Finding Signal Peptides in Human Protein Sequences Using Recurrent Neural Networks

Martin Reczko[1], Petko Fiziev[2], Eike Staub[2], and Artemis Hatzigeorgiou[3]

[1] Synaptic Ltd., Science and Technology Park of Crete
P.O. Box 1447, 711 10 Voutes Heraklion, Greece
[2] metaGen Pharmaceuticals GmbH, Oudenader Str.16, D-13347 Berlin, Germany
[3] Department of Genetics, University of Pennsylvania, School of Medicine
Philadelphia, PA 19104-6145, USA

Abstract. A new approach called Sigfind for the prediction of signal peptides in human protein sequences is introduced. The method is based on the bidirectional recurrent neural network architecture. The modifications to this architecture and a better learning algorithm result in a very accurate identification of signal peptides (99.5% correct in fivefold cross-validation). The Sigfind system is available on the WWW for predictions (http://www.stepc.gr/ synaptic/sigfind.html).

1 Introduction

In the last few years the complete or nearly the complete sequence for a large number of genomes including also the human genome has been determined. One of the expected consequences from sequencing these genomes is the discovery of new drugs. The sorting of subcellular proteins is a fundamental aspect of finding such drugs, because such proteins have the potential to leave the cell and move through the organism.

In humans secretory proteins account for about one-tenth of the human proteome. As many functional characteristics of proteins can be correlated with more or less well-defined linear motifs in their aminoacid sequences, also in this case sorting depends on *signals* that can already be identified in the primary structure of a protein.

Signal peptides are N-terminal extensions on the mature polypeptide chain. They are cleaved from the mature part of the protein at the *cleavage site*. The basic design of signal peptides is constituted by three distinct regions: a short positively charged N-terminal domain (N-region) a central hydrophobic stretch of 7-15 residues (H-region) and a more polar C-terminal part of 3-7 residues that defines a processing site for the signal peptidase enzyme (C-region) [18].

Prediction of signal peptides has a long history starting back in early eighties by simple algorithms based on hydrophobicity calculations [8] and the later use of a positional weight matrix for the cleavage site [17]. The application of more sophisticated methods such as rule-based systems [7], neural networks (NNs) [13], hidden Markov models (HMMs) [14] and linear programming [9] has increased the levels of accuracy considerably. A more detailed review can be found in [12].

R. Guigó and D. Gusfield (Eds.): WABI 2002, LNCS 2452, pp. 60–67, 2002.

In this paper a new method for defining signal peptides is introduced. The method is based on a novel neural network architecture called bidirectional recurrent neural nets (BRNN) that was recently introduced by [2]. The BRNN architecture is used for sequential learning problems with sequences of finite lengths. Other recurrent NNs can only model causal dynamical system, where the output of the system at a certain time does not depend on future inputs. The BRNN model has a symmetric memory for storing influences of past and future inputs to an output at time t.

This non causal behavior is useful for the prediction of properties of biosequences, since most properties are influenced by elements of those sequences both upstream and downstream of that property. The concept of time does not apply to this class of sequential learning problems, so we know about the 'future' sequence following a sequence position t and the system is not violating any physical causality.

We modify the architecture and introduce the combination with a more efficient learning algorithm that produces networks with better generalization performance in less computational time. In the results section we present a comparison with the methods available on the Internet and we are documenting the achieved improvements.

2 Material and Methods

The data used here is the same data as was used for the first SignalP prediction system [13] that was made available by the authors[1]. That data was derived from SWISS-PROT version 29 [1]. For the removal of redundancy in their data set, Nielsen et. al. [13] eliminated any pair of sequences with more than 17 identical residues in a local alignment.

From this non-redundant data set we use only the human sequences which contain 416 signal peptides and the N-terminal part of 97 cytoplasmic and 154 nuclear proteins as negative examples.

2.1 Neural Network Architecture

The neural network architecture employed for the task is a modified version of the BRNN as described in [2]. The BRNN architecture is summarized here and the modifications are introduced.

