Bioinformatics, 35(13), 2019, 2208–2215
doi: 10.1093/bioinformatics/bty910
Advance Access Publication Date: 16 November 2018
Original Paper

Sequence analysis

Semi-supervised learning of Hidden Markov Models for biological sequence analysis

Ioannis A. Tamposis[1], Konstantinos D. Tsirigos[2], Margarita C. Theodoropoulou[1], Panagiota I. Kontou[1] and Pantelis G. Bagos[1,*]

[1]Department of Computer Science and Biomedical Informatics, University of Thessaly, 35100 Lamia, Greece and [2]Department of Bio and Health Informatics, Technical University of Denmark, Kgs Lyngby, Denmark

*To whom correspondence should be addressed.

Associate Editor: John Hancock

Received on June 4, 2018; revised on October 29, 2018; editorial decision on October 30, 2018; accepted on November 9, 2018

Abstract

Motivation: Hidden Markov Models (HMMs) are probabilistic models widely used in applications in computational sequence analysis. HMMs are basically unsupervised models. However, in the most important applications, they are trained in a supervised manner. Training examples accompanied by labels corresponding to different classes are given as input and the set of parameters that maximize the joint probability of sequences and labels is estimated. A main problem with this approach is that, in the majority of the cases, labels are hard to find and thus the amount of training data is limited. On the other hand, there are plenty of unclassified (unlabeled) sequences deposited in the public databases that could potentially contribute to the training procedure. This approach is called semi-supervised learning and could be very helpful in many applications.

Results: We propose here, a method for semi-supervised learning of HMMs that can incorporate labeled, unlabeled and partially labeled data in a straightforward manner. The algorithm is based on a variant of the Expectation-Maximization (EM) algorithm, where the missing labels of the unlabeled or partially labeled data are considered as the missing data. We apply the algorithm to several biological problems, namely, for the prediction of transmembrane protein topology for alpha-helical and beta-barrel membrane proteins and for the prediction of archaeal signal peptides. The results are very promising, since the algorithms presented here can significantly improve the prediction performance of even the top-scoring classifiers.

Contact: pbagos@compgen.org

Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 Introduction

Hidden Markov Models (HMMs) are probabilistic models initially developed for applications in speech recognition (Rabiner, 1989). During the last two decades they were successfully applied for various tasks in computational molecular biology (Durbin *et al.*, 1998). These include gene finding (Krogh *et al.*, 1994), multiple sequence alignment (Eddy, 1995), prediction of signal peptides (Bagos *et al.*, 2009a,b; Nielsen and Krogh, 1998), prediction of bacterial lipoproteins (Bagos *et al.*, 2008; Juncker *et al.*, 2003), prediction of cell-wall sorting signals (Litou *et al.*, 2008), prediction of protein secondary structure (Asai *et al.*, 1993) and prediction of transmembrane protein topology (Bagos *et al.*, 2004; Krogh *et al.*, 2001). In several of these applications, such as topology prediction of transmembrane proteins, HMMs have been found to perform significantly better compared to other sophisticated Machine-Learning techniques such as Neural Networks or Support Vector Machines, as shown by evaluation studies (Bagos *et al.*, 2005; Moller *et al.*, 2001; Viklund and Elofsson, 2004). In the recent

years, consensus topology prediction methods have also been introduced, including TOPCONS (Tsirigos *et al.*, 2015) and CCTOP (Dobson *et al.*, 2015). These methods combine the outputs of several other prediction methods into a consensus prediction, using dynamic programming.

HMMs are generative models and in their basic formulation operate in an *unsupervised* manner, since they simply describe a finite mixture of multinomial distributions where the mixture probabilities form a 1st order Markov chain. In this setting, during the *training* phase we maximize $P(\mathbf{x}|\theta)$, which is the probability of the data given the model, whereas in the *decoding* phase we recover the hidden sequence of states that are most likely to have generated the data. In this manner, the only 'supervision' needed is that of providing a reliable set of homologous sequences. When there is a need to compare different competing models, supervision is used indirectly, i.e. we train the different models separately and in the testing phase we simply choose the one with the highest probability (i.e. in a database search). In other applications, such as structure prediction, a sequence of labels ($\mathbf{y}$) is tied to each observation sequence ($\mathbf{x}$), corresponding to the different attributes that we wish to predict. There we usually maximize $P(\mathbf{x}, \mathbf{y}|\theta)$, which is the joint probability of the sequences and the labels given the model, or $P(\mathbf{y}|\mathbf{x},\theta)$, which is the probability of labels given the sequences and the model. These approaches typically correspond to a *supervised* learning procedure where each sequence $\mathbf{x}$ is accompanied by a complete sequence of well-defined labels $\mathbf{y}$.

Semi-supervised learning is placed somewhere in between supervised learning (i.e. complete labels for all the training sequences) and unsupervised learning (i.e. completely unlabeled training sequences). Various approaches have been proposed, for instance using naive Bayesian classifiers (Nigam *et al.*, 2000; Yarowsky, 1995) and HMMs (Inoue and Ueda, 2003; Ji *et al.*, 2008) usually in the context of text categorization and information extraction. The motivation of these methods lies in the fact that, in many areas of research, labeled examples are relatively hard to find, whereas there are plenty of unclassified (unlabeled) data that can potentially be used to improve the performance of the classifier. This situation is also common in molecular biology where the huge amount of protein sequences deposited in public databases (usually deciphered by conceptual translation) contradicts the relatively few examples of proteins with experimentally verified function. Semi-supervised methods have also been proposed for other computational biology problems: remote homology detection (Shah *et al.*, 2008), gene expression prediction (Hafez *et al.*, 2017), protein prediction (El-Manzalawy *et al.*, 2016) and proteomics applications (Fischer *et al.*, 2006; Kall *et al.*, 2007). Here, we present a novel approach for semi-supervised learning of HMMs for biological sequence analysis along with its explanation in terms of the Expectation-Maximization (EM) algorithm. A major advantage of the method is the ease of implementation and we expect that it will be widely used.

