Sequence Clustering with the Self-Organizing Hidden Markov Model Map

Christos Ferles, and Andreas Stafylopatis

***Abstract*— A hybrid approach combining the Self-Organizing Map (SOM) and the Hidden Markov Model (HMM) is presented. The Self-Organizing Hidden Markov Model Map (SOHMMM) establishes a cross-section between the theoretic foundations and algorithmic realizations of its constituents. The respective architectures and learning methodologies are blended together in an attempt to meet the increasing requirements imposed by the deoxyribonucleic acid (DNA), ribonucleic acid (RNA), and protein chain molecules. Addressing many of the most intriguing biological sequence analysis problems is achieved through its automatic raw sequence data learning mechanism. Since the SOHMMM carries out probabilistic sequence analysis with little or no prior knowledge, it can have a variety of applications in clustering, dimensionality reduction and visualization of large-scale sequence spaces, and also, in sequence discrimination, search and classification. A comprehensive series of experiments based on the globin protein family demonstrates SOHMMM's sophisticated characteristics and advanced capabilities.**

I. Introduction

THE advent of novel and efficient experimental technologies, primarily genome sequencing, microarrays and mass spectrometry, has led to an exponential growth of linear descriptions of protein, deoxyribonucleic acid (DNA) and ribonucleic acid (RNA) chain molecules requiring automated analysis. Altogether, these high-throughput technologies are capable of rapidly producing terabytes of data that are too overwhelming for conventional biological approaches. As a response scientists use algorithms, statistics, and other mathematical techniques to decipher the language of DNA [1].

Conventional computer science algorithms and trite statistical techniques have been useful, but are increasingly unable to address many of the most interesting sequence analysis problems. This is due to the inherent complexity of biological systems, brought about by biological tinkering, and also, due to our lack of comprehensive theory of life's organization and function at the molecular level. Neural Networks (NNs), Hidden Markov Models (HMMs), Bayesian networks and similar machine learning approaches, on the other hand, are ideally suited for domains characterized by the presence of large amounts of data, complex structures and the absence of general theories [2]. The fundamental idea behind these approaches is to learn the theory automatically from the data through a process of inference, parameter adaptation, model fitting, or learning from examples.

In order to overcome the limitations of HMMs, attempts have been made for combining HMMs and NNs to form hybrid models that contain the expressive power of artificial NNs with the sequential time series aspect of HMMs [3]. A type of (labeled) clustering has been achieved by training several HMMs (components) in parallel and using some form of competitive/unsupervised learning to construct a composite HMM [4]. According to this approach the Baum-Welch learning algorithm [5] has been used in its purest form to automatically partition the sequences of a single protein family into clusters (subfamilies) of similar sequences.

The Self-Organizing Hidden Markov Model Map (SOHMMM) is the offspring of a crossover between the Self-Organizing Map (SOM) algorithm [6], [7] and the HMM theory [8], [2]. The SOHMMM's core consists of a novel unified/hybrid SOM-HMM algorithm. Both components' corresponding architectures are intimately fused. The model is coupled with a raw sequence data training method, that blends together the SOM unsupervised learning and the HMM dynamic programming algorithms. The ultimate objective is to merge and strengthen the advantages of the two algorithms that constitute the SOHMMM, while, at the same time, minimizing and going beyond any possible drawbacks. Epigrammatically, the SOHMMM approach: is based on a very rich probabilistic framework; proves ideal for analyzing sequences derived from chain molecules; integrates the clustering, dimensionality reduction and visualization disciplines in a unified scheme; provides procedures for sequence discrimination, search and classification; covers an extended set of distributions which represents the input sequence space in a more faithful manner.

II. Background and Prerequisites

A. Hidden Markov Model

The present section is based on the approach in [8]. Consider an alphabet $S = \{s_1, s_2, \ldots, s_N\}$ and a sequence of random variables q_t, $t \in \mathbb{N}^*$ assuming values in S. S is usually called the state space whereas the symbols s_n in the alphabet are called states. Let the sequence of random

Manuscript received June 26, 2008; revised August 12, 2008.