The state vectors are defined as F_t and B_t in $\mathbb{R}^n$ and $\mathbb{R}^m$ and contain the context information while reading forward and backwards, respectively. They are calculated as:

$$F_t = \phi(F_{t-1}, U_t) \qquad \text{and} \tag{1}$$

$$B_t = \beta(B_{t+1}, U_t) \tag{2}$$

[1] The SignalP datasets are available at: ftp://virus.cbs.dtu.dk/pub/signalp

where $\phi()$ and $\beta()$ are nonlinear transition functions realized by the multilayer perceptrons (MLPs) $\aleph_\phi$ and $\aleph_\beta$ and the vector $U_t \in \mathbb{R}^k$ is the input at time $t \in [1, T]$. The state vectors are initialized with $F_0 = B_{T+1} = 0$.

The output $Y_t \in \mathbb{R}^s$ is calculated after the calculation of F_t and B_t has finished using

$$Y_t = \eta(F_t, B_t, U_t) \tag{3}$$

where $\eta()$ is again realized by a MLP $\aleph_\eta$. In the realization of [2], the MLP $\aleph_\eta$ contains one hidden layer that is connected to U_t and the state units in F_t and B_t are connected directly to the output layer.

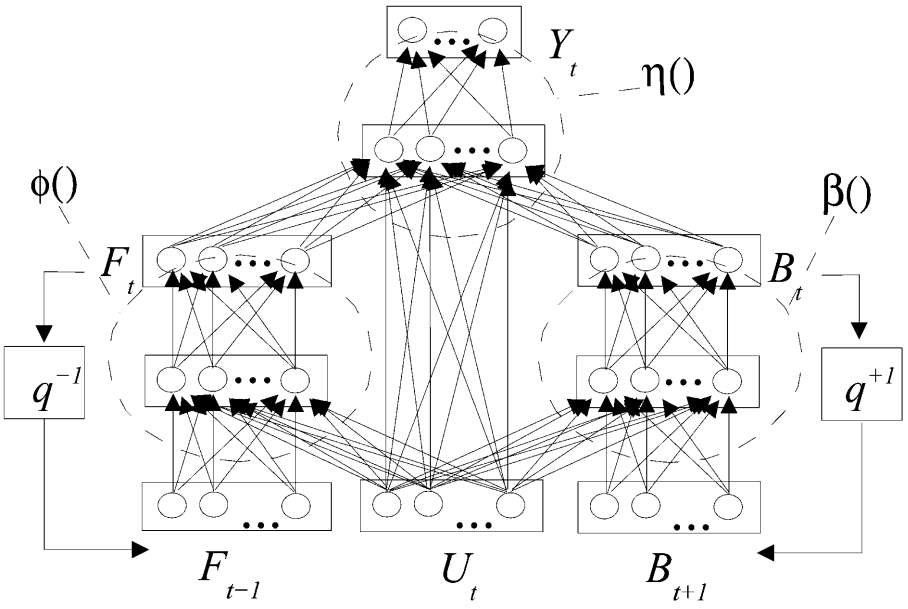

Fig. 1. The modified bidirectional recurrent neural network architecture. The input at time t is applied at U_t. Y_t contains the output vector. F_t is the forward state vector, and B_t the backward state vector. The shift operator q^{-1} copies the last F_t state vector while reading the input sequence forwards while the operator q^{+1} copies the last B_t state vector while reading the input backwards.

In our modified BRNN, the state units are not connected directly to the output, but to the hidden layer. It is reasonable to assume that several activation patterns occurring on the combination of the forward and backward state units are not linearly separable and thus cannot be distinguished without the additional transformation performed by this hidden layer. There is an implicit asymmetry in the amount of context information accumulated in the state vectors. At the start of the sequence ($t = 1$), the forward state vector F_1 contains

information about only one input vector U_1 whereas the backward state vector B_1 has accumulated all information of the complete sequence. At the end of the sequence ($t = T$), the opposite situation occurs. The processing of the state vectors with a hidden layer can help detecting these asymmetric situations and process the more relevant activations.

The architecture is shown in figure 1 where the shift operator q^{-1} to copy the forward state vector is defined as $X_{t-1} = q^{-1}X_t$ and the inverse shift operator q^{+1} to copy the backward state vector is defined as $X_{t+1} = q^{+1}X_t$.

