Proteomics 2004, *4,* 1591–1596 DOI 10.1002/pmic.200300769

Prediction of the subcellular localization of eukaryotic proteins using sequence signals and composition

Martin Reczko[1, 3] and Artemis Hatzigeorgiou[2]

[1]Bioinformatics Lab, Institute of Computer Science, Foundation for Research and Technology – Hellas (FORTH), Heraklion, Crete, Greece
[2]Department of Genetics, University of Pennsylvania, School of Medicine, Philadelphia, PA, USA
[3]Synaptic Ltd., Science and Technology Park of Crete, Voutes Heraklion, Greece

A tool called Locfind for the sequence-based prediction of the localization of eukaryotic proteins is introduced. It is based on bidirectional recurrent neural networks trained to read sequentially the amino acid sequence and produce localization information along the sequence. Systematic variation of the network architecture in combination with an efficient learning algorithm lead to a 91% correct localization prediction for novel proteins in fivefold cross-validation. The data and evaluation procedure are the same as the non-plant part of the widely used TargetP tool by Emanuelsson *et al.* The Locfind system is available on the WWW for predictions (http://www.stepc.gr/~synaptic/locfind.html).

Received	19/10/03
Revised	23/1/04
Accepted	2/2/04

Keywords: Eukaryotic proteins / Neural networks / Subcellular localization

1 Introduction

A highly optimized machinary post-translationally sorts and transports newly synthesized proteins encoded in the nuclear genome from the cytosol to different subcellular compartments [1]. Knowledge about the subcellular location of a protein restricts the possible function of a protein [2] and is a valuable annotation to filter large amounts of protein sequences for which a precise functional annotation is not available. Other uses of such predictions are testing the localization of designed proteins, the large-scale screening of proteomes for proteins with desired localization as targets for drug design, or the construction of DNA microarrays. Localization prediction is also useful for the interpretation of multiple spots originating from the same protein possibly due to post-translational modifications in two-dimensional electrophoresis experiments conducted in proteomics [3]. Generally, the development of these kinds of computational tools is a typical example where biologists and bioinformaticians mutually benefit from progress in their fields, where the computational models get more and more realistic with the advent of proteome analyses with detailed information about protein targets and the targeting mechanisms and biologists can refine these models and use them for large-scale predictions.

The most commonly used methods for pedicting subcellular localization are reviewed in [3–5]. The usual distinction between these methods defines three classes: detection of targeting signals [6–10], detection of different amino acid compositions [11–13], and the use of evolutionary relationships between proteins targeted to the same compartment [14, 15]. Some approaches combine the use of signals and sequence composition explicitly [16, 17] or implicitly [18]. This implicit combination of sequence composition and signals using recurrent neural networks is extended in this work to a multiclass localization prediction.

Correspondence: Martin Reczko, Bioinformatics Lab, Institute of Computer Science, Foundation for Research and Technology – Hellas (FORTH) P. O. Box 1385, 71110 Heraklion, Crete, Greece
E-mail: reczko@ics.forth.gr
Fax: +30-2810-391602

Abbreviations: BRNN, bidirectional recurrent neural network; **MCC**, Mathews correlation coefficient; **MLP**, multilayer perceptron; **mTP**, mitochondrial targeting peptide; **NN**, neural network; **SP**, signal peptide; **ROC**, receiver operating characteristics