C. Ferles, and A. Stafylopatis are with the Intelligent Systems Laboratory, School of Electrical and Computer Engineering, National Technical University of Athens, 15780 Zografou, Athens, Greece (e-mail: christos.ferles@gmail.com; andreas@cs.ntua.gr).

variables $\{q_t\}_{t=1}^{\infty}$ be a Markov chain. The conditional probabilities (for $t>1$)

$$a_{ij} = P(q_t = s_j \mid q_{t-1} = s_i),\ 1 \le i \le N,\ 1 \le j \le N \quad (1)$$

are assumed to be independent of t, are called (stationary) one step transition probabilities, and they obey standard stochastic constraints. Such a Markov chain with stationary transition probabilities is called homogeneous. Consequently a transition matrix $A = \{a_{ij}\}$ with these properties is called a stochastic matrix. At time $t = 1$ the state q_1 is specified by the initial state probability distribution $\pi = \{\pi_j\}$ where

$$\pi_j = P(q_1 = s_j),\ 1 \le j \le N. \quad (2)$$

Let $\{Y_t\}_{t=1}^{\infty}$ be a random process with a finite state space $V = \{v_1, v_2, \ldots, v_M\}$, where M need not equal N. The processes $\{q_t\}_{t=1}^{\infty}$ and $\{Y_t\}_{t=1}^{\infty}$ are for any $t \ge 1$ related by the conditional probability distributions

$$b_j(k) = P(Y_t = v_k \mid q_t = s_j),\ 1 \le j \le N,\ 1 \le k \le M. \quad (3)$$

We set $B = \{b_j(k)\}$ and we shall call this the emission probability matrix. This is also another stochastic matrix.

Suppose $O = o_1 o_2 \ldots o_T$ is an observation sequence where each observation o_t assumes a value from V, and T is the number of observations in the sequence. We obtain the joint probability of O as a marginal distribution by summing over all possible paths of the state sequence. Thus

$$P(O \mid A, B, \pi) = \sum_{q_1, q_2, \ldots, q_T} \pi_{q_1} b_{q_1}(o_1) a_{q_1 q_2} b_{q_2}(o_2) \ldots a_{q_{T-1} q_T} b_{q_T}(o_T). \quad (4)$$

It should be noted that, in certain cases, the indexes of the transition, initial and emission probabilities are denoted as q_t and/or o_t, and not as i and/or j and/or k. Such an approach is followed whenever the exact states and/or exact observation symbols are insignificant for the formulation/calculation under consideration, even though these values are considered to be given and specific.

Consequently, a complete specification of a HMM requires specification of the cardinalities of the two state spaces (namely N and M), specification of the observation symbols, specification of the stochastic matrices A and B, and specification of the initial probability distribution π. Henceforth, we may use the compact notation for the model

$$\lambda = (A, B, \pi). \quad (5)$$

A procedure that facilitates all computational thinking with a HMM is known as the forward-backward algorithm. Consider the forward variable $\alpha_t(i)$ as the joint probability of the observation sequence up to time $t \le T$ and of the hidden Markov chain being in state s_i at time t, given the model λ

$$\alpha_t(i) = P(o_1 o_2 \ldots o_t,\ q_t = s_i \mid \lambda). \quad (6)$$

Also, consider the backward variable $\beta_t(i)$ as the probability of the observation sequence from time $t+1$ till the end T conditioned on the hidden Markov chain being in the state s_i at time t, given the model λ

$$\beta_t(i) = P(o_{t+1} o_{t+2} \ldots o_T \mid q_t = s_i,\ \lambda). \quad (7)$$

A recursive solution for $\alpha_t(i)$ and $\beta_t(i)$ is the following:

$$\alpha_1(i) = \pi_i b_i(o_1),\ 1 \le i \le N \quad (8)$$

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij} \right] b_j(o_{t+1}),\ 1 \le t \le T-1,\ 1 \le j \le N \quad (9)$$

$$\beta_T(i) = 1,\ 1 \le i \le N \quad (10)$$

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(o_{t+1}) \beta_{t+1}(j),\ t = T-1, T-2, \ldots, 1,\ 1 \le i \le N \quad (11)$$