2 Materials and methods

2.1 Hidden Markov Models

A Hidden Markov Model (HMM) is a probabilistic model consisting of states forming a 1st order Markov chain (Durbin *et al.*, 1998; Rabiner, 1989). Two states k, l of a Hidden Markov model are connected by means of the transition probabilities α_{kl}. Assuming a protein sequence $\mathbf{x}$ of length L denoted by $\mathbf{x} = x_1, x_2, \ldots, x_L$ where the x_is are the amino acids, we usually denote the 'path' (i.e. the sequence of states) ending up to a particular position of the amino acid sequence (the sequence of symbols), by π. Each state k is associated with an emission probability $e_k(x_i)$, which is the probability of a particular symbol x_i to be emitted by that state. The total probability of a sequence given the model is calculated by summing over all possible paths using the forward or the backward algorithm as follows:

$$P(\mathbf{x}|\theta) = \sum_{\pi} P(\mathbf{x}, \pi|\theta) = \sum_{\pi} a_{B\pi_1} \prod_{i=1}^{L} e_{\pi i}(x_i) a_{\pi_i \pi_{i+1}} \quad (1)$$

The generalization from one to many sequences is trivial and we will consider only one training sequence $\mathbf{x}$ from now on. Traditionally, HMM training is performed using the Baum-Welch algorithm (Baum, 1972; Durbin *et al.*, 1998; Rabiner, 1989) which is a special case of the Expectation-Maximization (EM) algorithm for missing data (Dempster *et al.*, 1977). Missing data in this case is the path π (i.e. the sequence of states), since, if we knew the exact path, the Maximum Likelihood (ML) estimates could be easily derived by counting the observed transitions and emissions. An alternative to the Baum-Welch algorithm, even though not widely used, is the gradient-descent algorithm proposed by Baldi and Chauvin (1994). In any case, in these models, maximization of the likelihood of Eq. (1) corresponds to an *unsupervised learning* problem. Usually, in speech recognition, as well as in some biological problems, observations classified in different classes are collected, the models are trained separately and, in the testing phase, the highest scoring among the competing models is used for classification. In other applications, such as multiple sequence alignment, the learning is performed in a completely unsupervised manner (with the exception of having to collect the members of the protein family).

In other biological sequence analysis problems, where we want to classify various segments along the sequence, we often use labeled sequences for training. In such cases, each amino acid sequence $\mathbf{x}$ is accompanied by a sequence of labels $\mathbf{y}$ for each position i in the sequence ($\mathbf{y} = y_1, y_2, \ldots, y_L$). Consequently, we declare a new probability distribution, the probability $\delta_k(y_i = c)$ of a state k having a label c. In most applications, this probability is just a delta-function, since a particular state is not allowed to match more than one label. Krogh proposed a simple modification of forward and backward algorithms in order to incorporate information from labeled data (Krogh, 1994). The likelihood to be maximized in such situations is the joint probability of the sequences ($\mathbf{x}$) and the labels ($\mathbf{y}$) given the model, in which the summation has to be done only over those paths Π_y that are in agreement with the labels $\mathbf{y}$:

$$P(\mathbf{x}, \mathbf{y}|\theta) = \sum_{\pi} P(\mathbf{x}, \mathbf{y}, \pi|\theta) = \sum_{\pi \in \Pi_y} P(\mathbf{x}, \pi|\theta) \quad (2)$$

This typically corresponds to a *supervised learning* procedure. With the use of labeled sequences, we can also perform a kind of discriminative training, with a criterion known as Conditional Maximum Likelihood (CML). In this approach, we wish to maximize the probability of the labels given the sequences: $P(\mathbf{y}|\mathbf{x},\theta) = P(\mathbf{x}, \mathbf{y}|\theta)/P(\mathbf{x}|\theta)$ (Krogh, 1997). The CML criterion is computationally expensive since it requires calculating the likelihood twice. Furthermore, there is no EM algorithm for training and one has to use general gradient-based methods (Bagos *et al.*, 2004). On the other hand, it is known that it can yield better results if sufficient data with labels of high-quality are available.

In most of the cases, training is performed using labeled data, whereas testing using unlabeled. In the past, we presented algorithms for constrained prediction, that is, in the context discussed

here, we used partially labeled sequences in the decoding phase (Bagos *et al.*, 2006). We presented some trivial modifications to the forward (and backward) algorithm that consisted simply of setting the intermediate variables equal to zero for each position i and state k that is not in agreement with the prior experimental information. This is conceptually similar to training using labeled sequences, where one allows only those paths Π_y that are in agreement with the labeling $\mathbf{y}$ to contribute to the total likelihood. Similar modifications were made in all the decoding algorithms used for HMMs. These modified algorithms practically allow the fixation of a given segment of the sequence to a labeling that is known *a priori* (for instance, by using experimental techniques). Here, we simply extend these ideas further, by allowing training to also be performed including unlabeled or partially labeled sequences in the training set.

2.2 Semi-supervised of Hidden Markov Models

As we already mentioned, semi-supervised learning is placed somewhere in-between supervised and unsupervised learning (Chapelle *et al.*, 2006). Different approaches have been proposed using naive Bayesian classifiers (Nigam *et al.*, 2000) and HMMs (Inoue and Ueda, 2003; Ji *et al.*, 2008) usually in the context of text categorization and information extraction. These semi-supervised HMM methods are presented in an entirely different framework from the one used in biological sequence analysis. The main difference is that, in most sequence analysis problems, the 'unit' of observation is a residue i along the sequence $\mathbf{x}$, and thus, we may encounter examples with complete labels (labeled data), examples with no labels at all (unlabeled data) and examples with incomplete labels along the sequence (partially labeled data). The biological sequence analysis problems differ also from other similar problems; for instance, in speech recognition, the (incomplete) sequence of labels is shorter and needs to be aligned during the training against the sequence of observations (Krogh and Riis, 1999). Furthermore, in some other scientific areas, the term label refers to the knowledge of the path (π) through the model, and thus a semi-supervised approach (training from partially labeled data) is simply the case in which part of the data has the precise state path determined (Scheffer *et al.*, 2001).

