Exploiting the past and the future in protein secondary structure prediction

Pierre Baldi*
Dept. of Information and Computer Science
University of California, Irvine
Irvine, CA 92697-3425

Søren Brunak
Center for Biological Sequence Analysis
The Technical University of Denmark
DK-2800 Lyngby, Denmark

Paolo Frasconi, Giovanni Soda
Dept. of Informatics and Systems
University of Florence
50139 Florence, Italy

Gianluca Pollastri
Dept. of Information and Computer Science
University of California, Irvine
Irvine, CA 92697-3425

Abstract

Motivation: Predicting the secondary structure of a protein (alpha-helix, beta-sheet, coil) is an important step towards elucidating its three dimensional structure, as well as its function. Presently, the best predictors are based on machine learning approaches, in particular neural network architectures with a fixed, and relatively short, input window of amino acids, centered at the prediction site. Although a fixed small window avoids overfitting problems, it does not permit to capture variable long-ranged information.
Results: We introduce a family of novel architectures which can learn to make predictions based on variable ranges of dependencies. These architectures extend recurrent neural networks, introducing non-causal bidirectional dynamics to capture both upstream and downstream information. The prediction algorithm is completed by the use of mixtures of estimators that leverage evolutionary information, expressed in terms of multiple alignments, both at the input and output levels. While our system currently achieves an overall performance close to 76% correct prediction—at least comparable to the best existing systems—the main emphasis here is on the development of new algorithmic ideas.
Availability: The executable program for predicting protein secondary structure is available from the authors free of charge.
Contact: *pfbaldi@ics.uci.edu*, *gpollast@ics.uci.edu*, *brunak@cbs.dtu.dk*, *paolo@dsi.unifi.it*.

1 Introduction

Computational predictive tools for the structure and function of proteins have become increasingly important as a result of genome and other sequencing projects. One significant step towards elucidating the structure and function of a protein, is the prediction of its secondary structure (SS). The SS consists of local folding regularities maintained by hydrogen bonds and traditionally subdivided into three classes: alpha-helices, beta-sheets, and coils representing all the rest. In alpha-helices, backbone hydrogen bonds link residues i and $i+4$, whereas in beta-sheets, hydrogen bonds link two sequence segments, in either parallel or antiparallel fashion. The SS structure can be sensitive to single amino acid changes and depends on both local and long-ranged interactions.

The sequence preferences and correlations involved in these structures have made SS prediction one of the classical problems in computational molecular biology, and one where machine learning approaches have been particularly successful [see (Baldi & Brunak, 1998) for a detailed review]. In particular, many different feedforward neural network (NN) architectures have been applied to this task (Qian & Sejnowski, 1988; Rost & Sander, 1994; Riis & Krogh, 1996). The influential early work of Qian and Sejnowski was based on a fully connected NN, with a local input window of typical length 13 amino acids with orthogonal encoding, and a single hidden layer. The output layer consisted of three sigmoidal units with orthogonal encoding of the SS classes for the residue located at the center of the input window. A significant improvement was obtained by cascading the previous architecture with a second network to clean up the output of the lower network. The cascaded architecture reached a performance of $Q_3 = 64.3\%$, with the correlations $C_\alpha = 0.41$ for helices, $C_\beta = 0.31$ for sheets, and $C_\gamma = 0.41$ for coils. Throughout this article, we use the standard performance measures reviewed in (Baldi *et al.*, 2000a). Unless we specify otherwise, Q_3 percentages are measured on a per residue basis. Prediction of SS based on single sequences and local windows seem to be limited to < 65–69% accuracy. Increasing the size of the window, however, does not lead to improvements because of the overfitting problem associated with large networks.

Building upon the work of (Qian & Sejnowski, 1988),

*and Department of Biological Chemistry, College of Medicine, University of California, Irvine. To whom all correspondence should be addressed.

for a long time the best SS prediction performance has been achieved by the PHD scheme (Rost & Sander, 1993; Rost & Sander, 1994). Rost and Sander used a number of machine learning techniques including early stopping, ensemble averages of different networks, and a weighting scheme to compensate for the well known composition biases of large low-similarity databases (roughly 30% helices, 20% sheets, and 50% coils). Most of the improvements, however, seem to result from the use of input profiles, derived from multiple alignments, that can leverage evolutionary information—the SS being considerably more conserved than the primary structure. In the 1996 Asilomar blind prediction competition CASP2 (Critical Assessment of Protein Structure Prediction), this method outperformed all others (Eisenberg, 1997), reaching a performance level of 74%. While input profiles contain information not present in individual sequences, it is worth noting that they also discard information by losing intra-sequence correlations.

