BIOINFORMATICS ORIGINAL PAPER Vol. 27 no. 22 2011, pages 3123–3128
doi:10.1093/bioinformatics/btr549

Structural bioinformatics Advance Access publication October 3, 2011

Improving the detection of transmembrane β-barrel chains with N-to-1 extreme learning machines

Castrense Savojardo[1,2], Piero Fariselli[1,2,*] and Rita Casadio[1]

[1]Biocomputing Group, Department of Biology, University of Bologna CIRI-Health Science and Technology, 40126 Bologna,[2]Department of Computer Science, University of Bologna, 40127 Bologna, Italy

Associate Editor: Alfonso Valencia

ABSTRACT

Motivation: Transmembrane β-barrels (TMBBs) are extremely important proteins that play key roles in several cell functions. They cross the lipid bilayer with β-barrel structures. TMBBs are presently found in the outer membranes of Gram-negative bacteria and of mitochondria and chloroplasts. Loop exposure outside the bacterial cell membranes makes TMBBs important targets for vaccine or drug therapies. In genomes, they are not highly represented and are difficult to identify with experimental approaches. Several computational methods have been developed to discriminate TMBBs from other types of proteins. However, the best performing approaches have a high fraction of false positive predictions.

Results: In this article, we introduce a new machine learning approach for TMBB detection based on N-to-1 Extreme Learning Machines that significantly outperforms previous methods achieving a Matthews correlation coefficient of 0.82, a probability of correct prediction of 0.92 and a sensitivity of 0.73.

Availability: The method and the cross-validation sets are available at the web page http://betaware.biocomp.unibo.it/BetAware.

Contact: piero.fariselli@unibo.it

Received on June 3, 2011; revised on September 8, 2011; accepted on September 28, 2011

1 INTRODUCTION

Transmembrane β-barrel (TMBB) proteins cross the lipid bilayer with a series of β-strands arranged in a cylindrical geometry and forming a structure that resembles a barrel (Schulz, 2000). TMBBs can be membrane anchors or membrane-bound enzymes endowed with functions relevant to the entire cell metabolism and including active ion transport, passive nutrient intake and defense against attack proteins (Schulz, 2000). While all living organisms have transmembrane proteins organized into all α-helical bundles, TMBBs are presently found in the outer membranes of Gram-negative bacteria, mitochondria and chloroplasts. TMBBs are estimated to be encoded by as many as 2–3% of the genes in Gram-negative bacteria (Casadio *et al.*, 2003; Freeman and Wimley, 2010; Wimley, 2003). However, very few TMBB structures are available at atomic resolution from Gram-negative organisms [about 0.1% of all the structures from Gram-negative organisms in the Protein Data Bank (PDB), http://www.pdb.org/pdb/home/ home.do].

*To whom correspondence should be addressed.

Several computational methods have been developed to predict TMBBs from protein sequences. Two prediction problems can be addressed: (i) the prediction of protein topology (both strand localization along the protein sequence and orientation of the loops with respect to the membrane plane); and (ii) TMBB detection in genomes. When the interest focuses on the prediction of the protein topology, it has been shown (Bagos *et al.*, 2005) that the best performing methods are based on hidden Markov models (Bagos *et al.*, 2004; Bigelow *et al.*, 2004; Martelli *et al.*, 2002) or Grammatical Restrained Hidden Conditional Random Fields (Fariselli *et al.*, 2009).

The detection of TMMBs in a set of proteins, as large as the whole genome, is more difficult due to the cryptic nature of the TMBB structure as compared with that of the all α-membrane proteins (Wimley, 2002). Computational methods for TMBB detection can identify candidate genes in order to perform experimental validations. To address this task, a wide variety of algorithms has been developed including: approaches based on sequence homology detection (Remmert *et al.*, 2009), machine learning (Bigelow *et al.*, 2004; Casadio *et al.*, 2003; Gromiha and Suwa, 2006) and physicochemical properties of TMBBs (Freeman and Wimley, 2010; Wimley, 2002).