A first example of semi-supervised learning, even though not mentioned as such, was used in the case of topology prediction of membrane proteins (Krogh *et al.*, 2001). The labels in such a case correspond to the transmembrane regions and to the intracellular and extracellular loops, respectively. However, even if the observed labels arise from crystallographically determined structures we cannot identify the boundaries of the lipid bilayer precisely. Thus, these inherently misplaced labels may bias the training, resulting into poor discriminative capability. In some of the most successful models for predicting the membrane-spanning alpha helices (Kall *et al.*, 2004; Krogh *et al.*, 2001; Viklund and Elofsson, 2004), an optimization procedure was used, according to which a model was initially trained, the labels were partially disregarded around the ends of the transmembrane helices and predictions conditional on the remaining labels were performed in order to re-label the data until the final training procedure.

2.2.1 Self-training

The algorithm presented is an instance of the so-called self-training approach, which is a commonly used technique for semi-supervised learning (Yarowsky, 1995). In the usual formulation, a classifier is first trained with a small amount of labeled data and is then used to classify the unlabeled data (Fig. 1A). Typically, the most confident unlabeled sequences, together with their predicted labels, are added

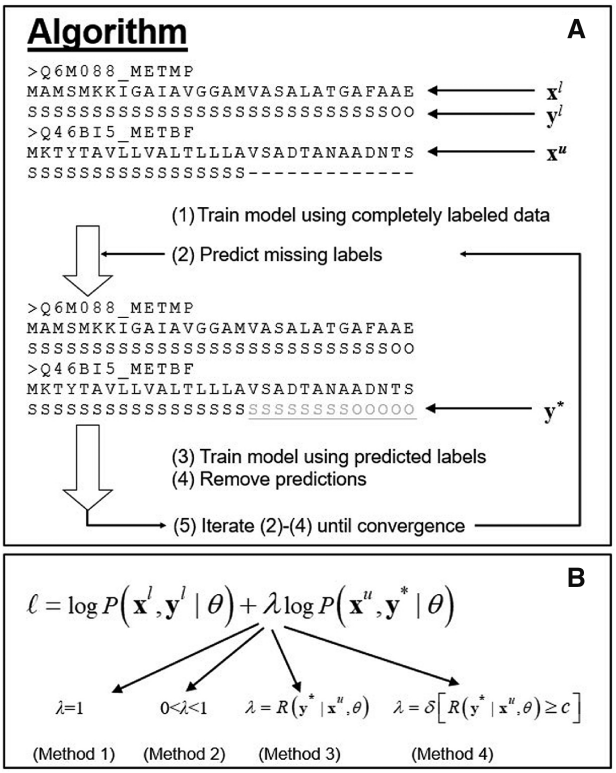

Fig. 1. (**A**) A schematic illustration of the algorithm. (**B**) The likelihood that is being maximized in the four variants of the algorithm described in the text

to the training set, the classifier is re-trained and the procedure is repeated. From now on, we denote the complete-label data by ($\mathbf{x}^l$, $\mathbf{y}^l$) and the unlabeled or incompletely labeled ones by ($\mathbf{x}^u$). Thus, the algorithm is formulated as:

1. Use the completely labeled data ($\mathbf{x}^l$, $\mathbf{y}^l$) to train an initial model (θ).
2. Use θ to predict the labels ($\mathbf{y}^*$) of the unlabeled or partially labeled data ($\mathbf{x}^u$).
3. Use the newly labeled data ($\mathbf{x}^u$, $\mathbf{y}^*$) along with the completely labeled ones ($\mathbf{x}^l$, $\mathbf{y}^l$) in order to train a new model (θ^*).
4. Remove the predicted labels ($\mathbf{y}^*$) in order to obtain the initial dataset. Use the new model θ^* to replace θ.
5. Iterate steps (2)–(4) until convergence.

Self-training is a wrapper method, meaning that, in general, any classifier can be used without modifications. Interestingly, the same procedure was used with Bayesian classifiers (Nigam *et al.*, 2000) and HMMs (Inoue and Ueda, 2003). Some general comments have to be made concerning self-training. First of all, different combinations of algorithms can be used for training (i.e. Baum-Welch, gradient-descent, etc.) and for decoding (i.e. Viterbi, Posterior-Viterbi, etc.). However, discriminative methods based on the CML approach are not expected to perform well, since their performance is related to the quality of the available labels. Nevertheless, the Baum-Welch algorithm is preferred since it is easily implemented and globally convergent. Secondly, if the incomplete data consist solely of completely unlabeled sequences, we can use the standard decoding algorithms. If, on the other hand, we have sequences with partial labels (incompletely labeled data), then the modified algorithms for constrained predictions described previously (Bagos *et al.*, 2006) have to be used in step (2). Lastly, since the classifier in the simplified

version presented above (which we call Method 1), uses its own predictions to teach itself, we can imagine that a classification mistake can reinforce itself. A possible solution to this is by 'unlearning' unlabeled sequences if the prediction confidence drops below a threshold. Another commonly used heuristic is to down-weigh all unlabeled data, multiplying their contribution to the total log-likelihood by a (constant) factor λ, where $0<\lambda<1$. Alternatively, we can use some metrics of prediction reliability such as the ones proposed by Melen *et al.* (2003) (see next section).

2.2.2 Convergence properties and extensions of the algorithm

The general framework of self-training is considered a reminiscent of the EM algorithm. The Expectation (E-step) is performed in step (2), whereas the Maximization (M-step) is performed in step (3). The performance of the EM algorithm is known to be dependent on the initial values, and this is highlighted in step (1) where the completely labeled data are used to build the initial model. It is easily understood that, if we have very few or low-quality labeled data, the initialization will provide an inadequate classifier and the whole procedure will lead to poor performance. Since self-training is a wrapper algorithm, a major disadvantage is that its convergence properties cannot be easily studied, except for special cases (Abney, 2004). In the case of HMMs, a brief explanation of the EM behavior of the algorithm is outlined below. We first consider the complete log-likelihood and decompose it in two independent terms corresponding to that of the labeled and unlabeled data, respectively:

$$l = \log P(\mathbf{x}^l, \mathbf{y}^l, \mathbf{x}^u|\theta) = \log P(\mathbf{x}^l, \mathbf{y}^l|\theta) + \log \sum_{\mathbf{y}^u} P(\mathbf{x}^u, \mathbf{y}^u|\theta) \quad (3)$$