In (Riis & Krogh, 1996), further NN architectural and machine learning refinements are used, such as an adaptive encoding of the input amino acids by the NN weight sharing technique to reduce the number of free parameters. Different networks are designed for each SS class by leveraging biological knowledge, such as the periodicity of alpha-helices, with output filtering and ensemble averaging. Finally, predictions made from individual sequences are combined at the *output* level, using both multiple alignments and a maximum entropy weighting scheme (Krogh & Mitchinson, 1995). In spite of a considerable amount of architectural design, the final performance with multiple alignments is practically identical to (Rost & Sander, 1994) with an overall accuracy of $Q_3 = 71.3\%$, and correlations $C_\alpha = 0.59$, $C_\beta = 0.50$, and $C_\gamma = 0.41$.

More recently, Cuff and Barton (Cuff & Barton, 1999) have compared and combined the main existing predictors. On the particular data sets used in their study, the best isolated predictor is still PHD with $Q_3 = 71.9\%$. At the last 1998 CASP3 competition, the best results were obtained by (Jones, 1999), using a relatively simple but large NN architecture. Out of the 35 blind sequences, the program selected 23 and achieved a performance of $Q_3 = 77.6\%$ per protein, or $Q_3 = = 75.5\%$ per residue. The improvements seem to result in part from the use of PSI-BLAST generated profiles, although this is somewhat controversial (Cuff & Barton, 1999) and not directly reproducible since the filters used to process the raw profiles are not described in sufficient detail in (Jones, 1999).

Thus it appears today that to further improve SS prediction one should use distant information, in sequences and alignments, that is not contained in *local* input windows. This is particularly clear in the case of beta-sheets where stabilizing bonds can be formed between amino acids far apart. This, however, poses two related challenges: (1) avoiding the overfitting associated with large-input-windows; (2) detecting *sparse* and weak long-ranged signals to modulate the significant local information, while ignoring the additional noise found over larger distances. In this paper, we approach the prediction problem in a new way, introducing an algorithm that uses the whole protein sequence rather than a short substring.

To begin with, protein SS prediction can be formulated as the problem of learning a synchronous sequential translation from strings in the amino acid alphabet to strings in the SS alphabet. This task is a special form of grammatical inference. Although several symbolic algorithms exist for learning grammars (Angluin & Smith, 1987), to the best of our knowledge they have not led to successful protein SS predictors presumably because of their scarce robustness in the presence of noisy data. Connectionist approaches, on the other hand, are based on statistical learning and therefore tend to exhibit greater robustness. The main connectionist architectures that have been investigated for grammatical inference are recurrent neural networks (RNN), with both first- (Cleeremans, 1993) and second-order (Giles *et al.*, 1992) connections, as well as and input-output hidden Markov models (IOHMM) (Baldi & Chauvin, 1996; Bengio & Frasconi, 1996). Both RNNs and IOHMMs are sensible alternatives to methods based on a fixed-width input window. The expressive power of these models enables them to capture distant information in the form of contextual knowledge stored into hidden state variables. In this way, they can overcome the main disadvantage of feedforward networks, namely the linear growth of the number of parameters with the window size. Intuitively, these models are parsimonious because of the implicit weight sharing resulting from their stationarity, i.e. parameters do not vary over time. Thus, it would make sense to tackle the SS prediction problem using RNNs or IOHMMs.

A more careful analysis, however, reveals a basic limitation of standard RNNs and IOHMMs in computational biology. In fact, both classes of models are *causal* in the sense that the output at time t does not depend on future inputs. Causality is easy to justify in dynamics that attempt to model the behavior of physical systems, or that need to operate in real time. Clearly, in these cases the response at time t cannot depend on stimulae that the system has not yet encountered. But biological sequences are not really temporal: the conformation and function of a region in a sequence may strongly depend on events located both upstream and downstream. Thus, to tackle the SS prediction problem, we develop a connectionist architecture that provides a non-causal generalization of RNNs. Our proposal is motivated by the assumption that both adaptive dynamics and non-causal processing are needed to overcome the drawbacks of local fixed-window approaches. Furthermore, we leverage evolutionary information, both at the input and output levels, using a mixture-of-estimators approach. While our current system achieves an overall performance exceeding 75% correct prediction—at least comparable to the best existing systems—the main emphasis here is on the development of new algorithmic ideas.