In this article, we tackle the problem of TMBB detection and significantly improve over existing algorithms taking advantage of previous works. In particular, we exploit the effort made by Freeman and Wimley (2010) in generating a reliable dataset and defining a thorough comparison with previous methods. Furthermore, we refine the approach of Pollastri and co-workers (Mooney *et al.*, 2011) by incorporating their N-to-1 method in an Extreme Learning Machine (ELM) framework (Huang *et al.*, 2006a).

2 N-TO-1 ELMS

2.1 N-to-1 neural networks

Recently, a new formulation of single hidden layer feedforward neural networks (SLFNs) aimed at encoding an entire protein sequence into a single object has been described by Pollastri and co-workers (Mooney *et al.*, 2011). The basic idea is to encode in a single hidden layer of a neural network the *piece-wise* information defined by all the segments generated by a sliding window over the protein sequence. In a more formal way, given a protein sequence of length N and a non-linear sigmoid activation function σ, it is possible to map the entire input sequence of length N into the single hidden layer vector $\vec{H}$ by 'enrolling' an input sliding window $\vec{X}$(centered into each

protein residue) as:

$$H_h = k\sum_{j=1}^{N}\sigma(<\vec{W}_h,\vec{X}_j>) \quad (1)$$

where H_h is the h-th element of the hidden neuron vector $\vec{H}$, $\vec{W}_h$ is the weight vector that connects the j-th input window $\vec{X}_j$ with the h-th hidden neuron and the brackets '<>' define the dot product of the two vectors (here we implicitly include in the weight matrix W also the bias thresholds of the hidden neurons). For sake of simplicity we set k equal to $1/N$. With this choice $\vec{H}$ represents the average hidden layer of the entire sequence of length N and it becomes independent of the sequence length.

For each protein of length N, the hidden layer encodes the information of N input vectors ($\vec{X}_j$), obtained by sliding a symmetric window of an odd length ($L=2n+1$) along the protein sequence, one residue at a time. The encoding vector $\vec{X}_j$ consists of $20*L$ components, where 20 is the number of residue types. $\vec{X}_j$ is computed starting from the position-specific score matrix (PSSM) as internally computed by PSI-BLAST (Altschul *et al.*, 1997). When the n most terminal residues (N- and C-termini) are encoded, the empty positions are set with zero values. For each protein we generate a PSSM by aligning its sequence against the Uniref90 dataset (www.uniprot.org/help/uniref).

The final network output (used to assign the prediction) for a given protein is obtained by considering $\vec{H}$ as a normal hidden layer:

$$O_k = \sigma(<\vec{\beta}_k,\vec{H}>) \quad (2)$$

where β is the weight matrix that connects the hidden layer $\vec{H}$ with the neural network output vector $\vec{O}$. Equations (1) and (2) allow to encode N different input windows into a single hidden layer, making it possible to treat a protein sequence as a single object (and independently of its length). However, differently from Pollastri and co-workers (Mooney *et al.*, 2011), we take advantage of an ELM approach to set the network junctions (Huang *et al.*, 2006a).

2.2 ELMs

The main idea of our approach is to couple the N-to-1 encoding with an ELM approach (Huang *et al.*, 2006a). One interesting theoretical point about neural networks with ELM is the fact that differently from many other learning methods their universal approximation capability has been proved (Huang and Chen, 2007, 2008; Huang *et al.*, 2006b). In the ELM context, the first layer of weights (W) is set with random values. Only the second layer (β) is linearly trained. The first layer generates a non-linear 'random' encoding of the entire protein by mapping in a feature space for each sliding window. Once the N-to-1 *random*-enrolling is generated, the basic ELM machinery is used to train the β learnable parameters.