2 Materials and methods

2.1 Data

The data used for training, validation, and testing of the networks was obtained from the TargetP web-site (http://www.cbs.dtu.dk/services/TargetP). In this study,

we use only the non-plant data set consisting of 371 mitochondrial targeting peptides (mTP), 715 signal peptides (SPs), and 1214 nuclear and 438 cytosolic sequences combined into the 'other' class. Each set of sequences $ALLSEQS_{class}$ for the three classes (class in [mTP, SP, other]) is split into five equally sized disjunct subsets $CLASSSUBSET_{classes,set}$ (set = 1 . . . 5). The five subsets of each class contribute to one of the five partitions $PARTITION_{set}$. A partition $PARTITION_{set}$ contains all the subsets $CLASSSUBSET_{class,set}$ for the three classes (class in [mTP, SP, other]). To conduct a fivefold cross-validation, five different networks are trained using a training set that is the combination of three different partitions TRA_{xval} = $PARTITION_{set_a}$ + $PARTITION_{set_b}$ + $PARTITION_{set_c}$. Of the remaining two partitions, one is used as a validation set VAL_{xval} = $PARTITION_{set_c}$ for early stopping of the learning algorithm, optimization of the network architecture, and postprocessing of the network outputs while the other is used only for the single final test set TES_{xval} = $PARTITION_{set_d}$ for which predictive performance is measured. The choices for the different permutations are restricted such that the five tests are different and the validation sets are different.

2.2 Neural network architecture

A known limitation in the application of standard feedforward neural networks (NNs) for sequence analysis is the tradeoff between the need for processing large sequence fragments and the generalization capability decreasing with the size of the network. One solution is the use of recurrent NNs that sequentially process small sequence fragments or single elements and are able to learn to store features over long distances in the sequence using internal activations in feedback loops. The NN architecture employed here is the bidirectional recurrent neural network (BRNN) introduced in [19]. A modified version of this architecture has shown good performance learning the task of sequence-based prediction of signal peptides [18]. One advantage of recurrent networks and especially BRNNs is the ability to make use of unaligned localized sequence signals and the overall sequence content in longer ranges. The BRNN architecture is summarized in the following and the modifications to learning algorithm are introduced.

To store both the context to the left and right for any position in a sequence, the BRNN has two feedback loops with two state vectors defined as $F_t \in IR^n$ and $B_t \in IR^m$ that contain the context information while reading forward and backwards, respectively. They are calculated as:

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

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

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

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

$$Y_t = \eta(F_t, B_t) \quad (3)$$

where $\eta()$ is again realized by a single layer transfer function $\aleph_\eta$. In the realization of [19], $\aleph_\eta$ contains one hidden layer that is connected to U_t. To find the optimal network architectures in this application, all sizes of the different layers are varied systematically and for each setting the network is trained until the highest Mathews correlation coefficient (MCC) is measured on the validation set (VAL_{xval}). It turns out that in all cross-validated data sets the optimal network architecture has none of those hidden units of the original BRNN architecture connecting the input layer without any feedback input to the output layer. As any information in the input layer affecting directly the current output can be passed through the forward or backward states, the processing capabilites will not be reduced without these units. The architecture is shown in Fig. 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 $X_{t+1} = q^{+1}X_t$.

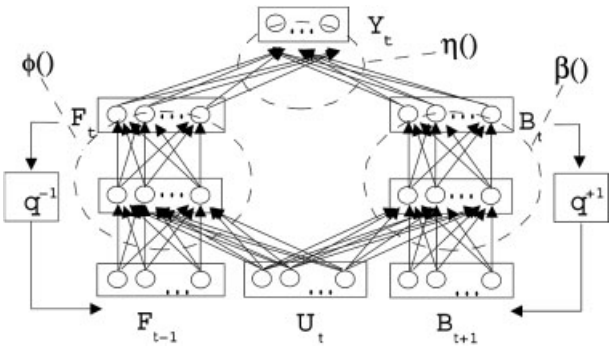

Figure 1. Modified BRNN architecture. The input at time t is applied at U_t. Y_t contains the output vector. F_t is the forward state vector, and B_t the backward state vector. The shift operator q^{-1} copies the last F_t state vector while reading the input sequence forwards while the operator q^{+1}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

2.3 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 [20] is used to calculate the gradient

for a relative entropy error measure [21] for all weights. The unfolding is illustrated in Fig. 2. For each position in the sequence, the activations in a copy of the complete network are calculated using also the activations of the copies to the left and right of that position. As the networks at each position are just copies of the original recurrent net, a single weight in the recurrent network occurs identical in the copies at all positions. These occurencies of the weight are called the "shared positions" of the weight. The transformation of a recurrent NN into an equivalent feedforward NN is called backpropagation through time and was first described in [22] and the application of backpropagation learning to these networks was introduced in [20]. 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 [23]. 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.