2 Methods and Algorithms

2.1 Data Preparation

The assignment of the SS categories to the experimentally determined 3D structure is nontrivial and is usually performed by the widely used DSSP program (Kabsch & Sander, 1983). DSSP works by assigning potential backbone hydrogen bonds (based on the 3D coordinates of the backbone atoms) and subsequently by identifying repetitive bonding patterns. Two alternatives to this assignment scheme are the programs STRIDE and DEFINE. In addition to hydrogen bonds, STRIDE uses also dihedral angles (Frishman & Argos, 1995). DEFINE uses difference distance matrices for evaluating the match of interatomic distances in the protein to those from idealized SS (Richards & Kundrot, 1988). While assignment methods impact prediction performance to some extent (Cuff & Barton, 1999), here we concentrate exclusively on the DSSP assignments.

A number of data sets were used to develop and test our algorithms. We will refer to each set using the number of sequences contained in it. The first high quality data used in this study was extracted from the Brookhaven Protein Data Bank (PDB) (Bernstein & et al., 1977) release 77 and subsequently updated. We excluded entries if:

- They were not determined by X-ray diffraction, since no commonly used measure of quality is available for NMR or theoretical model structures.
- The program DSSP could not produce an output, since we wanted to use the DSSP assignment of protein secondary structure (Kabsch & Sander, 1983).
- The protein had physical chain breaks (defined as neighboring amino acids in the sequence having C^{α}-distances exceeding 4.0Å).
- They had a resolution worse than 1.9Å, since resolutions better than this enables the crystallographer to remove most errors from their models.
- Chains with a length of less than 30 amino acids were also discarded.

From the remaining chains, a representative subset with low pairwise sequence similarities was selected by running the algorithm #1 of Hobohm *et al.* (1992), using the local alignment procedure search (rigorous Smith-Waterman algorithm) (Myers & Miller, 1988; Pearson, 1990) using the pam120 matrix, with gap penalties -12, -4. Thus we obtained a data set consisting of 464 distinct protein chains, corresponding to 123,752 amino acids, roughly 10 times more than what was available in (Qian & Sejnowski, 1988).

Another set we used is the EMBL non-redundant PDB subsets that can be accessed by ftp at the site `ftp.embl-heidelberg.de`. Data details are in the file `/pub/databases/pdb_select/README`. The extraction is based on the file: `/pub/databases/pdb_select/1998_june.25.gz` containing a set of non-redundant (25%) PDB chains. After removing 74 chains on which the DSSP program crashes, we obtained another set of 824 sequences, overlapping in part with the former ones.

In addition, we also used the original set of 126 sequences of Rost and Sander, corresponding to a total of 23,348 amino acid positions, as well as the complementary set of 396 non-homologue sequences (62,189 amino acids) prepared by Cuff and Barton (Cuff & Barton, 1999). Both sets can be downloaded at `http://circinus.ebi.ac.uk:8081/pred_res/`.

Finally, we also constructed two more data sets, containing all proteins in PDB which are at least 30 amino acids long, produce DSSP output without chain breaks, and have a resolution of at least 2.5 Å. Furthermore the proteins in both sets have less than 25% identity to any of the 126 sequences of Rost and Sander. In both sets, internal homology is reduced again by Hobohm's #1 algorithm, keeping the PDB sequences with the best resolution. For one set, we use the standard 25% threshold curve for homology reduction. For the other set, however, we raise, the threshold curve by 25%. The set with 25% homology threshold contains 826 sequences, corresponding to a total of 193,249 amino acid positions, while the set with 50% homology threshold contains 1180 sequences (282,303 amino acids).

Thus, to the best of our knowledge, our experiments are based on the currently largest available corpora of non-redundant data. In all but two experiments (see below), profiles were obtained from the HSSP database (Schneider *et al.*, 1997) available at `http://www.sander.embl-heidelberg.de/hssp/`.