The first term corresponds to the log-likelihood of the labeled data that can be calculated using the existing algorithms for labeled sequences, whereas the second term is the log-likelihood of the unlabeled data. Concerning the second term, the general EM-framework which treats the unknown labels ($\mathbf{y}^u$) as missing data states that, instead of maximizing the joint log-likelihood directly, we can maximize an auxiliary function Q, defined as the expectation of the joint log-likelihood given the observed data. Thus, the auxiliary function will be:

$$Q = E_{\mathbf{y}^u}[\log P(\mathbf{x}^u, \mathbf{y}^u|\theta)] = \sum_{\mathbf{y}^u} P(\mathbf{y}^u|\mathbf{x}^u, \theta) \log P(\mathbf{x}^u, \mathbf{y}^u|\theta) \quad (4)$$

If we apply Eq. (4) to the labeled data ($\mathbf{x}^l$, $\mathbf{y}^l$) and treat their labels as observed quantities (i.e. use the fact that $P(\mathbf{y}^l|\mathbf{x}^l,\theta)=1$), the expression for the complete dataset will be:

$$Q = \log P(\mathbf{x}^l, \mathbf{y}^l|\theta) + \sum_{\mathbf{y}^u} P(\mathbf{y}^u|\mathbf{x}^u, \theta) \log P(\mathbf{x}^u, \mathbf{y}^u|\theta) \quad (5)$$

From this perspective, the self-training algorithm outlined above can be considered as a discrete approximation to EM; for completely labeled data we use the observed class memberships directly (the true labels $\mathbf{y}^l$), whereas, for unlabeled data, we use the predicted labels ($\mathbf{y}^*$) which are those that maximize the posterior probability $P(\mathbf{y}^*|\mathbf{x}^u,\theta)$. In other words, we avoid the summation over all possible labels by thresholding the probabilities to 0 or 1. The modified likelihood that is being maximized in this case could be termed 'label-optimized log-likelihood' following the work of Juang and Rabiner (1990) and has the form:

$$Q^* = \log P(\mathbf{x}^l, \mathbf{y}^l|\theta) + \log P(\mathbf{x}^u, \mathbf{y}^*|\theta) \quad (6)$$

From the general EM framework and the Jensen's inequality it is obvious that Q^* and Q are always bounded below ℓ and, thus, maximizing Eq. (5) or Eq. (6) also maximizes Eq. (4). As already mentioned, a common heuristic that was shown to perform well is to down-weight unlabeled data, multiplying their contribution to the total log-likelihood by a factor λ, where $0<\lambda<1$ (Nigam *et al.*, 2000). In other words, the $\log P(\mathbf{x}^u, \mathbf{y}^*|\theta)$ term in Eq. (6) is replaced by $\lambda \log P(\mathbf{x}^u, \mathbf{y}^*|\theta)$ with $0<\lambda<1$. We call this method Method 2. However, the optimal (constant) value of λ can only be found with cross-validation. In the case of HMMs, an alternative would be to weight each prediction, not by a constant factor, but by its confidence or, in other words, its posterior probability. A useful approach in this regard could be to use some metrics for prediction reliability proposed by Melen *et al.* (2003). We recall from previous works (Bagos *et al.*, 2006; Kall *et al.*, 2005) that the sum of the posterior states probabilities over the states that share the same label c is called the Posterior Label Probability (PLP):

$$P(y_i = c|\mathbf{x}, \theta) = \sum_k P(\pi_i = k|\mathbf{x}, \theta)\delta_k(y_i = c) \quad (7)$$

The posterior state probabilities can be easily calculated from the standard forward and backward algorithms (Durbin *et al.*, 1998; Rabiner, 1989). Averaging the PLPs of the predicted labels for each sequence gives us an estimate for the reliability of the prediction:

$$R(\mathbf{y}^*|\mathbf{x}^u, \theta) = \frac{\sum_i P(y_i^*|\mathbf{x}^u, \theta)}{L} \quad (8)$$

In general, any decoding algorithm can be used, but the Optimal Accuracy Posterior Decoder (Kall *et al.*, 2005), which maximizes a measure related to Eq. (8), seems to be the obvious choice. Nevertheless, this quantity can be used to down-weight the contribution of the predicted labels to the label-optimized log-likelihood of Eq. (6). Thus, we have the objective function:

$$Q^W = \log P(\mathbf{x}^l, \mathbf{y}^l|\theta) + R(\mathbf{y}^*|\mathbf{x}^u, \theta) \log P(\mathbf{x}^u, \mathbf{y}^*|\theta) \quad (9)$$

It is clear that the Reliability of the prediction can be considered as an individual (i.e. for each sequence s) weighting factor λ that additionally is being adapted during training; thus, it is expected to perform well without the need for fine-tuning. Obviously, $Q^W \leq Q^* \leq \ell$ and, thus, maximizing Eq. (9) maximizes Eq. (4) as well. This option is called Method 3. In order for Methods 2 and 3 to work, we have to properly weigh the expected counts A_{kl} and $E_k(b)$ computed by the forward-backward algorithm. Thus, all counts derived from an unlabeled sequence $\mathbf{x}^u$ have to be multiplied by $R(\mathbf{y}^*|\mathbf{x}^u,\theta)$ (or with λ). Consequently, these two variants require some modifications to the standard Forward and Backward algorithms. All three methods presented so far include all the unlabeled data in the training, using, however, different weighting schemes (Fig. 1B). Another easily implemented variant would be to include only the unlabeled data with reliabilities over a certain cutoff in the training. This method (Method 4) is more restrictive since the optimal threshold needs to be identified, but has the additional advantage of fewer computations since only a fraction of the unlabeled data is included in the training phase. On the other hand, Method 1 is more easily implemented but more crude. Nevertheless, the comparative performance of each method needs to be identified in a case-by-case basis, since each particular problem with different labeled and unlabeled sequences may exhibit different properties.