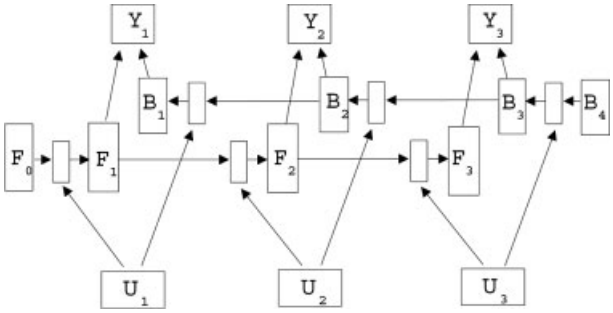

Figure 2. Unfolded feedforward structure of a BRNN processing an input sequence U_t with length $T = 3$. First, the forward states $F_0, \ldots, F_3$ are calculated sequentially from left to right, while the backward states $B_4, \ldots, B_1$ are calculated beginning with the end of the sequence and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

A learning algorithm that performs well without depending on the magnitude of weight gradients is the resilient backpropagation (RPROP) algorithm [24]. 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 $\Delta_{ij}^{(epoch)}$ and defined as

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

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 Δ_{ij}(epoch) are determined using a sign-dependent adaptation process.

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

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 $\Delta_{ij}^{(epoch)}$ 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.

2.4 Performance measures

The classification performance is measured by calculating the percentage of correct classifications, the Mathews correlation coefficient [25, 21], the sensitivity and the specificity defined as *sensitivity* = *cp*/(*cp* + *fn*) and *specificity* = *cp*/(*cp* + *fp*) 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.

2.5 Data representation

All sequences were truncated after the first 90 *N*-terminal residues. A protein is processed by the BRNN by sequentially coding the residues in the sequence into the input layer one by one using the standard one out of 20 code. The three different localization classes are represented using three output neurons having normalized exponentials [21] as activation functions (the so-called "soft-max" output). The target values for the output units along the sequence have an activation of 1.0 for one of the classes mTP or SP, if the residue in the sequence is part of the peptide indicating the corresponding localization and ranging from the *N*-terminal to the cleavage site of that peptide. To assign a localization class to the sequence of output activations, the average activation corresponding to the classes mTP and SP is calculated in a region of

Table 1. Parameters of the five networks for optimal performance on the corresponding validation sets

Net	No. of forward state units (*n*)	No. of backward state units (*m*)	No. of forward hidden units	No. of backward hidden units	avgregion	thresh
1	4	4	5	3	21	0.25
2	4	4	4	3	25	0.25
3	5	4	3	3	25	0.25
4	4	4	5	3	19	0.25
5	5	4	3	3	26	0.25

fixed size from to *N*-terminal end of the protein. The parameter "avgregion" determines the number of residues in this region. If the class with the largest average activation in this region exceeds a threshold "thresh", the sequence is predicted into that class and into the "other" class else. The values for "avgregion" and "thresh" are varied systematically to give the optimal Mathews correlation on the validation set. These optimal values and the parameters for the architecture for the five networks are shown in Table 1. Taking into account that the targeting sequences have very different lengths for the different target classes, separate values for the "avgregion" for each class are reasonable. As this adds more dimensions to the systematic variations, we restrict ourselves to a uniform size for all classes in this study due to computational limitations.

