_{t+2}}, \dots, \boldsymbol{O_L}|S_t = i, \boldsymbol{\lambda})}{P(\boldsymbol{O}|\lambda)}$$

$$= \frac{P(S_t = i, \boldsymbol{O_1}, \boldsymbol{O_2}, \dots, \boldsymbol{O_t}\,|\boldsymbol{\lambda})P(S_{t+1} = j|S_t = i)P(\boldsymbol{O_{t+1}}, \boldsymbol{O_{t+2}}, \dots, \boldsymbol{O_L}\,|S_{t+1} = j, S_t = i, \boldsymbol{\lambda})}{P(\boldsymbol{O}|\lambda)}$$

$$= \frac{P(S_t = i, \boldsymbol{O_1}, \boldsymbol{O_2}, \dots, \boldsymbol{O_t}\,|\boldsymbol{\lambda})P(S_{t+1} = j|S_t = i)P(\boldsymbol{O_{t+1}}|S_{t+1} = j, \lambda)P(\boldsymbol{O_{t+2}}, \dots, \boldsymbol{O_L}|S_{t+1} = j, \boldsymbol{\lambda})}{P(\boldsymbol{O}|\lambda)}$$

$$= \frac{\alpha_t(i)a_i(j)\,\varphi_j\big(O_{t+1,N}\big)\prod_{k=2}^{N} b_j\big(O_{t+1,k-1}, O_{t+1,k}\big)\,\beta_{t+1}(j)}{P(\boldsymbol{O_1}, \boldsymbol{O_2}, \dots, \boldsymbol{O_L}|\boldsymbol{\lambda})}$$

$$= \frac{\alpha_t(i)a_i(j)\,\varphi_j\big(O_{t+1,N}\big)\prod_{k=2}^{N} b_j\big(O_{t+1,k-1}, O_{t+1,k}\big)\,\beta_{t+1}(j)}{P\big(O_{1,1}, O_{1,2}, \dots, O_{1,N}\big) \dots P(O_{L,1}, O_{L,2}, \dots, O_{L,N})}$$

$$= \frac{\alpha_t(i)a_i(j)\,\varphi_j\big(O_{t+1,N}\big)\prod_{k=2}^{N} b_j\big(O_{t+1,k-1}, O_{t+1,k}\big)\,\beta_{t+1}(j)}{P\big(O_{1,1}|\,O_{1,2}\big)\; \dots P(O_{1,N-1}|O_{1,N})P(O_{1,N}) \dots P(O_{L,1}|\,O_{L,2})\; \dots P(O_{L,N-1}|O_{L,N})P(O_{L,N})} \qquad \{4\}$$

We also define the $\gamma_t(i)$ as the probability in state *i* at time t given the observation sequence **O**= $\{\boldsymbol{O_1}, \boldsymbol{O_2}, \dots, \boldsymbol{O_L}\}$ and model $\lambda = (\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{\pi}, \boldsymbol{\varphi})$ then it can be proven:

$$\gamma_t(i) = P(S_t = i|\boldsymbol{O}, \lambda) = \frac{P(S_t = i, \boldsymbol{O}|\boldsymbol{\lambda})}{P(\boldsymbol{O}|\boldsymbol{\lambda})}$$
$$= \frac{P(S_t = i, \boldsymbol{O_1}, \dots, \mathbf{O_t}|\boldsymbol{\lambda}) P(\boldsymbol{O_{t+1}}, \dots, \mathbf{O_L}|S_t = i, \boldsymbol{\lambda})}{P(\boldsymbol{O}|\boldsymbol{\lambda})}$$
$$= \frac{\alpha_t(i)\beta_t(i)}{P(O_{1,1}|\, O_{1,2}) \ \dots P(O_{1,N-1}|O_{1,N}) P(O_{1,N}) \dots P(O_{L,1}|\, O_{L,2}) \ \dots P(O_{L,N-1}|O_{L,N}) P(O_{L,N})} \qquad \{5\}$$

Baum-Welch method is indeed an implementation of general EM (Expectation-Maximization) method (14). As indicated by its name, EM algorithm involves a two-step (E-step and M-step) procedure which will be recursively used (4). Baum-Welch works by maximizing a proxy to the log likelihood, and updating the current model to be closer to the optimal model. Each iteration of Baum-Welch is guaranteed to increase the log-likelihood of the data. In this paper Generalized Baum-Welch works in the following way for each sequence in the training set of sequences:

1. Calculate forward probabilities with the forward algorithm
2. Calculate backward probabilities with the backward algorithm
3. Calculate the contributions of the current sequence to the transitions of the model, calculate the contributions of the current sequence to the emission probabilities of the model.
4. Calculate the new model parameters (start probabilities, transition probabilities, emission probabilities)
5. Calculate the new log likelihood of the model
6. Stop when the change in log likelihood is smaller than a given threshold or when a maximum number of iterations is passed.