Differently from what commonly thought, the input weights of SLFNs do not need to be learned (Huang, 2003). Since in ELM the hidden layer does not need to be tuned and the hidden layer parameters can be fixed, the output weights can then be resolved using a least-square method. In practice, taking advantage of the fact that only the output layer (β) is trainable, the non-linearity of Equation (2) can be removed and β values can be obtained by the fitting the targets as:

$$T = \beta \cdot H \quad (3)$$

where $T=[\sigma^{-1}(t_1,),\sigma^{-1}(t_2),\ldots\sigma^{-1}(t_N),]^T$ is the matrix of transformed labels of protein classification by means of the inverse of the activation function $\sigma^{-1}(x)=\log[(1-x)/x]$. In order to prevent overflow, classification labels [1 or 0 for x in $\sigma^{-1}(x)$] are substituted with real values (for instance, 0.99 for 1 and 0.01 for 0).

Our training algorithm can be stated as follows. Given a training set of protein sequences $\{P_i\}$ with their corresponding classification labels $\{t_i\}$, the learning algorithm proceeds as:

Step 1. Set the number of hidden neurons.

Step 2. Assign randomly the input weights W.

Step 3. For each protein P_i generate a vector $\vec{H}_i$ using the N-to-1 encoding of Equation (1) to obtain the final H matrix.

Step 4. Calculate the output weights β as:

$$\beta = H^{-1}T \quad (4)$$

where $T=[\sigma^{-1}(t_1,),\sigma^{-1}(t_2),\ldots\sigma^{-1}(t_N),]^T$ is as in Equation (3) and H^{-1} is the *Moore–Penrose* generalized inverse of the H matrix (Huang, 2003).

Differently from traditional learning algorithms, ELM not only tends to reach the smallest training error but also the smallest norm of output weights (Huang *et al.*, 2006a). This implies that the H^{-1} choice minimizes both:

$$\begin{aligned} &Minimize\|T-\beta H\| \\ &Minimize\|\beta\| \end{aligned} \quad (5)$$

In analogy with Support Vector Machines, minimizing the norm of the output weight $||\beta||$ is similar to maximizing the distance of the separating margins of the two different classes in the ELM feature space $2/||\beta||$ (Huang *et al.*, 2010).

Another invaluable advantage of ELMs with respect to the classical gradient-based algorithms is their speed. Published papers addressed this issue although on problems different from the one we address here and reported that accuracy is somewhat similar (in some cases slightly lower) when back-propagation is compared to ELM (Alhamdoosh *et al.*, 2010; Haung, 2006). For our application, the back-propagation version is at least 30 times slower with more hyper-parameters (learning rate, training cycles, etc.) to set (data not shown). The ELM speed during the learning phase allows to test a larger number of different models.

In our current implementation, the random weights (W) implicitly contain the threshold biases (set as uniform random values). On the contrary the weights of the second layers (β) do not contain biases, since it has been demonstrated that the hyper-plane crossing the origin is sufficient for universal function approximation (Huang and Chen, 2007, 2008; Huang *et al.*, 2006b).

Once a random set of weights is generated (W) and the trainable parameters (β) are set according to Equation (4), the predictions of a set of unknown proteins (or of the validation/test sets) are assigned using Equations (1) and (2).

3 DATASET AND MEASURES OF PERFORMANCE

3.1 Dataset

To train and test our method we rely on the dataset generated by Freeman and Wimley (2010). This dataset (NRPDB) consists of 14 238 chains derived from PDB, 48 of which are TMBBs from Gram-negative bacteria while 14 190 are non-TMBB proteins.

Since the set contains a significant amount of sequence similarity, we clustered chains by local similarity (computed with BLAST) and we created 10 different subsets that internally confine sequence homology. By this sequence identity between pairs of proteins belonging to two different subsets is <25% (this is both for TMBB and non-TMBB proteins).