The forward-backward algorithm presented above allows the evaluation of the probability (4), so that the computational requirement is linear to the sequence length

$$P(O \mid \lambda) = \sum_{j=1}^{N} \alpha_t(j) \beta_t(j). \quad (12)$$

B. On-line Gradient Descent Algorithm for HMMs

In the present study, we are interested in parameter estimation, that is, finding the best possible HMM $\lambda = \lambda(x)$ that minimizes the posterior $-\log P(\lambda(x) \mid O)$ or possibly the likelihood $f(x) = -\log P(O \mid \lambda(x))$. Whenever a function $f(x)$ is differentiable, one can try to find its minima by using one of the oldest optimization algorithms, gradient descent [9], [10]. Gradient descent is an iterative procedure, where the parameter of interest, say x, is adjusted according to the rule

$$x^{next} = x^{now} - \eta \nabla f(x)\big|_{x = x^{now}} \quad (13)$$

where η is the learning rate, which can be fixed or adjusted during the learning process. The gradient descent equations on the negative log-likelihood can be derived directly by exploiting a useful reparameterization. We reparameterize the HMM using normalized exponentials, in the form

$$a_{ij} = e^{w_{ij}} \Big/ \sum_{l=1}^{N} e^{w_{il}},\ b_j(t) = e^{r_{jt}} \Big/ \sum_{l=1}^{M} e^{r_{jl}},\ \pi_j = e^{u_j} \Big/ \sum_{l=1}^{N} e^{u_l} \quad (14)$$

with w_{ij}, r_{jt}, u_j as the new variables [11].

It is obvious that the new variables w_{ij}, r_{jt}, u_j can also be

arranged in a series of matrices, namely $W = \{w_{ij}\}$, $R = \{r_{jt}\}$ and $U = \{u_j\}$. Correspondingly, a complete specification of a HMM requires specification of the cardinalities of the two state spaces (namely N and M), specification of the observation symbols, and specification of the matrices W, R and U. Henceforth, we may use the compact notation for the model

$$\lambda = (W, R, U). \tag{15}$$

An algorithm for on-line gradient descent on the negative log-likelihood, with w_{ij}, r_{jt}, u_j as parameters to be estimated, can be derived from (13)-(15)

$$w_{ij}^{(next)} - w_{ij}^{(now)} = -\eta\,\partial\left[-\log P(O \mid \lambda)\right] \Big/ \partial w_{ij}\Big|_{w_{ij}=w_{ij}^{(now)}} = \eta\, P(O \mid \lambda)^{-1}\,\partial P(O \mid \lambda) \Big/ \partial w_{ij}\Big|_{w_{ij}=w_{ij}^{(now)}},$$
$$1 \le i \le N,\ 1 \le j \le N. \tag{16}$$

Similarly,

$$r_{jt}^{(next)} - r_{jt}^{(now)} = \eta\, P(O \mid \lambda)^{-1}\,\partial P(O \mid \lambda) \Big/ \partial r_{jt}\Big|_{r_{jt}=r_{jt}^{(now)}},$$
$$1 \le j \le N,\ 1 \le t \le M \tag{17}$$

$$u_j^{(next)} - u_j^{(now)} = \eta\, P(O \mid \lambda)^{-1}\,\partial P(O \mid \lambda) \Big/ \partial u_j\Big|_{u_j=u_j^{(now)}},\ 1 \le j \le N. \tag{18}$$