Therefore the estimated emission and transition probabilities will be as follows (2):

$$\hat{b}_u(i,j) = \frac{\sum_{\{t:O_{t,k-1}=i, O_{t,k}=j\}} \gamma_t(u)}{\sum_{\{t:O_{t,k}=j\}} \gamma_t(u)} \ , \forall k = 2, \dots, n$$

$$\hat{a}_i(j) = \frac{\sum_t \xi_t(i,j)}{\sum_t \gamma_t(i)}$$

Generalization of Viterbi Algorithm

One of the most important problems in a hidden Markov model (HMM) is, given observations $\boldsymbol{O} = \{O_1, O_2, \dots, O_L\}$ and the model $\boldsymbol{\lambda}$, how we choose the states $\boldsymbol{S} = \{S_1, S_2, \dots S_L\}$ from 3n+3 possible states to maximize the probability of observing the sequence? (12). The Viterbi algorithm finds the single best state path for the given observations. In other words, the Viterbi algorithm provides overall most likely path.

In generalized Viterbi algorithm, we have to find the optimal state sequences which could best explain the given observations according to dependency between observations. The solutions for this problem rely on the optimality criteria we have chosen. The most widely used criterion is to maximize $P(\boldsymbol{O}, \boldsymbol{S}|\boldsymbol{\lambda})$. It represents the probability (for discrete distribution) or likelihood (for continuous distribution) of observing observation sequence given their joint distribution. Therefore the probability of the state path and observation sequence given the model in a PHMM would be as follows:

$$P(\boldsymbol{O}, \boldsymbol{S}|\boldsymbol{\lambda}) = P(\boldsymbol{O}|\boldsymbol{S}, \boldsymbol{\lambda}) P(\boldsymbol{S}|\boldsymbol{\lambda}) \ = \pi_{S_1} \, b_{S_1}(\boldsymbol{O_1}) a_{S_1}(S_2) \, b_{S_2}(\boldsymbol{O_2}) \dots a_{S_{L-1}}(S_L) b_{S_L}(\boldsymbol{O_L})$$

Where

$$b_{S_t}(\boldsymbol{O_t}) = P(\boldsymbol{O_t}|S_t) = P(O_{t,N}|S_t) \prod_{k=2}^{n} P(O_{t,k-1}|S_t, O_{t,k}) \ = \varphi_{S_t}(O_{t,N}) \prod_{k=2}^{n} b_{S_t}(O_{t,k-1}, O_{t,k})$$

To convert the products into summations, U(**S**) defines as follows:

$$U(\boldsymbol{S}) = -\ln\big(P(\boldsymbol{O},\boldsymbol{S}|\lambda)\big) = -\left[\ln\big(\pi_{S_1}\big) + \sum_{t=1}^{L} \ln\left(a_{S_{t-1}}(S_t) b_{S_t}(\boldsymbol{O_t})\right)\right] \quad \{6\}$$

Consequently,

$$\max_S P(\boldsymbol{O},\boldsymbol{S}|\lambda) \leftrightarrow \min_S U(\boldsymbol{S})$$

This reformation now enables us to view terms -ln($a_{S_{t-1}}(S_t) b_{S_t}(\boldsymbol{O_t})$) as the cost (or distance). The problem then can be seen as finding the shortest path via Viterbi Algorithm.

Let $U_t(S_1, \dots S_t)$ be the first *t* terms of *U*(**S**) and $\delta_t(i)$ be the minimal accumulated cost when we are in state *i* at time *t*,-

$$U_t(S_1, \dots S_t) = -[\ln\big(\pi_{S_1}\big) + \sum_{i=1}^{t} \ln(a_{S_{i-1}}(S_i) b_{S_i}(\boldsymbol{O_i}))]$$
$$\delta_t(i) = \min_{S_1, S_2, \dots S_t = i} U_t(S_1, \dots S_{t-1}, S_t = i)$$

Therefore, Viterbi algorithm then can be implemented by four steps:

1. Initialize the $\delta_1(i)$ for all $1 \le i \le 3n+3$;

$$\delta_1(i) = -\ln\big(\pi_{S_i}\big)$$

2. Inductively calculate the $\delta_t(i)$ for all $1 \le i \le 3n+3$ from time *t* = 1 to *t* = L:

$$\delta_t(i) = \min_{1 \le j \le 3n+3} [\delta_{t-1}(j) - \ln(a_{S_j}(S_i) b_{S_i}(\boldsymbol{O_i})]$$

3. Then we get the minimal vale of *U*(**S**):

$$\min_S U(\boldsymbol{S}) = \min_{1 \le i \le 3n+3} [\delta_L(i)]$$

4. Finally we trace back the calculation to find the optimal state path $\boldsymbol{S} = \{S_1, \dots S_L\}$

Results and Discussion

To evaluate the performance of generalized profile hidden Markov model, we use the top twenty protein families from the Pfam database (Table 1) which is a well-known database of protein families (15). Proteins are generally composed of one or more functional regions, commonly termed domains. Different combinations of domains give rise to the diverse range of proteins found in nature. The identification of domains that occur within proteins can therefore provide insights into their function. The Pfam database contains 16230 families. Pfam is a database of protein families that includes their annotations and multiple sequence alignments generated using hidden Markov models.

Table 1. Top Twenty protein families in Pfam database

profile	Number of sequence	
	Seed	Full
ABC_tran	60	163029
RVT 1	155	126258
COX1	94	118265
GP120	24	105452
WD40	1842	101999
RVP	50	93675
zf-C2H2	195	88330
Response_reg	57	75322
Cytochorm B N	92	70463
HA TPase c	662	70410
BPD transp 1	81	70027
MFS_1	196	69503
Oxidored q1	33	60333
Pkinase	54	56691
Cytochrom_B_C	114	51006
RVT_thumb	41	50191
Adh short	230	50144
Acetyltransf 1	243	46279
Helicase_C	491	42435
HTH_1	1556	41545

There are two components in Pfam: Pfam-A and Pfam-B. The entries of the pfam-A have high quality and these twenty protein families belong to Pfam A. Pfam-A is the manually curated portion of the database. For each entry a protein sequence alignment and a hidden Markov model is stored. These hidden Markov models can be used to search sequence databases with the HMMER package written by Sean Eddy (2).

To assess the performance of the generalized PHMM, 20 sequences from each family are randomly removed. So we have 400 sequences removed in total. We consider these data as TEST sequences while the other sequences form the training set. Because of computational problem, we only repeat this procedure10 times.

Given the training sequences of twenty protein families, the transition matrix $\boldsymbol{A}_{(3n+3)\times(3n+3)}$ and the emission matrix $\boldsymbol{B}_{(3n+3)\times 20}$ are estimated using generalized and common Baum-Welch algorithms. Then, each removed sequences (a sequence of TEST data) is returned to all families (not only that family which has been removed from). The log-likelihood value of each test sequence for all protein families is computed. Then the numbers of correctly assigned test sequences to the twenty protein families are counted (Table 2).

Table 2. The average numbers of correctly assigned sequences

Profiles	Common Baum-Welch	Generalized Baum-Welch
ABC_tran	10.6	18.7
RVT 1	14.3	18.8
COX1	12.5	16.0
GP120	18.3	18.3
WD40	14.0	16.2
RVP	12.0	18.7
zf-C2H2	6.3	18.9
Response_reg	14.8	16.3
Cytochorm B N	14.9	16.2
HA TPase c	14.0	18.4
BPD transp 1	14.0	16.4
MFS_1	16.5	16.1
Oxidored q1	16.3	16.9
Pkinase	6.1	14.4
Cytochrom_B_C	14.4	16.8
RVT_thumb	12.2	18.1
Adh short	10.6	14.6
Acetyltransf 1	12.7	18.1
Helicase_C	14.1	14.0
HTH_1	5.3	14.0

Result show that assignment of protein sequences to protein families under the generalized Baum-Welch algorithm have higher accuracy than common Baum-Welch algorithm. Since the profile hidden Markov model finds local optima, it is important to choose

initial parameters carefully. We perform the algorithm with different initial parameters in a way that the transition probabilities into Match states are larger than transition probabilities into other states.