3 Results and discussion

3.1 Performance on novel test sets

The classification results on the test sets $TES_1 \ldots TES_5$ for all five independent networks are summarized in Table 2 with the scores of TargetP [9] in parentheses. The overall percentage of correct classifications of Locfind (91.3 ($\pm$ 0.99)%) is slightly better than the corresponding percentage of TargetP (90.0 ($\pm$ 1.1)%). The performance of the Locfind predictions for the mTP class is clearly higher than the predictions in TargetP as the correlation coefficient indicates (0.77 *versus* 0.73).

3.2 Using the network activation as a reliability index

The average activation on the avgregion *N*-terminal residues gives also an indication for the reliability of the classification. By varying the acceptance threshold, the classifications can be discerned between higher sensitivity or higher specificity. The relationship between sensitivity and specificity by varying this threshold is shown in the so-called "receiver-operating-characteristic" (ROC) [21] curves for the different classes in Figs. 3, 4, and 5. The Web version of Locfind reports the average activation for each class and each of the five networks in order to assess the prediction reliability. Furthermore, the decisions using the thresholds for optimal Mathews correlation of the five networks are combined into one prediction using a majority vote and the higher sum of average activations in case of a voting tie. This usually leads to another small increase in predictive performance.

Table 2. Locfind prediction performance on novel test-sets in fivefold cross-validation (scores of TargetP [9] in parentheses)

Data set	True category	No. of sequences	Predicted category			Sensitivity	MCC
			mTP	SP	Other		
Non-plant	mTP	371	290 (330)	23 (9)	58 (32)	0.78 (0.89)	0.77 (0.73)
	SP	715	2 (13)	668 (683)	45 (19)	0.93 (0.96)	0.89 (0.92)
	Other	1652	63 (152)	47 (49)	1542 (1451)	0.93 (0.88)	0.84 (0.82)
	Specificity		0.82 (0.67)	0.91 (0.92)	0.94 (0.97)		

In total 91.3($\pm$0.99)% (90.0($\pm$1.1)%) correct predictions for the non-plant predictor, where the standard deviations refer to the spread in performance of the five different networks and test sets used for cross-validation.

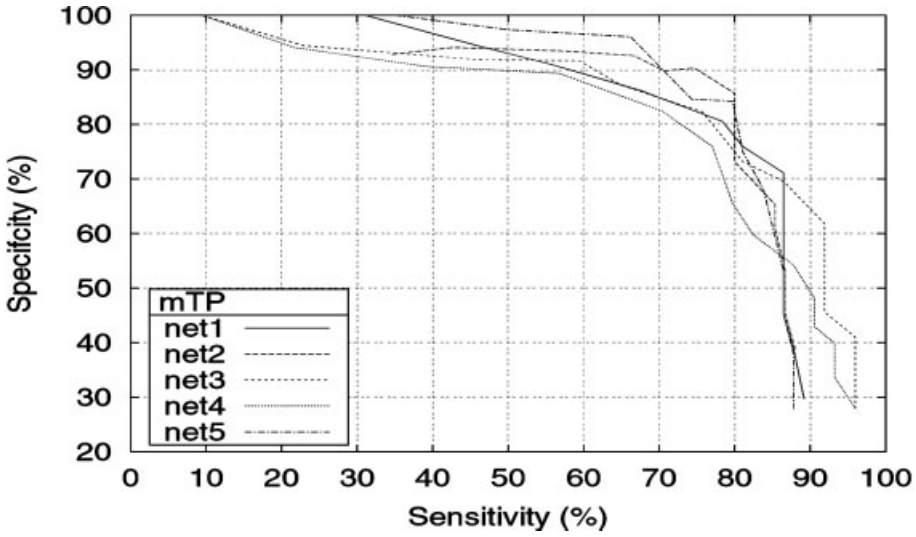

Figure 3. ROC analysis for mTP.