With the use of lengthy and extensive analytic/algebraic calculations (partially based on the analysis in [8]), the on-line gradient descent equations on the negative log-likelihood are

$$w_{ij}^{(next)} = w_{ij} + \eta\, P(O \mid \lambda)^{-1} a_{ij} \sum_{l=1}^{T-1} \left[\alpha_l(i) b_j(o_{l+1}) \beta_{l+1}(j) - \alpha_l(i)\beta_l(i)\right],$$
$$1 \le i \le N,\ 1 \le j \le N \tag{19}$$

$$r_{jt}^{(next)} = r_{jt} + \eta\, P(O \mid \lambda)^{-1} \sum_{l=1}^{T} \left[I\{o_l = t \mid \lambda\}\alpha_l(j)\beta_l(j) - b_j(t)\alpha_l(j)\beta_l(j)\right],$$
$$1 \le j \le N,\ 1 \le t \le M \tag{20}$$

$$u_j^{(next)} = u_j + \eta\, P(O \mid \lambda)^{-1} \pi_j \left[b_j(o_1)\beta_1(j) - P(O \mid \lambda)\right] = u_j + \eta\pi_j \left[P(O \mid \lambda)^{-1} b_j(o_1)\beta_1(j) - 1\right],\ 1 \le j \le N. \tag{21}$$

The right hand sides of (19)-(21) are evaluated at the current estimates of the HMM's parameters (namely W, R, and U).

III. The Self-Organizing Hidden Markov Model Map

A. General Overview of the SOHMMM

Studies conducted during many years by a great number of researchers have convincingly shown that the best self-organizing results are obtained if the following two partial processes are implemented in their purest forms [6]:

1) decoding of that neuron that has the best match with the input data pattern (the so-called "winner");
2) adaptive improvement of the match in the neighborhood of neurons centered around the "winner."

The SOHMMM may be described formally as a nonlinear, ordered, smooth mapping of observation sequence data onto the elements of a regular, low-dimensional array. The mapping is implemented in the following way, which resembles the two afore mentioned processes. Assume first O is an observation sequence. With each element e in the SOHMMM array we associate a HMM λ_e. Considering the probability of O given λ_e (likelihood), denoted $P(O \mid \lambda_e)$, the image of an input observation sequence O on the SOHMMM array is defined as the array element that matches best with O. This array element has the index

$$c = \arg\max_e \{P(O \mid \lambda_e)\} \tag{22}$$

or, equivalently,

$$c = \arg\min_e \{-\log P(O \mid \lambda_e)\}. \tag{23}$$

Our task is to define the HMM λ_e in such a way that the mapping is ordered, descriptive and representative of the distribution of O. Consider, as in Fig. 1, a two-dimensional ordered array of nodes, where each node has a HMM λ_e associated with it. Moreover, consider the neighborhood set NB_c around the model λ_c, which matches best with O. Here NB_c consists of all neurons up to a certain radius on the grid from neuron c. The next task is to adjust the parameters of all HMMs within NB_c to minimize $-\log P(O \mid \lambda_e)$, that is, to gain some knowledge from the same input O. Actually, we attempt to optimize the parameters of every $\lambda_e \in NB_c$ so as to best describe how a given observation comes about. This is achieved by employing the smooth on-line learning algorithm detailed previously.

B. The SOHMMM Prototype

Let $O = o_1 o_2 .. o_T$ be an observation sequence where each observation o_t assumes a value from the alphabet $F = \{f_1, f_2, \ldots, f_G\}$, and T is the number of observations in the sequence. In addition, let Λ be a class of HMMs such that the corresponding observation symbols ($V = \{v_1, v_2, \ldots, v_M\}$) constitute a superset of the alphabet F ($F \subseteq V$). A further assumption is that two distinct HMMs $\lambda_e, \lambda_e \in \Lambda$, in general, have different cardinalities of the corresponding state spaces

(namely $N^{(c)} \neq N^{(e)}$ and $M^{(c)} \neq M^{(e)}$), different observation symbols ($V^{(c)} \neq V^{(e)}$), and different matrices W, R and U (difference with respect to a matrix refers to nonidentical dimensions and/or nonidentical values of corresponding elements).