3.2 Measuring the performance

Here TP, TN, FP and FN are, respectively, the true positives, the true negatives, the false positives and the false negatives with respect to the TMBB class. Performances are evaluated adopting the following scoring indexes:

- Accuracy (AC) evaluates the number of correctly predicted TMBB proteins divided by the total number of proteins:

$$\mathrm{AC} = \frac{\mathrm{TP}+\mathrm{TN}}{\mathrm{TP}+\mathrm{TN}+\mathrm{FP}+\mathrm{FN}} \tag{6}$$

- Specificity (SP) (sometimes this is also defined as sensitivity of the negative class) is the number of correctly predicted proteins in the non-β-transmembrane class divided by the total number of proteins in the class:

$$\mathrm{SP} = \frac{\mathrm{TN}}{\mathrm{TN}+\mathrm{FP}} \tag{7}$$

- Sensitivity (SN) is the number of correctly predicted proteins divided by the total number of observed TMBB proteins:

$$\mathrm{SN} = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}} \tag{8}$$

- Positive predicted value (PPV), also referred to as 'probability of correct prediction', measures the probability that TMBBs are correctly assigned among the predicted TMBBs. PVV is defined as the number of correctly predicted TMBBs divided by the total number of predicted TMBBs

$$\mathrm{PPV} = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FP}} \tag{9}$$

- F1is defined as the harmonic mean of PPV and SN:

$$\mathrm{F1} = \frac{2 \times \mathrm{PPV} \times \mathrm{SN}}{\mathrm{PPV}+\mathrm{SN}} \tag{10}$$

- The Matthews correlation coefficient (MCC) is:

$$\mathrm{MCC} = \frac{(\mathrm{TP} \times \mathrm{TN} - \mathrm{FP} \times \mathrm{FN})}{\sqrt{(\mathrm{TP}+\mathrm{FP}) \times (\mathrm{TP}+\mathrm{FN}) \times (\mathrm{TN}+\mathrm{FP}) \times (\mathrm{TN}+\mathrm{FN})}} \tag{11}$$

3.3 Model selection procedure

ELM models depend on some hyper-parameters (i.e. random initial weights, number of hidden neurons, the input window) that can influence the method performance and need to be optimized. For this and in order to avoid over-fitting, we carried out a 10-fold cross-validation procedure on the training set, based on a distinction among validation and test sets. In particular, given the 10 non-homologous subsets generated as described above from NRPDB, we trained on the 10 different learning sets as follows. Eight subsets at a time are adopted for training and the remaining two are used for validation and testing. Furthermore, in order to cope with the stochastic nature of ELMs, for each window/hidden-neuron pair, we generated five random sets of weights (W). Scoring values for model selection are obtained by averaging on the 10 different 'validation' subsets. More formally, if T_i is the i-th test set ($i=0\ldots9$), the corresponding validation set V_i and training set L_i are defined as $V_i = T_{(i+1) \bmod 10}$ and $L_i = \{T_{(i+2) \bmod 10}, T_{(i+3) \bmod 10}, \ldots T_{(i+9) \bmod 10}\}$, where $(k) \bmod j$ is the modulo operation that computes the remainder of division of k by j.

For the ensemble selection we adopted an exhaustive procedure based on MCC scores computed on the validation sets. First, we selected the N-to-1 ELM models whose MCC scored ≥0.77. Then for each combination of these models, we defined an 'ensemble' by averaging their predictions. Finally, the best performing ensemble on the validation sets was retained.

The final performance was assessed on the test sets without any further hyper parameter tuning.

Table 1. MCC scores on the validation sets as a function of the window width and of the number of hidden neurons

Win\#H	25	50	100	200	400	800	1600	3200
1	36 (2)	53 (2)	65 (1)	71 (1)	71 (1)	73 (1)	74 (1)	64 (1)
3	43 (5)	42 (3)	51 (2)	58 (1)	69 (2)	***78 (1)***	***77 (1)***	58 (1)
5	44 (3)	45 (3)	47 (4)	55 (2)	63 (2)	73 (1)	76 (2)	54 (2)
7	44 (2)	47 (3)	48 (1)	55 (3)	64 (2)	75 (2)	***78 (1)***	54 (3)
9	44 (4)	50 (4)	50 (2)	55 (1)	63 (2)	70 (3)	75 (2)	55 (1)
11	47 (5)	51 (1)	51 (3)	55 (2)	64 (1)	71 (2)	***77 (1)***	55 (2)
13	45 (2)	52 (2)	54 (3)	56 (3)	62 (2)	70 (3)	74 (2)	56 (3)
15	43 (3)	54 (1)	55 (4)	59 (5)	64 (1)	69 (2)	73 (2)	59 (5)
17	47 (3)	58 (5)	57 (5)	58 (3)	63 (2)	69 (3)	75 (2)	58 (3)
19	45 (5)	56 (1)	59 (3)	60 (3)	64 (3)	68 (1)	71 (2)	59 (3)
21	45 (5)	56 (1)	59 (3)	60 (3)	64 (3)	68 (1)	68 (2)	59 (3)