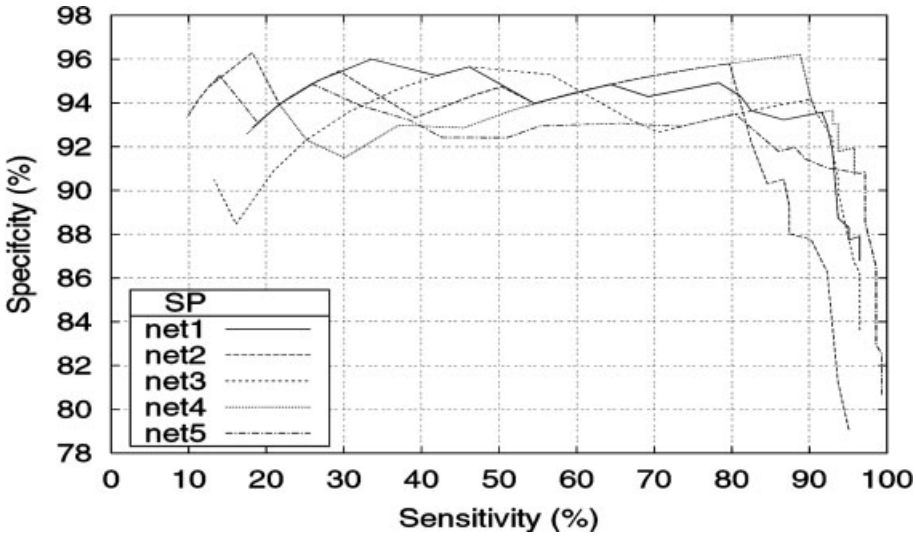

Figure 4. ROC analysis for SP.

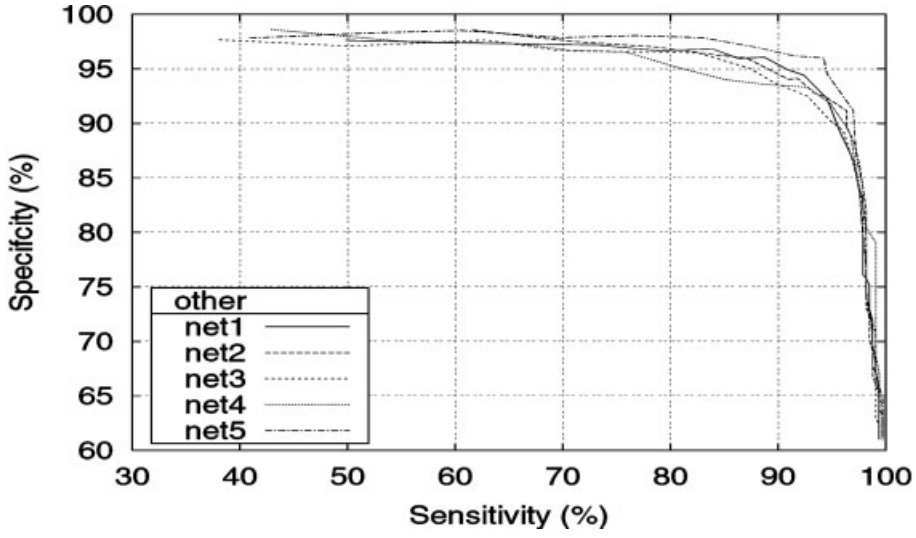

Figure 5. ROC analysis for the class "other".

4 Concluding remarks

Locfind is a novel tool for the sequence-based prediction of the localization of eukaryotic proteins. The number of parameters adjusted by the learning algorithm is significantly lower than in the usual feedforward neural network approaches to this problem and for that reason, better generalization can be expected. Apart from its high prediction accuracy it hence shares the advantage of fast execution that was already shown for the related tool Sigfind [18]. We are currently extending this tool to include also the plant data sets used in TargetP. While trying to improve the prediction accuracy we should of course be aware of the limitations due to proteins that are targeted simultaneously to two or more cellular compartments [26], an interesting phenomenon that should be reflected in the data sets used.

5 References

[1] Schatz, G., Dobberstein, B., *Science* 1996, *271*, 1519–1526.