2.2.3 General comments

As we noted earlier, in typical HMM applications, as missing data we consider the path of states (π) and the Baum-Welch algorithm

uses the notation of Eq. (4) with π instead of $\mathbf{y}$. Thus, in the semi-supervised case, we have two levels of 'missing' data that are being treated in different steps. The lower-level 'missing' data ($\boldsymbol{\pi}$) are being dealt with only in step (3) where the higher-level 'missing' data ($\mathbf{y}$) are kept fixed, whereas $\mathbf{y}$ is optimized externally by successively iterating steps (2)–(4). A full EM approach could, in principle, be obtained from Eqs. (3, 4) if we also consider the path (π); then Eq. (3) becomes:

$$\begin{aligned} l &= \log \sum_{\pi} P(\mathbf{x}^l, \mathbf{y}^l, \mathbf{x}^u, \pi|\theta) \\ &= \log \sum_{\pi} P(\mathbf{x}^l, \mathbf{y}^l, \pi|\theta) + \log \sum_{\mathbf{y}^u} \sum_{\pi} P(\mathbf{x}^u, \mathbf{y}^u, \pi|\theta) \end{aligned} \tag{10}$$

Whereas Eq. (4) becomes:

$$Q = \sum_{\mathbf{y}^u} \sum_{\pi} P(\mathbf{y}^u, \pi|\mathbf{x}^u, \theta) \log P(\mathbf{x}^u, \mathbf{y}^u, \pi|\theta) \tag{11}$$

It is clear that, in such a case, the M-step is difficult since we have to deal with π and $\mathbf{y}^u$ simultaneously. The algorithm proposed here resembles closely the approach known as the Expectation Conditional Maximization (ECM) algorithm, which replaces the M-step with a sequence of conditional maximization (CM) steps in which each parameter is maximized individually, conditionally on the other parameters remaining fixed (Meng and Rubin, 1993). ECM is more broadly applicable than EM, shares its desirable convergence properties, but avoids a computationally difficult or even intractable M-step. Similar approximations to EM are known for years. For instance, in standard HMM parameter estimation (where $\boldsymbol{\pi}$ is the missing data), an alternative to the Baum-Welch algorithm is the so-called Viterbi learning or segmental k-means algorithm (Juang and Rabiner, 1990). In this variant, the emission and transition counts are computed only by the most probable path produced by the Viterbi algorithm, instead of summing over all possible paths. Based on Juang and Rabiner (1990), it is obvious that the self-training algorithm described above will always converge to a local maximum. A situation that remotely resembles semi-supervised learning has been used in the case of transmembrane protein topology prediction. It has been found that an initially trained model could be further refined in order to be applied on test sequences using the Baum-Welch algorithm in an unsupervised manner. This approach improved the prediction accuracy in several applications (Tusnady and Simon, 2001; Viklund and Elofsson, 2004). The authors of these works were driven to this approach following a different rationale, namely the concept of improving the prediction accuracy by incorporation of information from homologues. The key difference of such an approach lies in the fact that, in the second step, the algorithm maximizes the marginal distribution of the unlabeled data $P(\mathbf{x}^u|\theta)$, that is, it completely disregards the distribution of the labels.

2.3 Datasets

We evaluated the newly developed methodology on three important problems in computational sequence analysis. In order to accurately measure the contribution of the new method, each predictor was trained using the standard supervised approach and the semi-supervised approaches previously described. To prepare our dataset, we: (i) extracted and selected proteins, (ii) merged labeled protein subsets with the unlabeled subsets and (iii) reduced the redundancy of the sequences with a rigorous threshold (30% similarity). For topology prediction of alpha-helical membrane proteins we used the HMM-TM method (Bagos *et al.*, 2006) trained on a dataset of 308 membrane proteins with known three-dimensional structures (Tsirigos *et al.*, 2015) and for the semi-supervised learning we used the dataset of 2126 proteins from ExTopoDB (Tsaousis *et al.*, 2010) whose structures are not known but limited experimental information about the topology is available. We treated these sequences as partially labeled ones, since we used only the experimental information regarding the localization of various parts of the sequence. The redundancy-reduced set contains 286 labeled and 1466 unlabeled sequences. In the case of topology prediction of beta-barrel outer membrane proteins, we used the PRED-TMBB2 method (Tsirigos *et al.*, 2016) trained with a non-redundant dataset of 49 outer membrane proteins with known three-dimensional structures. In this case, the semi-supervised learning procedure consisted of including a dataset of 1005 experimentally verified outer membrane proteins from OMPdb (Tsirigos *et al.*, 2010) whose structures are not available (unlabeled sequences) in the training phase. The redundancy-reduced set contains 49 labeled and 980 unlabeled sequences. In the third case, we used a previously published dataset of 70 signal peptides from archaeal proteins (Bagos *et al.*, 2009a,b). The dataset consisted of 25 proteins with experimentally verified cleavage site (labeled sequences) and 45 sequences of signal peptides for which we did not know the location of the cleavage site precisely (partially labeled sequences). In all cases, the HMM architectures were exactly the same as in the original publications and a 10-fold cross-validation procedure was used to assess the prediction accuracy. Concerning transmembrane proteins, the number of correctly predicted residues (Q), the segment overlap (SOV) measure and the number of correctly predicted topologies were evaluated. In the case of signal peptides, we evaluated the accuracy in predicting the precise location of the cleavage site as well as the accuracy in discriminating signal peptides from non-signal peptide sequences.

3 Results

The results obtained from the 10-fold cross-validation procedure regarding transmembrane protein topology prediction are presented in Figure 2. We first consider the use of self-training in order to incorporate information from unlabeled or partially labeled sequences. Since for both alpha-helical and beta-barrel outer membrane proteins we had rather large datasets with labeled data, we decided to vary the amount of labeled data and compare the classification accuracy of standard supervised learning of HMMs (no unlabeled sequences) against a self-training learner that uses unlabeled sequences. This was done in order to identify the potential effect of unlabeled data in relation to the labeled data, since previous studies have shown that the size of the (labeled) dataset is crucial up to a certain degree (Bagos *et al.*, 2009a,b; Tamposis *et al.*, 2018).