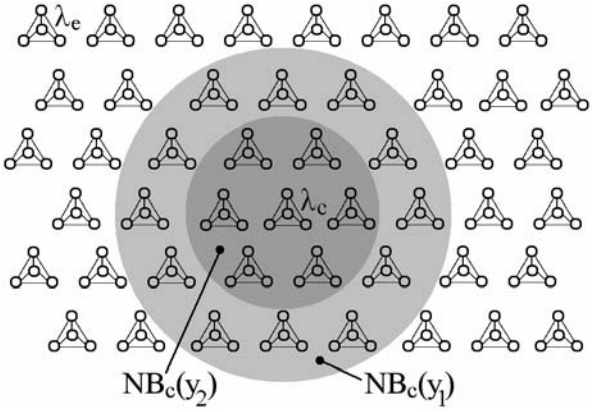

Fig. 1. A paradigm of the SOHMMM, and an example of topological neighborhood ($y_1<y_2$).

The SOHMMM displayed in Fig. 1 defines a mapping from the input observation sequence space onto a two-dimensional array of neurons. With every neuron e, a HMM $\lambda_e \in \Lambda$, also called reference HMM, is associated. The lattice type of the array can be defined to be rectangular, hexagonal or even irregular. In the simplest case, an input observation sequence O is connected to all HMMs in parallel. In an abstract scheme it may be imagined that the input O, by means of some parallel computing mechanisms, is compared with all the λ_e and the location of the best match, with respect to the negative log-likelihood, is defined as the location of the response. Actually, the exact magnitude of the response need not be determined; the input is simply mapped onto this location, like in a set of decoders. We may then claim that the SOHMMM is a nonlinear projection of the input observation sequence onto the two-dimensional display. Thus, the definition of the best matching HMM, indicated by the subscript c, is given by (23).

During learning, or the process in which the nonlinear projection is formed, those HMMs that are topologically close in the array up to a certain geometric distance will activate each other to learn something from the same input O. This will result in a local relaxation or smoothing effect on the parameters of HMMs in this neighborhood, which in continued learning leads to global ordering. In order to adjust the HMMs' parameters to minimize the corresponding negative log-likelihoods ($-\log P(O \mid \lambda_e)$), we follow a hybrid approach by fusing intimately the original incremental SOM training algorithm and the on-line gradient descent algorithm for the negative log-likelihood (19)-(21). The resulting SOHMMM unsupervised learning algorithm is an iterative procedure, where the parameters of interest, namely $W^{(e)}$, $R^{(e)}$ and $U^{(e)}$, are adjusted according to the rules

$$w_{ij}^{(e)}(y+1) = w_{ij}^{(e)}(y) + \eta(y) h_{ce}(y) \cdot \left[P(O \mid \lambda)^{-1} \mathsf{a}_{ij} \sum_{l=1}^{T-1} \left[\alpha_l(i) b_j(o_{l+1}) \beta_{l+1}(j) - \alpha_l(i) \beta_l(i) \right] \Big|_{\lambda_e, y} \right], \quad 1 \le i \le N,\ 1 \le j \le N \tag{24}$$

$$r_{jt}^{(e)}(y+1) = r_{jt}^{(e)}(y) + \eta(y) h_{ce}(y) \cdot \left[P(O \mid \lambda)^{-1} \sum_{l=1}^{T} \left[I\{o_l = t \mid \lambda\} \alpha_l(j) \beta_l(j) - b_j(t) \alpha_l(j) \beta_l(j) \right] \Big|_{\lambda_e, y} \right], \quad 1 \le j \le N,\ 1 \le t \le M \tag{25}$$

$$u_j^{(e)}(y+1) = u_j^{(e)}(y) + \eta(y) h_{ce}(y) \cdot \left[\pi_j \left[P(O \mid \lambda)^{-1} b_j(o_1) \beta_1(j) - 1 \right] \Big|_{\lambda_e, y} \right], \quad 1 \le j \le N \tag{26}$$

where $y = 0, 1, 2, \ldots$ is an integer, the discrete time coordinate. The function $\eta(y)$ plays the role of a scalar learning rate factor ($0<\eta(y)<1$), and, usually, is decreasing monotonically in time (at least during the ordering process). In the relaxation process, the function $h_{ce}(y)$ has a very central role: it acts as the neighborhood function, a smoothing kernel defined over the lattice points. For convenience, it is necessary that $h_{ce}(y) \to 0$ when $y \to \infty$. Usually $h_{ce}(y) = h\left(\left\|\delta_c - \delta_e\right\|, y\right)$, where δ_c, $\delta_e \in \Re^2$ are the location vectors of HMMs λ_c and λ_e on the array. With increasing $\left\|\delta_c - \delta_e\right\|$, $h_{ce} \to 0$. The width and form of h_{ce} define the stiffness of the elastic surface to be fitted to the input data.