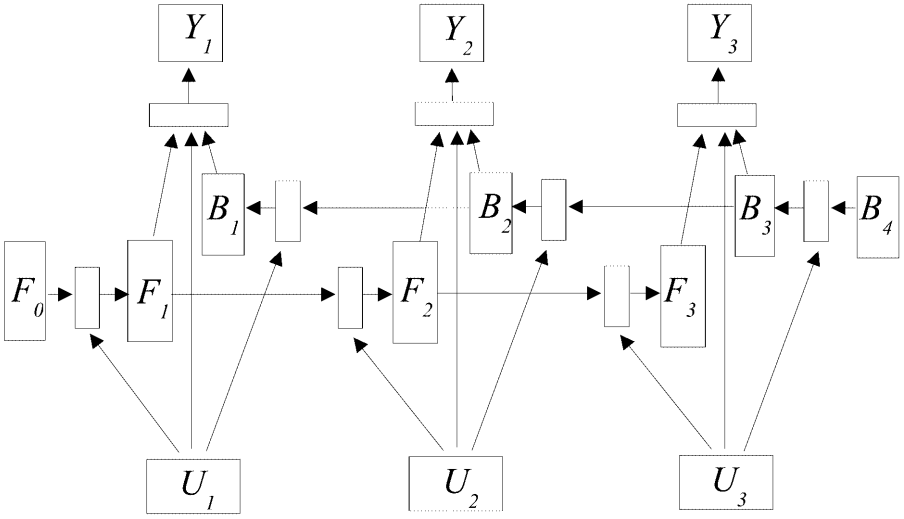

Fig. 2. The unfolded feedforward structure of a BRNN processing an input sequence U_t with length $T = 3$. First, the forward states $F_0, \ldots, F_3$ are calculated sequentially from left to right, while the backward states $B_4, \ldots, B_1$ are calculated beginning with the end of the sequence and continuing from right to left. After all F_t and B_t are processed, the output sequence Y_t can be calculated.

2.2 Learning Algorithm

To adapt the weight parameters in the MLPs $\aleph_\eta$,$\aleph_\phi$ and $\aleph_\beta$ the recurrent NN is unfolded in time and the backpropagation algorithm [16] is used to calculate the gradient for all weights. The unfolding is illustrated in figure 2. This transformation of a recurrent NN into a equivalent feedforward NN is called backpropagation through time and was first described in [11] and the application of backpropagation learning to these networks was introduced in [16].

The gradient for a weight is summed up for all training sequences and each shared position within each sequence.

One well known obstacle when training recurrent NNs is the problem of the vanishing gradients [3]. In most cases, the gradient decays exponentially with

the number of layers in a NN. Since the unfolded network has at least as many layers as the length of the input sequences, there is a limit for storing features that might be relevant for the prediction of a property and are separated by longer sequences from that property.

A learning algorithm that performs well without depending on the magnitude of weight gradients is the resilient backpropagation (RPROP) algorithm [15]. Only the sign of the derivative is considered to indicate the direction of the weight update. The size of the weight change is determined by a weight-specific update-value $\triangle_{ij}^{(epoch)}$ and defined as

$$\triangle w_{ij}^{(epoch)} = \begin{cases} -\triangle_{ij}^{(epoch)} & , \text{ if } \frac{\partial E}{\partial w_{ij}}^{(epoch)} > 0 \\ +\triangle_{ij}^{(epoch)} & , \text{ if } \frac{\partial E}{\partial w_{ij}}^{(epoch)} < 0 \\ 0 & , \text{ else} \end{cases} \tag{4}$$

where $\frac{\partial E}{\partial w_{ij}}^{(epoch)}$ denotes the summed gradient information over all patterns of the pattern set (one *epoch* in batch learning). After each batch update, the new update-values $\triangle_{ij}(epoch)$ are determined using a sign-dependent adaptation process.

$$\triangle_{ij}^{(epoch)} = \begin{cases} \eta^+ * \triangle_{ij}^{(epoch-1)} & , \text{ if } \frac{\partial E}{\partial w_{ij}}^{(epoch-1)} * \frac{\partial E}{\partial w_{ij}}^{(epoch)} > 0 \\ \eta^- * \triangle_{ij}^{(epoch-1)} & , \text{ if } \frac{\partial E}{\partial w_{ij}}^{(epoch-1)} * \frac{\partial E}{\partial w_{ij}}^{(epoch)} < 0 \\ \triangle_{ij}^{(epoch-1)} & , \text{ else} \end{cases} \tag{5}$$

where $\eta^- = 0.5$ and $\eta^+ = 1.05$ are fixed values.