2.2 Bidirectional Recurrent Neural Nets

Letting t denote position within a protein sequence, the overall model can be viewed as a probabilistic model that outputs, for each t, a vector $O_t = (o_{1,t}, o_{2,t}, o_{3,t})$ with $o_{i,t} \geq 0$ and $\sum_i o_{i,t} = 1$. The $o_{i,t}$'s are the SS class membership probabilities. The output prediction has the functional form:

$$O_t = \eta(F_t, B_t, I_t) \tag{1}$$

and depends on the forward (upstream) context F_t, the backward (downstream context) B_t, and the input I_t at time t. The vector $I_t \in I\!R^k$ encodes the external input at time t. In the most simple case, where the input is limited to a single amino acid, $k = 20$ by using one-hot encoding. In this case, it is not necessary to include an extra input symbol to represent the terminal portions of the protein. Larger input windows extending over several amino acids are of course also possible. The function η is realized by a neural network $\mathcal{N}_\eta$ (see center and top connections in Figure 2). Thus to guarantee a consistent probabilistic interpretation, the three output units of network $\mathcal{N}_\eta$ are obtained as normalized exponentials (or *softmax*):

$$o_{i,t} = \frac{\exp(net_{i,t})}{\sum_{l=1}^{3} \exp(net_{l,t})} \quad i = 1, 2, 3$$

where $net_{i,t}$ is the activation of the i-th output unit at position t. The performance of the model can be assessed using the usual relative entropy between the estimated and the target distribution.

The novelty of the model is in the contextual information contained in the vectors $F_t \in I\!R^n$ and especially in $B_t \in I\!R^m$. These satisfy the recurrent bidirectional equations:

$$\begin{array}{rcl} F_t & = & \phi(F_{t-1}, I_t) \\ B_t & = & \beta(B_{t+1}, I_t) \end{array} \qquad (2)$$

Here $\phi(\cdot)$ and $\beta(\cdot)$ are learnable non-linear state transition functions. They can be implemented in different forms, but in this paper we assume that they are realized by two NNs, $\mathcal{N}_\phi$ and $\mathcal{N}_\beta$ (left and right subnetworks in Figure 2), with n and m logistic output units, respectively. Thus, $\mathcal{N}_\phi$ and $\mathcal{N}_\beta$ are fed by $n + k$ and $m + k$ inputs, respectively. Here also larger input windows are possible, especially in combination with the weight sharing approach described in (Riis & Krogh, 1996), and different inputs could be used for the computation of F_t, B_t, and O_t. The *forward* chain F_t stores contextual information contained at the left of time t and plays the same role as the internal state in standard RNNs. The novel part of the model is the presence of an additional *backward* chain B_t, in charge of storing contextual information contained at the right of time t, i.e. in the future. The actual form of the bidirectional dynamics is controlled by the connection weights in the subnetworks $\mathcal{N}_\phi$ and $\mathcal{N}_\beta$. As we shall see, these weights can be adjusted using a maximum-likelihood approach. Since eq. 2 involves two recurrences, two corresponding boundary conditions must be specified, at the beginning and the end of the sequence. For simplicity, here we use $F_0 = B_{T+1} = 0$, but it is also possible to adapt the boundaries to the data, extending the technique suggested in (Forcada & Carrasco, 1995) for standard RNNs.

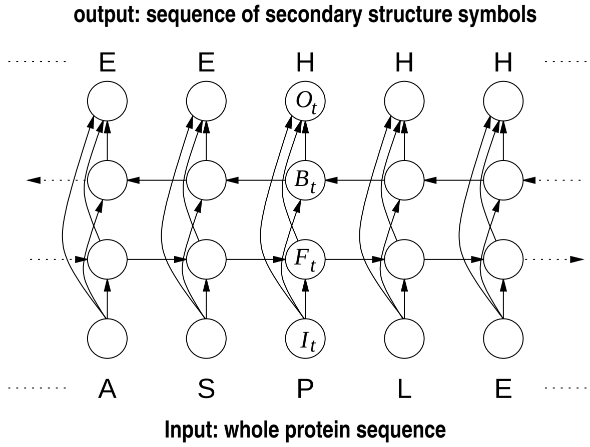

Figure 1: Direct dependencies among the variables involved in a bidirectional BRNN. The boundary conditions are provided by $F_0 = B_{T+1} = 0$, and by the inputs associated with the current protein sequence.

The discrete time index t ranges from 1 to T, the total length of the protein sequence being examined. Hence the probabilistic output O_t is parameterized by a RNN and depends on the input I_t and on the contextual information, from the *entire* protein, summarized into the pair (F_t, B_t). In contrast, in a conventional NN approach this probability distribution depends only on a relatively short subsequence of amino acids. Intuitively, we can think of F_t and B_t as "wheels" that can be "rolled" along the protein. To predict the class at position t, we roll the wheels in opposite directions from the N and C terminus up to position t and then combine what is read on the wheels with I_t to calculate the proper output using η.