In the literature, two simple choices for $h_{ce}(y)$ occur frequently. The simpler of them refers to a neighborhood set of array points around HMM λ_c (Fig. 1). Let their set index be denoted NB_c, whereby $h_{ce}(y) = 1$ if $e \in NB_c$ and $h_{ce}(y) = 0$ if $e \notin NB_c$. It is a common practice that the radius of $NB_c(y)$ is decreasing monotonically in time (at least during the ordering process). Another widely applied, smoother neighborhood kernel can be written in terms of the Gaussian function

$$h_{ce}(y) = \exp\left(-\left\|\delta_c - \delta_e\right\|^2 / 2\sigma^2(y)\right) \tag{27}$$

where the parameter $\sigma(y)$ defines the width of the kernel, and corresponds to the radius of $NB_c(y)$ above. $\sigma(y)$ is a monotonically decreasing function of time, too.

The SOHMMM on-line gradient descent unsupervised learning algorithm, detailed in the present section, is only representative of many possible alternative forms and, certainly, can give rise to a number of variants and different implementations.

IV. Experiments and Applications

The focus of experiments will be on globins. Globins form a well-known family of heme-containing proteins that reversibly bind oxygen, and are involved in its storage and transport. The globin protein family is a large family which is composed of subfamilies. From crystallographic studies, all globins have similar overall three-dimensional structures but widely divergent sequences. The globin sequences used here were extracted from the iProClass protein knowledgebase [12], a database that provides extensive data/information integration of over 90 biological databases. In total, 560 proteins belonging to the three major globin subfamilies were retrieved (namely hemoglobin α-chains, hemoglobin β-chains, and myoglobins). The resulting globin data set's composition is 194 α-globins, 216 β-globins, and 150 myoglobins. Consequently, for all the following experiments the twenty-letter amino acid alphabet of proteins should be considered.

All series of experiments were conducted on training sets consisting of 75 protein sequences picked at random from each one of the three main protein subfamilies. Thus, each training set contained 225 protein sequences, in total. The remaining 335 globins were withheld in order to study and test the SOHMMM on sequence data not used during the training process. The SOHMMM employs a rectangular 9x7 array of 63 HMM neurons, each of which incorporates state space cardinalities of size 11. To test SOHMMM's capabilities and performance, several experiments, with various/diverse learning rate and monotonically decreasing neighborhood functions, were conducted. Also, the maximum duration of both the ordering and tuning phases was set to 20 epochs. In the present study, the mean value of accurately clustered globin sequences is employed as a statistical performance measure (by taking into consideration all 560 protein sequences, both those used for training the SOHMMM and those not used in the learning process). An estimate of this measure, averaged over 10 replications of identical experiments, is 94.22±0.54%. Fig. 2 illustrates the results of a representative scenario after the completion of the SOHMMM on-line unsupervised learning algorithm. The SOHMMM succeeds in capturing the important statistical properties of globins, and manages to divide the training data set into three clusters of similar globin sequences. The examination of the illustrated results confirms that these clusters, which are formed during an automated unsupervised procedure, are distinct and coherent with well-defined boundaries. 181 hemoglobin α-chains, 205 hemoglobin β-chains, and 141 myoglobins are assigned to HMMs that form the clusters of their respective globin subfamilies. Thus, the vast majority of protein sequences are correctly clustered by being associated to HMMs that represent certain domains of the protein subfamilies' spaces. Only few protein sequences (33 strictly speaking) are assigned to HMMs lying at the boundaries of the clusters. These HMMs demonstrate a tendency to describe/represent globins belonging to two different protein subfamilies, thus, obstructing the accurate clustering of specific protein sequences. Nevertheless, such phenomena should be considered justifiable and expected from the moment SOHMMM produces a smooth mapping of the protein subfamilies on a low-dimensional display (in the form of adjacent clusters), and the HMMs under consideration are located at the boundaries of the three clusters.