Every time the partial derivative of the corresponding weight w_{ij} changes its sign, which indicates that the last update was too big and the algorithm has jumped over a local minimum, the update-value $\triangle_{ij}^{(epoch)}$ is decreased by the factor η^-. If the derivative retains its sign, the update-value is slightly increased in order to accelerate convergence in shallow regions.

Using this scheme the magnitude of the update-values is completely decoupled from the magnitude of the gradients. It is possible that a weight with a very small gradient has a very large weight change and vice versa, if it just is in accordance with the topology of the error landscape. This learning algorithm converges substantially faster than standard backpropagation and allows all BRNN simulations to be run on standard PC hardware.

3 Results

A sequence of the first 45 residues of the N-terminal is used for the input vectors $U_t \in [0,1]^{20}$ where each vector codes the residue in the standard one out of 20 coding. Each element of output sequence $Y_t \in [0,1]$ is set to 1, if the residue at position t is part of the signal peptide before the cleavage site and 0 otherwise.

The set of the signal peptide sequences and the two sets of non-secretory sequences are each split into 5 subsets of equal size. Fivefold crossvalidation is used by training 5 networks on 3/5 of each set, using 1/5 of each set as a validation set to stop the training and 1/5 of each set as a test set to measure the performance. The sizes of the different layers in the BRNN are determined experimentally. The number of units in the forward and backward state vectors is 5 or 6, the number of hidden units feeding into the state vectors is between 10 and 13 and the number of hidden units feeding into the output layer is between 2 and 6.

The criterion to discriminate the signal peptides is optimized using the validation set. If a segment of more than 7 consecutive output activations meets the condition $Y_t > 0.75, t \in [1, 45]$, the input sequence is classified as a signal peptide.

The performance is measured by calculating the correlation coefficient, [10] the sensitivity *sens* and specificity *spec* defined as

$$sens = cp/(cp + fn) \tag{6}$$

$$spec = cp/(cp + fp), \tag{7}$$

where cp is the number of correct positive examples, fn is the number of false negative and fp is the number of false positive examples. The results are given in table 1. The average percentage of correctly predicted sequences is 99.55%

Table 1. Performance on the test sets in fivefold crossvalidation.

dataset	sensitivity	specificity	correlation
1	100.0	100.0	1.0
2	100.0	98.8	0.984
3	100.0	98.8	0.984
4	100.0	98.8	0.984
5	100.0	100.0	1.0
avg	100.0	99.3	0.99

Another test set is created by extracting human protein sequences from the release 39.25 of SWISS-PROT that have the feature keyword SIGNAL and that are not contained in the used training set. The non signal peptide proteins used in this set are full-length mRNAs with the N-terminus experimentally verified. Since the data set for training the SignalP2 system is not available, it is not guaranteed that all of these sequences are not novel to that system. For the performance measurements shown in table 2 and the general use of the Sigfind system, the predictions of the 5 networks are combined using a majority vote of the jury of these networks.

For scanning targets in complete genomes, the execution time of an algorithm should not be neglected. As shown in table 3 the Sigfind system compares very favorably with SignalP2.

Table 2. Performances on an independent testset.

system	sensitivity	specificity	correlation
signalp2 with NN and HMM	88.28	96.55	0.851
signalp2 with NN	93.24	92.83	0.856
signalp2 with HMM	90.99	94.83	0.857
signalp2 with NN or HMM	95.94	91.42	0.867
sigfind	98.64	93.59	0.918

Table 3. Execution times on a Sun Sparc.

system	222 signal peptides	210 non signal peptides
signalp2	300 seconds	251 seconds
sigfind	1.98 seconds	1.94 seconds

4 Availability

The sigfind system can be used for predictions using a WWW interface with the URL http://www.stepc.gr/ synaptic/sigfind.html .

5 Discussion

The approach introduced here is shown to be very accurate for the identification of signal peptides in human protein sequences. The identification of signal peptides has become of major importance in the analysis of sequenced data. Programs like SignalP2 and Psort [7] are widely used for this purpose and produce in combination quite reliable predictions. The algorithm that we developed here can be used as an computationally efficient, additional tool to find signal peptides.