Win, width of the sliding input window; #H, number of hidden neurons. The numbers are in percentages with the SD reported within parentheses. The average value and SD are computed on five different random initializations. In bold–italic we list MCC scores ≥0.77.

4 RESULTS AND DISCUSSION

4.1 Performance of the ELM-based predictor

The aim of this article is to generate a reliable method for finding new TMBB proteins that do not share sequence similarity with those of our training sets. In order to properly train the method, it is necessary that sequence similarity between training and testing proteins is kept at a minimum (<25%). For this reason, we split NRPDB in 10 different subsets so that any pair of sequences selected from two different subsets does not have a local sequence similarity ≥25%. We performed a cross-validation procedure on the 10 different subsets distinguishing among validation and test sets as described in Section 3.3. Different ELM models, differing from each other in the number of neurons of the hidden layer and in the dimension of the sliding window, are evaluated. We report the MCC scores for each tested model on the validation subsets in Table 1. Each MCC score is obtained averaging over five different random initializations of the first layer of weights (W). It appears that MCC values are >0.70 for nearly any input window in the range of 800–1600 hidden neurons. In ELM, the first layer of weights is randomly set with values that are independent of the training examples (so that in our case the

Table 2. Performance of the ensemble and its components on the test sets

Method	MCC	SEN	SPE	PPV	AC	F1
ELM_w7/1600	77 (2)	64 (2)	100 (0)	92 (3)	100 (0)	75 (2)
ELM_w11/1600	75 (3)	59 (6)	100 (0)	99 (3)	99 (1)	73 (4)
ELME	82 (2)	73 (4)	99 (1)	92 (3)	99 (1)	81 (3)

ELM_wx/1600, the single N-to-1 ELM model with window size x and 1600 hidden neurons; ELME, predictor that averages the predictions of the selected ELM_wx models. For index definition see Section 3.2. Numbers are in percentages and SD are reported within parentheses.

number of hidden neurons is the number of parameters to set). For sake of comparison, a classical network with an input window of seven residues (and the same encoding of ELM) supplied with only 20 neurons in the hidden layers needs to train >2800 parameters. The number of training examples in our application is one order of magnitude higher than the number of optimal hidden neurons, ensuring that training can be properly accomplished. Moreover, the minimal norm associated to the ELM training (see Section 2.2) strongly reduces the over-fitting problems. In Table 1 we also list (within parentheses) values of MCC standard deviations (SDs) for all the window/hidden neuron combinations. The small variations around the average obtained indicate that the method is quite robust and insensitive to the random initialization. Notably, this is true also for a single residue long sliding window, indicating that N-to-1 ELMs are able to detect significant differences between TMBBs and other proteins even encoding a single residue at a time in the hidden layer.

Among the four models that obtained an average MCC score ≥0.77 in Table 1, we computed all their possible ensembles as:

$$O_E = \frac{1}{N}\sum_{x=1}^{N} O_x \qquad (12)$$

where O_x is the output of the x-th selected model. The models for the ensemble were selected on the validation sets, using the criterion described above (see Section 3.3). After an exhaustive search we ended up with an ensemble of two models that on the validation sets achieved an MCC score of 0.83. The two models singled out for the ensemble have both 1600 neurons in the hidden layer and input windows of 7 and 11 residues, respectively.

The selected ensemble (ELME), together with its constituent models is then evaluated on the test sets (Table 2). Although scores obtained on the test sets are slightly lower than those computed using the validation sets (compare Table 1 with Table 2), ELME performances are still very high achieving an MCC score of 0.82 with SD equal to 0.02 (Table 2).