Figure 2 shows that the incorporation of both labeled and unlabeled data can increase the classification accuracy remarkably. More specifically, in the case of beta-barrels, the increase in SOV ranges from 0.1 to 8.5%, the increase in the fraction of correctly predicted topologies reaches 35% and the increase in the fraction of correctly predicted residues reaches 6.9%. Regarding alpha-helical TM proteins (with the largest labeled set), when the number of labeled sequences exceeds a certain amount, semi-supervised methods perform approximately the same as supervised learning. This is because the amount of labeled data is large enough to train accurate HMMs and additional unlabeled sequences provide little extra help. When the classifiers are trained from small amounts of labeled data, the four variants of the semi-supervised learning yield better results compared to standard supervised methods. More specifically, when less than 90 sequences were included in the training set, the increase

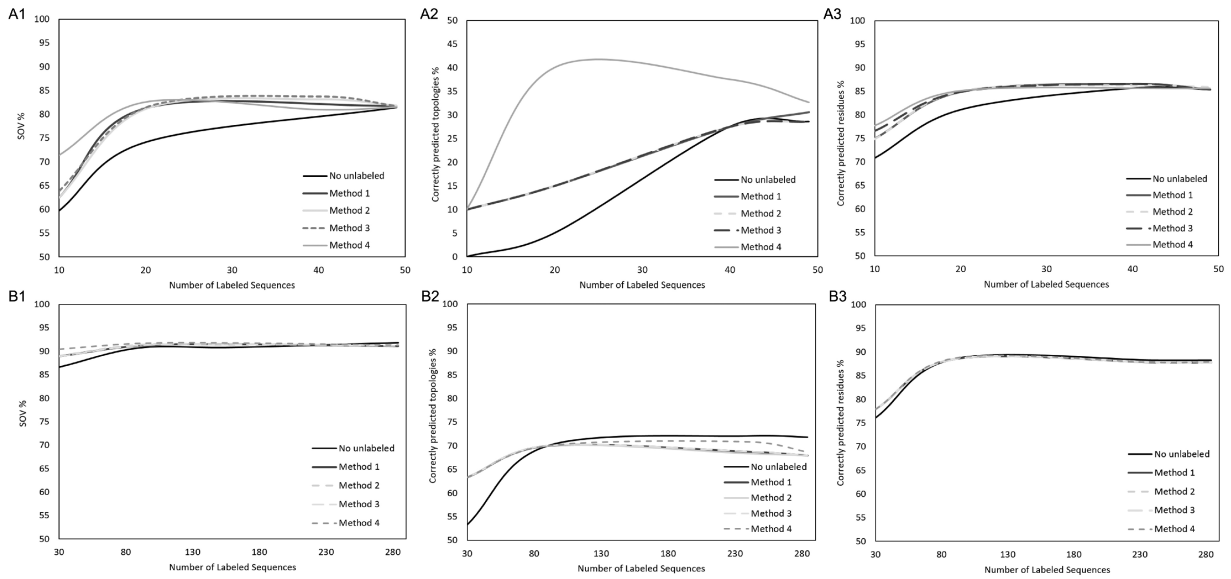

Fig. 2. Results from the 10-fold cross validation on transmembrane protein topology prediction by varying the size of the labeled dataset. (**A1**) beta-barrel outer membrane proteins (SOV). (**A2**) beta-barrel outer membrane proteins (correctly predicted topologies). (**A3**) beta-barrel outer membrane proteins (correctly predicted residues). (**B1**) alpha-helical membrane proteins (SOV). (**B2**) alpha-helical membrane proteins (correctly predicted topologies). (**B3**) alpha-helical membrane proteins (correctly predicted residues). In all cases the lines are smoothed curves

in SOV ranges from 0.5 to 3.8%, the increase in the fraction of correctly predicted topologies reaches 10% and the increase in the fraction of correctly predicted residues reaches 1.8%. In the case of signal peptides, the labeled dataset was already small, so we did not vary this number. The parameter estimates obtained in the semi-supervised manner led to 100% classification accuracy, while the estimates based on the labeled data only reached 83.3% accuracy. The increase in the fraction of correctly predicted cleavage sites ranges from 12.5 to 25% depending on the method. Of the four different methods, we observed that Methods 1, 2 and 3 have slightly better performance than Method 4 which uses only the unlabeled data with reliabilities passing a certain cutoff.

We also considered the effect of varying the amount of unlabeled data. We chose to keep a constant and rather small number of labeled sequences (25 for beta-barrel membrane proteins and 20 for alpha-helical ones) and varied the number of unlabeled ones. In general, we expected that the size of the unlabeled dataset will play a role, but we wanted to quantify its effect especially taking the different properties of the implemented methods into account. Figure 3 shows that, in the case of beta-barrel outer membrane proteins, even with as many as 100 sequences in the unlabeled dataset, Methods 1–3 that use all the unlabeled sequences, can increase the accuracy from 75.5% to almost 83% before reaching a plateau. Method 4, however, needs larger datasets in order to ensure that the most confident examples are found. For alpha-helical membrane proteins, the respective number at which this plateau occurs is rather high (300 sequences), but, nevertheless, in this case, all proposed methods behave similarly.

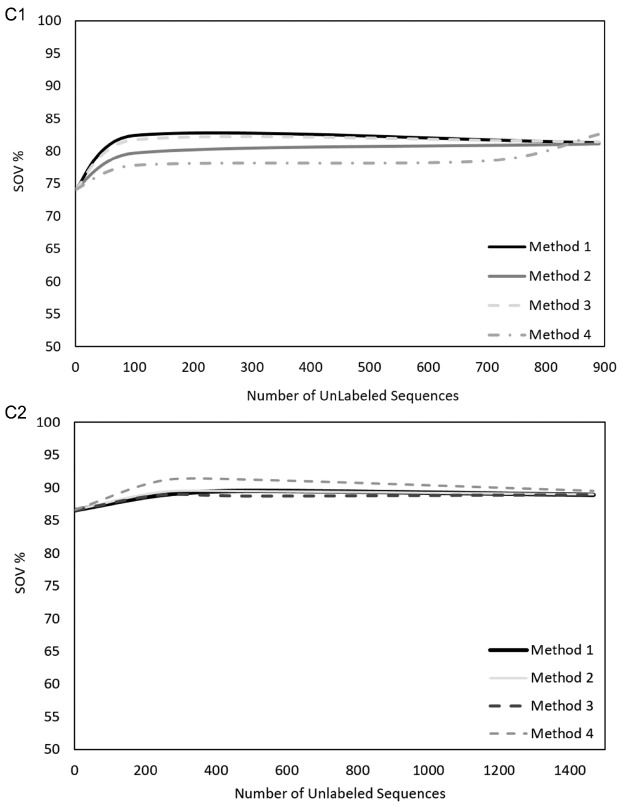

Fig. 3. Classification accuracy while varying the number of unlabeled sequences. (**C1**) Beta-barrel outer membrane proteins. (**C2**) Alpha-helical membrane proteins