[2] Eisenhaber, B., Bork, P., *Trends Cell Biol.* 1998, *9*, 169–170.

[3] Nakai, K., *J. Struct. Biol.* 2001, *134*, 103–116.

[4] Emanuelsson, O., von Heijne, G., *Biochim. Biophys. Acta* 2001, *1541*, 114–119.

[5] Nakai, K., *Adv. Prot. Chem.* 2000, *54*, 277–344.

[6] Nielsen, H., Engelbrecht, J., Brunak, S., von Heijne, G., *Prot. Eng.* 1997, *10*, 1–6.

[7] Nielsen, H., Brunak, S., von Heijne, G., *Prot. Eng.* 1999, *12*, 3–9.

[8] Claros, M. G., Vincens, P., *Eur. J. Biochem.* 1996, *241*, 779–786.

[9] Emanuelsson, O., Nielsen, H., Brunak, S., von Heijne, G., *J. Mol. Biol.* 2000, *300*, 1005–1016.

[10] Jagla, B., Schuchhardt, J., *Bioinformatics* 2000, *16*, 245–250.

[11] Reinhardt, A., Hubbard, T., *Nucleic Acids Res.* 1998, *26*, 2230–2236.

[12] Chou, K. C., *Prot. Eng.* 2001, *17*, 75–79.

[13] Hua, S., Sun, Z., *Bioinformatics* 2001, *14*, 721–728.

[14] Marcotte, E. M., Xenarios, I., van der Bliek, A. M., Eisenberg, D., *PNAS* 2000, *97*, 12115–12120.

[15] Mott, R., Schultz, J., Bork, P., Ponting, C. P., *Genome Res.* 2002, *12*, 1168–1174.

[16] Bannai, H., Tamada, Y., Maruyama, O., Nakai, K., Miyano, S., *Bioinformatics* 2002, *18*, 298–305.

[17] Drawid, A., Gerstein, M., *J. Mol. Biol.* 2000, *301*, 1059–1075.

[18] Reczko, M., Staub, E., Fiziev, P., Hatzigeorgiou, A., in: Guigo, R., Gusfield, D. (Eds.), *Algorithms in Bioinformatics, Proceedings of the 2nd Int. Workshop WABI 2002*, Rome, Italy, September 16–21, *Lecture Notes in Computer Science*, Vol. 2452, Springer, New York 2002, pp. 60–67.

[19] Baldi, P., Brunak, S., Frasconi, P., Pollastri, G., Soda, G., in: Sun, R., Giles, L. (Eds.), *Sequence Learning: Paradigms, Algorithms, and Applications*, Springer Verlag, New York 2000.

[20] Rumelhart, D. E., Hinton, G. E., Williams, R. J., in: Rumelhart, D. E., McClelland, J. L. (Eds.), *Parallel Distributed Processing: Explorations in the Microstructure of Cognition; Vol. 1: Foundations*, The MIT Press, Cambridge, MA, 1986.

[21] Baldi, P., Brunak, S., Chauvin, Y., Andersen, C. A. F., Nielsen, H., *Bioinformatics* 2000, *16*, 412–424.

[22] Minsky, M., Papert, S., *Perceptrons: An Introduction to Computational Geometry*, The MIT Press, Cambridge, MA, 1969, p. 145.

[23] Bengio, Y., Simard, P., Frasconi, P., *IEEE Trans. Neural Networks* 1994, *5*, 157–166.

[24] Riedmiller, M., Braun, H., in: Ruspini, H. (Ed.), *Proceedings of the IEEE International Conference on Neural Networks (ICNN 93)*, IEEE, San Francisco, CA, 1993, pp. 586–591.

[25] Mathews, B. W., *Biochem. Biophys. Acta* 1975, *405*, 442–451.

[26] Small, I., Wintz, H., Akashi, K., Mireau, H., *Plant Mol. Biol.* 1998, 38, 265–277.