4.2 Comparison with previous methods

Freeman and Wimley (2010) thoroughly compare performances of their algorithms with available methods (also based on machine learning approaches) on their newly generated dataset (NRPDB comprising 14 238 proteins, of which 48 are TMBBs). Thanks to their effort, here we can compare our results with their best methods under the same condition: we set a predictive threshold so that 46 or 37 of 48 true TMBBs are considered positive predictions. In Table 3 we compare our ELME with the two best algorithms obtained by Freeman and Wimley (2010). In particular, they defined

Table 3. ELME comparison with the best Freeman–Wimley algorithms

Method	TP	FP	TN	FN	MCC	SN	SP	AC	PPV	F1
FW[a]	46	599	13 591	2	0.26	0.96	0.96	0.96	0.07	0.13
MRS[a]	37	161	14 029	11	0.38	0.77	0.99	0.99	0.19	0.30
ELME[b](1)	46	88	14 102	2	0.57	0.96	0.99	0.99	0.34	0.51
ELME[b](2)	37	12	14 178	11	0.76	0.77	0.99	0.99	0.76	0.76

We changed the ELME predictive thresholds in order to match the same number of TPs [(1), threshold set to obtain 46 TPs; (2), threshold set to obtain 37 TPs] of FW and MRS, respectively.

[a]Results taken from Freeman and Wimley (2010); FW, Freeman–Wimley algorithm with all modifications included (Freeman and Wimley, 2010); MRS, FW algorithm with MRS filter (Freeman and Wimley, 2010).

[b]Results are obtained using a 10-fold cross-validation procedure. For index definition see Section 3.2.

a new algorithm (FW in Table 3) including several improvements over a previously published implementation (Wimley, 2002) and a filtered version based on a mean randomized score (MRS in Table 3).

Scores in Table 3 for FW and MRS were taken from Freeman and Wimley (2010). In that paper, FM and MRS performances were computed using some of the TMBB proteins included in the test sets to fit the main algorithm parameters (TMBB amino acid abundance values). Although NRPDB includes sequences that were used to generate abundance values for the FW algorithm, the authors explained that this was not a problem since their statistical approach was not readily subject to over-fitting due to the nature of the method and to the low number of parameters of the FW algorithm (120) (Freeman and Wimley, 2010). Machine learning approaches, however, need cross-validation to prove the robustness of the learning process and for this reason we adopt a 10-fold cross-validation procedure based on a distinction among validation and testing sets. Results reported in Table 3 for ELME were obtained using the 10-fold cross-validation procedure by averaging the scores of the 10 predictions (two models with five different initial random weights) obtained for each protein on its test sets and by adopting a predictive threshold to match the true positive values of FW and MRS. ELME outperforms both methods (Table 3) when the number of true positives is the same. In either case with ELME the number of false positives is drastically reduced and the PPV and MCC values are significantly improved.

The performance of ELME can be also evaluated in comparison with other available methods under the conditions set by Freeman and Wimley (2010), namely after matching the sensitivity values of the different algorithms based on machine learning (kNN, Hu and Yan, 2008; TMDB, Ou *et al.*, 2008) and pattern recognition (BOMP, Berven *et al.*, 2004) (Table 4). Also in this benchmark, ELME achieves the highest scores.

4.3 Genomic analysis

One of the main purposes of our N-to-1 ELM algorithm was to sort out TMBB proteins in genomic databases. We evaluated our method on the genome of *Escherichia coli* (K12 strain), which is one of the most comprehensively annotated genomes available. From UniProt we extracted the proteins from *E.coli* K12, which were not annotated as hypothetical or putative. Similarly of what has been done before by other authors, we did not consider proteins that were shorter than 60 or longer than 4000 residues (Freeman and Wimley, 2010).