An attempt to automatically discover subfamilies of globins using a competitive learning approach is described in [4]. The proposed methodology yields a composite HMM consisting of component HMMs. Eventually, each individual cluster is represented by a single component HMM. Seven non-empty clusters representing protein sequences from known globin subfamilies were constructed. Four of the clusters contained varied sequences which belonged to different globin subfamilies or to specific organisms. The entire number of α-globins, β-globins, and myoglobins was distributed to the remaining three (largest) clusters. Cluster 1 contained almost exclusively alpha, α-type, and α-like globins. Nearly all beta, β-type, and β-like globins were included in cluster 2. Finally, the subfamily of myoglobins was assigned to cluster 3. In essence, these results are in agreement with the findings of the series of experiments mentioned before. Such unified unsupervised learning-HMM hybrid approaches are able to discriminate and subsequently cluster the three major globin subfamilies correctly.

The present experimental setup, which is based on the globin protein family, establishes certain properties of the SOHMMM and further confirms some of its advantages. Evidently, the SOHMMM devises a mechanism for handling discrete symbol sequential data (of variable lengths) written in alphabets of arbitrary cardinalities, such as the twenty-letter amino acid alphabet of proteins. Also, the SOHMMM is able to access and exploit the latent information hidden in the spatial dependencies/correlations of protein chain molecules. The exact values of the training parameters (albeit inaccurate and sketchy) do not seem to affect the SOHMMM's efficiency and robustness. As has been shown, rough estimates of these parameters (i.e. the learning rate factor and neighborhood function) prove more than adequate. Moreover, even a small number of training cycles (20 epochs at maximum) is usually sufficient to ensure stability and convergence.

By implementing the self-organization and competitive strategies in their purest versions, the SOHMMM has succeeded in contriving higher abstractions (symbolisms) that build upon the probabilistic attributes of topologically neighboring HMMs. The SOHMMM approach was able to capture the important statistical properties of the globin protein family by covering a larger set of distributions, and consequently, by expressing relations inaccessible to single uncorrelated HMMs. Thus, it accomplished the tasks of identifying the three major globin subfamilies, and of

partitioning all protein sequences into the three clusters that represented these subfamilies. A matter of significant value is that both tasks have been carried out along a straightforward process of learning from protein chain molecules, without requiring any kind of prior or posterior knowledge. The SOHMMM reached a generalization level that eliminated overtraining phenomena, and, at the same time, clustered accurately the landslide of globins that had been used for training and testing. In this case, the

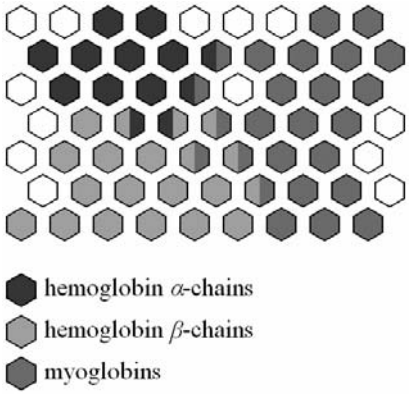

Fig. 2. The SOHMMM plane after the completion of the training process. Each hexagon represents a HMM neuron. HMMs assigned to protein sequences from each one of the three globin subfamilies are distinguished by distinct grayscale shades.

SOHMMM appears to be rather independent of the number of input protein sequences, something which became evident from the fact that less than half of the available globins sufficed to form the final model.