We also use the generalized Viterbi algorithm for determining the most probable path for each test sequence in corresponding protein family. Fort this purpose we use the following equation:

$$P(\boldsymbol{S}|\lambda) = \pi_{S_1}\, a_{S_1}(S_2) \ldots a_{S_{L-1}}(S_L)$$

or

$$\ln P(\boldsymbol{S}|\lambda) = \ln(\pi_{S_1}) + \sum_{i=1}^{L} \ln(a_{S_{i-1}}(S_i)) \quad \{7\}$$

Equation 7 indicates the log-likelihood values of the optimal (most likely) sequence of hidden states for a test sequence. We calculate this score for all test sequences into each family using generalized and common Viterbi algorithm. We then normalized the log-likelihood values (scores) into each family. Figure 6 indicates that the average of normalized score for most probable path into each protein family obtained by considering the one-by-one dependency are higher than common Viterbi algorithm.

REFERENCES

1. Rabiner, L.R., and Biing-Hwang, J. (1986) An introduction to hidden Markov models. *ASSP Mag., IEEE*, **3**, 4-16.
2. Eddy, S.R. (1998) Profile hidden Markov models. *Bioinformatics*, **14**, 755-763.
3. Dymarski, P. (2011) *Hidden Markov Models, theory and applications*. InTech Open Access Publishers.
4. Bilmes, J.A. (2003) Buried Markov models: A graphical-modeling approach to automatic speech recognition. *Comput. Speech Lang.*, **17**, 213-231.
5. Bahl, L.R., Brown, P.F., de Souza, P.V. and Mercer, R.L. (1986) Maximum mutual information estimation of hidden Markov model parameters for speech recognition. *Proc. ICASSP* 86, Pp. 49-52.
6. Selvaraj, L., and Ganesan, B. (2014) Enhancing speech recognition using improved particle swarm optimization based Hidden Markov Model. *Scientific World J*. DOI: 10.1155/2014/270576
7. Shannon, M., Heiga Zen, H., and Byrne, W. (2013) Autoregressive models for statistical parametric speech synthesis. *Audio, Speech, Lang. Proc., IEEE Trans.*, **21**, 587-597.
8. Sonnhammer, E.L.L, von Heijne, G. and Krogh, A. (1998) A hidden Markov model for predicting transmembrane helices in protein sequences. *ISMB 98 Proceedings*, *Pp*. 1-8.
9. Holmes, I. (2003) Using guide trees to construct multiple-sequence evolutionary HMMs. *Bioinformatics*, **19** (suppl. 1), i147-i157.
10. Qian, B. and Goldstein, R.A. (2004) Performance of an iterated T-HMM for homology detection. *Bioinformatics*, **20**, 2175-2180.
11. Siepel, A., and Haussler, D. (2004) Combining phylogenetic and hidden Markov models in biosequence analysis. *J.Comput. Biol.*, **11**, 413-428.
12. Rabiner, L.R. (1989) A tutorial on Hidden Markov Models and selected applications in speech recognition. *Proc. IEEE*, **77**, 257-285.
13. Viterbi, A.J. (1967) Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, **13**, 260-269.
14. 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. *Ann. Math. Statist.*, **41**, 164-171.

15. Finn, R.D., Coggill, P., Eberhardt, R.Y, Eddy, S.R., Mistry, J., Mitchell, A.L., Potter, S.C., Punta, M., Qureshi, M., Sangrador-Vegas, A., Salazar, G.A., Tate, J., and Bateman, A. (2016) The pfam protein families database. *Nucleic Acids Res.*, **44** (Database issue), D279-D285.
16. Gribskov, M., McLachlan, A.D. and Eisenberg, D. (1987) Profile analysis: detection of distantly related proteins. *Proc.Nat. Acad. Sci.*, **84**, 4355-4358.
17. Maor, A. and Doron Shaked, D. (2016) *User behavior recovery via Hidden Markov Models Analysis*. Hewlett Packard Enterprise Development LP. Pp. 1-6.
18. Wheeler, T.J., Jody Clements, J., Eddy, S.R., Hubley, R., Jone, T.A., Jurka, J., Smit, A.F.A., and Finn, R.D. (2012) Dfam: a database of repetitive DNA based on profile hidden Markov models. *Nucleic Acids Res.*, **41** (Database issue), D70-D82.
19. Wheeler, T.J., Clements, J., and Finn, R.D. (2014) Skylign: a tool for creating informative, interactive logos representing sequence alignments and profile hidden Markov models. *BMC Bioinformatics*, **15**, 7.
20. Petrushin, V.A. (2000) Hidden Markov Models: Fundamentals and applications- Part 1: Markov chains and mixture models. Online Symposium for Electronics Engineer [online], Available from: http://www.techonline.com/asee/ [accessed April 25 2005].