Table 4. ELME comparison with different approaches on NRPDB

Index	kNN[a]/FW[a]/ELME	TMBD[a]/FW[a]/ELME	BOMP[a]/FW[a]/ELME
Sensitivity	85.4/85.4/85.4	95.8/95.8/95.8	81.2/81.2/81.2
Specificity	97.4/99.1/99.8	93.6/95.8/99.3	98.4/99.1/99.9
Accuracy	97.4/99.0/99.8	93.7/95.8/99.3	98.4/99.1/99.8
MCC[b]	0.289/0.45/0.73	0.208/0.257/0.57	0.342/0.441/0.74

BOMP, Berven *et al.* (2004); FW, Freeman and Wimley (2010); kNN, Hu and Yan (2008); TMBD, Ou *et al.* (2008); ELME, this work.
[a]Values are taken from Freeman and Wimley (2010).
[b]Results are obtained using a 10-fold cross-validation procedure. For index definition see Section 3.2.

Table 5. N-to-1 ELM performance on *E.coli* genome

Method	MCC	SEN	SPE	PPV	AC	F1
ELM w7/1600	93 (1)	93 (2)	99 (1)	93 (2)	99 (1)	93 (1)
ELM w11/1600	93 (1)	90 (2)	99 (1)	96 (2)	99 (1)	93 (1)
ELME	95 (1)	90 (2)	100 (1)	100 (1)	99 (1)	95 (1)

For the legend see Table 2.

In particular, we selected the annotated TMBB proteins and the negative complement set using the following queries:

- organism:83333 AND keyword:181 AND reviewed:yes AND keyword:'Transmembrane beta strand' AND length:[60 TO 4000] NOT name:Probable NOT name:Putative NOT name: Uncharacterized NOT annotation:(type:location lipid anchor)
- organism:83333 AND keyword:181 AND reviewed:yes NOT keyword:'transmembrane beta strand' AND length:[60 TO 4000] NOT name:Probable NOT existence:'inferred from homology' NOT existence:predicted NOT existence:uncertain NOT annotation:(type: location 'Cell outer membrane') NOT name:Putative

With this selection we ended up with 30 TMBBs and 2515 non-TMBBs. Since there is some degree of similarity between the *E.coli* proteins and those in NRPDB, we adopted the following procedure to asses the genomic predictions:

- for each *E.coli* protein q, we run a BLAST search of q against NRPDB,
- we extracted the most similar sequence p from NRPDB (the highest BLAST hit of q in NRPDB),
- we selected the 1-to-N ELM models generated by the 10-fold cross-validation on NRPDB, from which p belonged to the 'test set',
- we predicted q with those models (whose parameters and hyperparameters were chosen in the validation set).

With this procedure, we simulated the case of a genomic analysis on never-seen before proteins. The results on *E.coli* set are reported in Table 5, where it appears that the N-to-1 ELME is more efficient in analyzing the *E.coli* annotated proteins than in scoring over NRPDB (compare Tables 2 and 5).

5 CONCLUSIONS

In this article, we present a new algorithm for the detection of TMBB proteins starting from their sequence. The main novelties of our work are of two kinds: methodological and applicative.

First of all, here we introduce a new very powerful and general method that combines two ideas: an N-to-1 whole protein encoding in a single hidden layer (Mooney *et al.*, 2011) and an ELM approach (Huang, 2003). Our new method is here applied to the detection of TMBB proteins. However, N-to-1 ELM models are very general and can be profitably applied to address other problems of computational biology.

In this article, we also show that the N-to-1 ELM approach outperforms previously developed methods of TMBB detection, including the most recent one (Freeman and Wimley, 2010). Differently from methods based on biochemical principles, the parameters of an N-to-1 ELM do not have an immediate physicochemical description. This can be a disadvantage when for the task at hand, a clear interpretation of the method parameter is needed. However, when the main goal is achieving the best predictive performance, a bottom up approach, such as learning rules from data mining, can be advantageous.

Funding: Ministero dell'Istruzione, dell'Università e della Ricerca (MIUR)-Fondo per gli Investimenti della Ricerca di Base (FIRB) grant for the Laboratorio Interazionale di BIoinformatica (LIBI) project delivered to R.C.

Conflict of Interest: none declared.