4 Conclusions

In this work, we proposed a novel and efficient method for semi-supervised learning of HMMs in biological sequence analysis problems. The method is based on the self-training approach, extends previous works on training and decoding using labeled and partially labeled data (Bagos *et al.*, 2006; Krogh, 1994) and it is very easily implemented. The motivation of semi-supervised learning methods lies in the fact that, in many areas of research, labeled examples are relatively hard to find, whereas there are plenty of unclassified (unlabeled) data that can potentially be used to improve the

performance of the classifier. This situation is common in molecular biology, where the huge amount of protein sequences that are deposited in the public databases contradict the relatively few examples of proteins with experimentally verified structure or function. Thus, we expect that semi-supervised learning of HMMs will have many applications in this area.

Although in general the convergence properties of self-training are not easily studied, in this particular case we presented justification of the procedure in terms of the EM algorithm and we showed that the algorithms presented here are instances of the ECM algorithm which replaces the M-step with a sequence of conditional maximization (CM) steps in which each parameter is maximized individually, conditionally on the other parameters remaining fixed (Meng and Rubin, 1993). Following the ECM framework, the convergence properties of the algorithm are guaranteed. However, the algorithm may converge to a local maximum and, as in every EM and EM-like algorithm, the starting values are crucial. This is highlighted in the initial step, where the completely labeled data are used to build the initial model. It is easily understood that, if we have very few or low-quality labeled data, the initialization will provide an inadequate classifier and the whole procedure will lead to poor performance. Nevertheless, in our applications, even with a small dataset, the algorithms appear robust.

We applied the proposed method to several problems of biological sequence analysis with encouraging results. More specifically, we focused on two of the most successful applications of HMMs in bioinformatics, namely the transmembrane topology prediction and the prediction of signal peptides. We evaluated some of our previously methods by varying the amount of unlabeled sequences as well as the amount of labeled sequences. Results show significant improvements in classification by using unlabeled and partially labeled data. The proposed algorithms, when the quantity of high-quality training data is rather small, perform always better compared to standard supervised methods, but the gain in accuracy is greater when the labeled data are limited and the unlabeled data abundant. In the opposite case, i.e. when the number of labeled sequences exceeds a certain amount, the algorithm sometimes had the same or a minor effect on the classification performance. Although the variants that we described have different properties, in practice, the differences in performance are minimal. Methods 1 and 4 are more easily implemented and can be applied by re-training any existing prediction method that uses HMMs, whereas Methods 2 and 3 require some modifications to the Forward and Backward algorithms. On the other hand, Method 4 is more restrictive since the optimal threshold needs to be identified, but has the advantage of fewer computations as compared to Methods 1–3, since only a fraction of the unlabeled data is included in the training phase. Nevertheless, the comparative performance of each method needs to be identified in a case-by-case basis, since each particular problem with different labeled and unlabeled sequences may exhibit different properties. In general, we can see that Methods 1–3 have almost the same performance in all situations. This is plausible since these algorithms use all data, i.e. use the same amount of information for semi-supervised training. In some cases, when random fragmented sets of the labeled data are used along with a vast amount of unlabeled data, noise may be added to the results. On the other hand, when very few high-quality data are available, the semi-supervised techniques offer better models. Method 4, which uses only the unlabeled data with reliabilities passing a certain cutoff, performs slightly worse, e.g. in the case of beta-barrels and better in the case of alpha-helical ones. Nevertheless, this performance and the exact gain in accuracy are problem-dependent and the user is advised to explore all possibilities. The method presented here can be applied to other problems as well (including gene-finding, prediction of protein sorting signals, prediction of functional sites in proteins, prediction of transcription start sites and so on), using standard HMM algorithms and/or he modified algorithms developed for constrained prediction.

Conflict of Interest: none declared.

References

Abney,S. (2004) Understanding the Yarowsky Algorithm. *Comput. Linguist.*, **30**, 365–395.

Asai,K. *et al.* (1993) Prediction of protein secondary structure by the hidden Markov model. *Comput. Appl. Biosci.*, **9**, 141–146.

Bagos,P.G. *et al.* (2004) Faster gradient descent conditional maximum likelihood training of Hidden Markov Models, using individual learning rate adaptation. In: Paliouras,G. and Sakakibara,Y. (eds) *Grammatical Inference: Algorithms and Applications*. Spinger, Berlin/Heidelberg, pp. 40–52.

Bagos,P.G. *et al.* (2004) A Hidden Markov Model method, capable of predicting and discriminating beta-barrel outer membrane proteins. *BMC Bioinformatics*, **5**, 29.

Bagos,P.G. *et al.* (2005) Evaluation of methods for predicting the topology of beta-barrel outer membrane proteins and a consensus prediction method. *BMC Bioinformatics*, **6**, 7.

Bagos,P.G. *et al.* (2006) Algorithms for incorporating prior topological information in HMMs: application to transmembrane proteins. *BMC Bioinformatics*, **7**, 189.

Bagos,P.G. *et al.* (2008) Prediction of lipoprotein signal peptides in Gram-positive bacteria with a Hidden Markov Model. *J. Proteome Res.*, **7**, 5082–5093.

Bagos,P.G. *et al.* (2009a) How many 3D structures do we need to train a predictor? *Genomics Proteomics Bioinf.*, **7**, 128–137.

Bagos,P.G. *et al.* (2009b) Prediction of signal peptides in archaea. *Protein Eng. Des. Sel.*, **22**, 27–35.

Baldi,P. and Chauvin,Y. (1994) Smooth on-line learning algorithms for Hidden Markov Models. *Neural Comput.*, **6**, 305–316.

Baum,L. (1972) An inequality and associated maximization technique in statistical estimation for probabilistic functions of Markov processes. *Inequalities*, **3**, 1–8.