The method is designed to determine if the N-terminal part of a protein is a signal peptide or not. However in many cases the complete N-terminal part of the sequence is not experimentally verified but predicted in silico. Particularly in protein sequences derived from genomic data it is possible that the real startcodon is not predicted correctly. The same occurs also for protein sequences derived out of expressed sequence tags (ESTs). The use of Sigfind on the whole protein might produce false positive predictions mostly on the location of transmembrane regions which has a similar structure as the signal peptides. For these reasons we are planning to combine the Sigfind predictions with reliable startcodon [4] and coding region predictions. For ESTs it would also be useful to apply a frame shift correction software [6,5]. The modified BRNN algorithm described has very great potential for these tasks as well as for other biosequence analysis problems.

References

1. Bairoch, A., Boeckmann, B.: The swiss-prot protein sequence data bank: current status. Nucleic Acids Res. 22 (1994) 3578–3580
2. Baldi, P., Brunak, S., Frasconi, P., Pollastri, G., Soda, G.: Bidirectional dynamics for protein secondary structure prediction. In: Sun, R., Giles, L. (eds.): Sequence Learning: Paradigms, Algorithms, and Applications. Springer Verlag (2000)
3. Bengio, Y., P.Simard, Frasconi, P.: Learning long-term dependencies with gradient descent is difficult. IEEE Trans. on Neural Networks, 5 (1994) 157–166
4. Hatzigeorgiou, A.: Translation initiation site prediction in human cDNAs with high accuracy. Bioinformatics, 18 (2002) 343–350
5. Hatzigeorgiou, A., Fizief, P., Reczko, M. Diana-est: A statistical analysis. Bioinformatics, 17 (2001) 913–919
6. Hatzigeorgiou, A., Papanikolaou, H., Reczko, M.: Finding the reading frame in protein coding regions on dna sequences: a combination of statistical and neural network methods. In: Computational Intelligence: Neural Networks & Advanced Control Strategies. IOS Press, Vienna (1999) 148–153
7. Horton, P., Nakai, K.: Better prediction of protein cellular localization sites with the k nearest neighbors classifier. In: ISMB (1997) 147–152
8. Kyte, J., Doolittle, R.: A simple method dor displaying the hydrophatic character of a protein. J. Mol. Biol., 157 (1982) 105–132
9. Ladunga, I.: Large-scale predictions of secretory proteins from mammalian genomic and est sequences. Curr. Opin. in Biotechnolgy, 11 (2000) 13–18
10. Mathews, B. W.: Comparison of the predicted and observed secondary structure of T4 phage lysozyme. Biochem. Biophys. Acta, Vol. 405 (1975) 442–451
11. Minsky, M., Papert, S.: Perceptrons: An Introduction to Computational Geometry. The MIT Press, Cambridge, Massachusetts (1969) 145
12. Nielsen, H., Brunak, S., von Heijne, G. Machine learning approaches for the prediction of signal peptides and other protein sorting signals. Protein Engineering, 12 (1999) 3–9
13. Nielsen, H., Engelbrecht, J., S.Brunak, von Heijne, G.: Identification of prokaryotic and eukaryotic signal peptides and prediction of their cleavage sites. Protein Engineering, 10 (1997) 1–6
14. Nielsen, H., Krogh, A.: Prediction of signal peptides and signal anchors by a hidden markov model. In: ISMB (1998) 122–130
15. Riedmiller, M., Braun, H.: A direct adaptive method for faster backpropagation learning: The RPROP algorithm. In: Ruspini, H., (ed.): Proceedings of the IEEE International Conference on Neural Networks (ICNN 93). IEEE, San Francisco (1993) 586–591
16. Rumelhart, D. E., Hinton, G. E., Williams, R. J.: Learning internal representations by error propagation. In: Rumelhart, D. E., McClelland, J. L. (eds.): Parallel Distributed Processing: Explorations in the microstructure of cognition; Vol. 1: Foundations. The MIT Press, Cambridge, Massachusetts (1986)
17. v. Heijne, G.: A new method for predicting signal sequence cleavage sites. Nucleid Acids Res., 14 (1986) 4683–4690
18. von Heijne, G.: Computer-assisted identification of protein sorting signals and prediction of membrane protein topology and structure. In: Advances in Computational Biology, volume 2, Jai Press Inc. (1996) 1–14