One of SOHMMM's main objectives is to produce simplified descriptions and summaries of sequence data sets. As has been shown, it projects globin subfamilies (which are high-dimensional/complex sequence data) as points on a two-dimensional display. These projections represent the input protein sequences in a lower-dimensional space in such a way that clusters and relations between globins are preserved as faithfully as possible. In addition, each cluster's constituent HMMs develop into decoders of their respective protein sequence domains. The globins which are assigned to a SOHMMM neuron are actually described/represented by a probabilistic model in a process resembling dimensionality reduction. Finally, an issue of significant importance is that once a SOHMMM has been successfully derived from a family of protein sequences, all HMM nodes can be labeled according to the assigned proteins' categories or annotate information, as in Fig. 2. Since the best matching neuron of any given unknown/unlabeled protein chain molecule can be computed, this protein can be classified as belonging to the cluster represented by the labeled HMM node. Processes based on this strategy can be used in discrimination tests, database searches and classification problems. If well-established class-specific training sets for protein families were available SOHMMMs could be derived (based on them), and, subsequently, be employed for similarity searches across protein databases. Inter alia, this series of experiments demonstrates an exemplar classification task according to which each one of the 335 globins, not used for training, was classified as belonging to the cluster/subfamily represented by the labeled HMM neuron that returned the highest probability.

V. Conclusion

In this paper, the theory of the SOHMMM framework has been briefly presented and the behavior of the model has been studied through experiments. The experimental approach that has been followed constitutes a first step in studying and analyzing important aspects of the SOHMMM. Also, attempts have been made to illustrate certain applications of the SOHMMM to characteristic problems, in an effort to point out how these techniques could be applied to more advanced/complex problems.

In real-case applications we need to consider more complex SOHMMM architectures, employing HMMs with many more states and typically sparser connectivity. The design or selection of such architectures is highly problem dependent. In biological sequences the linear aspects of the sequences are captured by the so-called left-right architectures, the most basic and widely used of which is the standard linear architecture. Hence, a potential expansion could involve the integration of left-to-right HMMs into the SOHMMM.

Finally, a subject of substantial significance is putting into practice the ideas of the SOHMMM in realistic applications. The efficiency of the SOHMMM framework, in a practical sense, will be proved if it succeeds in coping efficiently with a wide gamut of diverse real-case problems.

References

[1] D. W. Mount, *Bioinformatics: Sequence and Genome Analysis*, 2nd ed. New York: Cold Spring Harbor Laboratory Press, 2004.

[2] P. Baldi and S. Brunak, *Bioinformatics: The Machine Learning Approach*, 2nd ed. Cambridge, Massachusetts: The MIT Press, 2001.

[3] P. Baldi and Y. Chauvin, "Hybrid modeling, HMM/NN architectures, and protein applications," *Neural Computation*, vol. 8, pp. 1541-1565, Oct. 1996.

[4] A.Krogh, M. Brown, I. S. Mian, K. Sjolander and D. Haussler, "Hidden Markov models in computational biology: applications to protein modeling," *J. Molecular Biology*, vol. 235, pp. 1501-1531, Feb. 1994.

[5] L. E. Baum, "An inequality and associated maximization technique in statistical estimation for probabilistic functions of Markov processes," *Inequalities*, vol. 3, pp. 1-8, 1972.

[6] T. Kohonen, *Self-Organizing Maps*, 3rd ed. Berlin: Springer-Verlag, 2001.

[7] U. Seiffert and L. C. Jain, *Self-Organizing Neural Networks: Recent Advances and Applications*. Heidelberg, New York: Physica-Verlag 2002.

[8] T. Koski, *Hidden Markov Models for Bioinformatics*. Dordrecht, The Netherlands: Kluwer Academics Publishers, 2001.

[9] J.-S. R. Sang, C.-T. Sun and E. Mizutami, *Neuro-Fuzzy and Soft Computing: A Computational Approach to Learning and Machine Intelligence*. Upper Saddle River, New Jersey: Prentice-Hall, 1997.

[10] P. Baldi, "Gradient descent learning algorithms overview: a general dynamical systems perspective," *IEEE Trans. Neural Netw.*, vol.6, pp. 182-195, Jan. 1995.

[11] P. Baldi and Y. Chauvin, “Smooth on-line learning algorithms for hidden Markov models,” *Neural Computation*, vol. 6, pp. 305-316, March 1994.
[12] The iProClass Protein Knowledgebase (release 3.48). [Online]. Available: http://pir.georgetown.edu/