Chapelle,O. *et al.* (2006) *Semi-Supervised Learning. Adaptive Computation and Machine Learning*. MIT Press, London, UK.

Dempster,A.P. *et al.* (1977) Maximum likelihood from incomplete data via the EM algorithm. *J. R. Stat. Soc. B*, **39**, 1–38.

Dobson,L. *et al.* (2015) CCTOP: a Consensus Constrained TOPology prediction web server. *Nucleic Acids Res.*, **43**, W408–W412.

Durbin,R. *et al.* (1998) *Biological Sequence Analysis, Probabilistic Models of Proteins and Nucleic Acids*. Cambridge University Press, Cambridge, UK.

Eddy,S.R. (1995) Multiple alignment using hidden Markov models. *Proc. Int. Conf. Intell. Syst. Mol. Biol.*, **3**, 114–120.

El-Manzalawy,Y. *et al.* (2016) PlasmoSEP: predicting surface-exposed proteins on the malaria parasite using semisupervised self-training and expert-annotated data. *Proteomics*, **16**, 2967–2976.

Fischer,B. *et al.* (2006) Semi-supervised LC/MS alignment for differential proteomics. *Bioinformatics*, **22**, e132–e140.

Hafez,D. *et al.* (2017) McEnhancer: predicting gene expression via semi-supervised assignment of enhancers to target genes. *Genome Biol.*, **18**, 199.

Inoue,M. and Ueda,N. (2003) Exploitation of unlabeled sequences in Hidden Markov Models. *IEEE Trans. Pattern Anal. Mach. Intell.*, **25**, 1570–1581.

Ji,S. *et al.* (2008) Semisupervised learning of hidden Markov models via a homotopy method. *IEEE Trans. Pattern Anal. Mach. Intell.*, **31**, 275–287.

Juang,B.H. and Rabiner,L.R. (1990) The segmental K-means algorithm for estimating parameters of Hidden Markov Models. *IEEE Trans. Acoustics Speech Signal Process.*, **38**, 1639–1641.

Juncker,A.S. *et al.* (2003) Prediction of lipoprotein signal peptides in Gram-negative bacteria. *Protein Sci.*, **12**, 1652–1662.

Kall,L. *et al.* (2007) Semi-supervised learning for peptide identification from shotgun proteomics datasets. *Nat. Methods*, **4**, 923–925.

Kall,L. *et al.* (2004) A combined transmembrane topology and signal peptide prediction method. *J Mol. Biol.*, **338**, 1027–1036.

Kall,L. *et al.* (2005) An HMM posterior decoder for sequence feature prediction that includes homology information. *Bioinformatics*, **21**, i251–i257.

Krogh,A., (1994) Hidden Markov models for labelled sequences. In *Proceedings of the12th IAPR International Conference on Pattern Recognition*, pp. 140–144.

Krogh,A. (1997) Two methods for improving performance of an HMM and their application for gene finding. *Proc. Int. Conf. Intell. Syst. Mol. Biol.*, **5**, 179–186.

Krogh,A. *et al.* (2001) Predicting transmembrane protein topology with a hidden Markov model: application to complete genomes. *J. Mol. Biol.*, **305**, 567–580.

Krogh,A. *et al.* (1994) A hidden Markov model that finds genes in E. coli DNA. *Nucleic Acids Res.*, **22**, 4768–4778.

Krogh,A. and Riis,S.K. (1999) Hidden neural networks. *Neural Comput.*, **11**, 541–563.

Litou,Z.I. *et al.* (2008) Prediction of cell wall sorting signals in gram-positive bacteria with a hidden markov model: application to complete genomes. *J. Bioinform. Comput. Biol.*, **06**, 387–401.

Melen,K. *et al.* (2003) Reliability measures for membrane protein topology prediction algorithms. *J. Mol. Biol.*, **327**, 735–744.

Meng,X.L. and Rubin,D.B. (1993) Maximum likelihood estimation via the ECM algorithm: a general framework. *Biometrika,* **80**, 267–278.

Moller,S. *et al.* (2001) Evaluation of methods for the prediction of membrane spanning regions. *Bioinformatics*, **17**, 646–653.

Nielsen,H. and Krogh,A. (1998) Prediction of signal peptides and signal anchors by a hidden Markov model. *Proc. Int. Conf. Intell. Syst. Mol. Biol.*, **6**, 122–130.

Nigam,K. *et al.* (2000) Text classification from labeled and unlabeled documents using EM. *Mach. Learn.*, **39**, 103–134.

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

Scheffer,T. *et al.* (2001) Active Hidden Markov Models for information extraction. In: Hoffman,F. (ed), *IDA 2001*. Springer-Verlag, London, UK, pp. 309–318.

Shah,A.R. *et al.* (2008) SVM-HUSTLE—an iterative semi-supervised machine learning approach for pairwise protein remote homology detection. *Bioinformatics*, **24**, 783–790.

Tamposis,I.A. *et al.* (2018) Extending Hidden Markov Models to allow conditioning on previous observations. *J. Bioinf. Comput. Biol*, **16**, 18500191.

Tsaousis,G.N. *et al.* (2010) ExTopoDB: a database of experimentally derived topological models of transmembrane proteins. *Bioinformatics*, **26**, 2490–2492.

Tsirigos,K.D. *et al.* (2011) OMPdb: a database of β-barrel outer membrane proteins from Gram-negative bacteria. *Nucleic Acids Res.*, **39**, D324–D331.

Tsirigos,K.D. *et al.* (2016) PRED-TMBB2: improved topology prediction and detection of beta-barrel outer membrane proteins. *Bioinformatics*, **32**, i665–i671.

Tsirigos,K.D. *et al.* (2015) The TOPCONS web server for consensus prediction of membrane protein topology and signal peptides. *Nucleic Acids Res.*, **43**, W401–W407.

Tusnady,G.E. and Simon,I. (2001) The HMMTOP transmembrane topology prediction server. *Bioinformatics*, **17**, 849–850.

Viklund,H. and Elofsson,A. (2004) Best alpha-helical transmembrane protein topology predictions are achieved using hidden Markov models and evolutionary information. *Protein Sci.*, **13**, 1908–1917.

Yarowsky,D. (1995) Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics*. Cambridge, MA, pp. 189–196.