REFERENCES

Alhamdoosh,M. *et al*. (2010) Disulfide connectivity prediction with extreme learning machines. In *Proceeding of the International Conference on Bioinformatics Models, Methods and Algorithms*. SciTePress, Rome, January 26–29, pp. 1–10.

Altschul,S.F. *et al.* (1997) Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Res.*, **25**, 3389–3402.

Bagos,P.G. *et al.* (2004) PRED-TMBB: a web server for predicting the topology of beta-barrel outer membrane proteins. *Nucleic Acids Res.*, **32**, W400–W404.

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**, 1–13.

Berven,F.S. *et al.* (2004) BOMP: a program to predict integral β-barrel outer membrane proteins encoded within genomes of Gram-negative bacteria. *Nucleic Acids Res.*, **32**, W394–W399.

Bigelow,H.R. *et al.* (2004) Predicting transmembrane beta-barrels in proteomes. *Nucleic Acids Res.*, **32**, 2566–2577.

Casadio,R. *et al.* (2003) Fishing new proteins in the twilight zone of genomes: the test case of outer membrane proteins in Escherichia coli K12, Escherichia coli O157:H7, and other Gram-negative bacteria. *Protein. Sci.*, **11**, 1158–1168.

Fariselli,P. *et al.* (2009) Grammatical-restrained hidden conditional random fields for bioinformatics applications. *Algorithms Mol. Biol.*, **22**, 4–13.

Freeman,T.C. and Wimley,W.C. (2010) A highly accurate statistical approach for the prediction of transmembrane β-barrels. *Bioinformatics*, **26**, 1965–1974.

Gromiha,M.M. and Suwa,M. (2006) Discrimination of outer membrane proteins using machine learning algorithms. *Proteins*, **63**, 1031–1037.

Huang,G.B. (2003) Learning capability and storage capacity of two hidden-layer feedforward networks, *IEEE T. Neural Networ.*, **14**, 274–281.

Huang,G.B. *et al.* (2006a) Extreme learning machine: theory and applications. *Neurocomputing*, **70**, 489–501.

Huang,G.B. *et al.* (2006b) Universal approximation using incremental constructive feedforward networks with random hidden nodes. *IEEE T. Neural Networ.*, **17**, 879–892.

Huang,G.B. and Chen, L. (2007) Convex incremental extreme learning machine. *Neurocomputing*, **70**, 3056–3062.

Huang,G.B. and Chen, L. (2008) Enhanced random search based incremental extreme learning machine. *Neurocomputing*, **71**, 3460–3468.

Hu,J. and Yan,C. (2008) A method for discovering transmembrane β-barrel proteins in Gram-negative bacterial proteomes. *Comput. Biol. Chem.*, **32**, 298–301.

Huang,G.B. *et al.* (2010) Optimization method based extreme learning machine for classification, *Neurocomputing*, **74**, 155–163.

Martelli,P.L. *et al.* (2002) A sequence-profile-based HMM for predicting and discriminating beta barrel membrane proteins. *Bioinformatics*, **18**, S46–S53.

Mooney,Y. *et al.* (2011) De novo protein subcellular localization prediction by N-to-1 neural networks, In Lisboa,P.J. and Rizzo,R. (eds) *Computational Intelligence Methods for Bioinformatics and Biostatistics*. LNCS 6685, Springer, p. 31.

Ou,Y.Y. *et al.* (2008) TMBETADISC-RBF: discrimination of β-barrel membrane proteins using RBF networks and PSSM profiles. *Comput. Biol. Chem.*, **32**, 227–231.

Remmert,M. *et al.* (2009) Hhomp: prediction and classification of outer membrane proteins. *Nucleic Acids Res.*, **37**, W446–W451

Wimley,W.C. (2002) Toward genomic identification of β-barrel membrane proteins: composition and architecture of known structures. *Protein Sci.*, **11**, 301–312.

Wimley,W.C. (2003). The versatile beta-barrel membrane protein. *Curr. Opin. Struct. Biol.*, **13**, 404–411.

Schulz,G.E. (2000) β-Barrel membrane proteins. *Curr. Opin. Struct. Biol.*, **10**, 443–447.