The global mapping from the input amino acid sequence to the output SS sequence can be described by the graphical model shown in Fig. 1. The network represents the direct dependencies among the variables I_t, F_t, B_t, O_t, unrolled over time for $t = 1, \ldots, T$. Each node is labeled by one of the variables and arcs represent direct functional dependencies. Interestingly, the same graph would represent a Bayesian network if the relationships amongst I_t, F_t, B_t, O_t were probabilistic, rather than deterministic as in Eqs. 2 and 1. In fact, such probabilistic version of the architecture would yield a bidirectional generalization of IOHMMs. Unlike RNNs, however, propagation of information in bidirectional IOHMMs is computationally expensive. The underlying Bayesian network contains undirected loops that require the use of the junction tree algorithm (Jensen *et al.*, 1990). While inference in this network can be shown to be tractable, the corresponding time complexity of $O(n^3)$ for each time step (here n is the typical number of states in the chains) limits their practical applicability to the SS prediction task (Baldi *et al.*, 2000b).

An architecture resulting from Eqs. 2 and 1 is shown

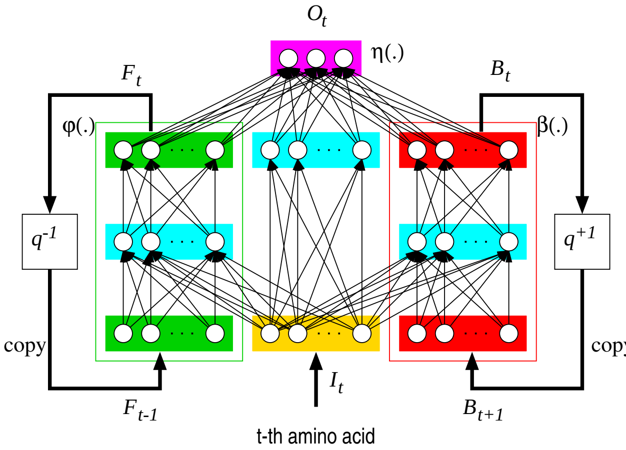

Figure 2: A BRNN architecture.

in Fig. 2 where, for simplicity, all the NNs have a single hidden layer. The hidden state F_t is copied back to the input. This is graphically represented in Fig. 2 using the causal *shift operator* q^{-1} that operates on a generic temporal variable X_t and is symbolically defined as $X_{t-1} = q^{-1}X_t$. Similarly, q, the inverse (or non-causal) shift operator is defined $X_{t+1} = qX_t$ and $q^{-1}q = 1$. As shown in Fig. 2, a non-causal copy is performed on the hidden state B_t. Clearly, removal of $\{B_t\}$ would result in a standard causal RNN.

The number of degrees of freedom of the model depends on the following factors: (1) the dimensions n and m of the forward and backward state vectors; (2) the number of hidden units in the three feedforward networks realizing the state transition and the output functions (see Fig. 2). It is important to remark that the BRNN has been defined as a stationary model, i.e. the connection weights in the networks realizing $\beta(\cdot)$, $\phi(\cdot)$ and $\eta(\cdot)$ do not change over time, i.e with respect to position along the protein. This is a form of weight sharing that reduces the number of free parameter and the risk of overfitting, without necessarily sacrificing the capability to capture distant information.

Since the graph shown in Fig. 1 is acyclic, nodes can be topologically sorted, defining unambiguously the global processing scheme. Using the network unrolled through time, the BRNN prediction algorithm updates all the states F_t from left to right, starting from $F_0 = 0$. Similarly, states B_t are updated from right to left. After forward and backward propagations have taken place, the predictions O_t can be computed. The forward and backward propagations need to be computed from end to end only once per protein sequence. As a result, the time complexity of the algorithm is $O(TW)$, where W is the number of weights and T the protein length. This is the same complexity as feedforward networks fed by a fixed-size window. In the case of BRNNs, W typically grows as $O(n^2)$ and the actual number of weights can be reduced by limiting the number of hidden units in the subnetworks for $\phi(\cdot)$ and $\beta(\cdot)$. Thus, inference in BRNNs is more efficient than in bidirectional IOHMMs, where complexity is $O(Tn^3)$ (Baldi *et al.*, 2000b).

Learning can be formulated as a maximum likelihood estimation problem, where the log likelihood is essentially the relative entropy function between the predicted and the true conditional distribution of the secondary structure sequence given the input amino acid sequence:

$$\ell = \sum_{\text{sequences}} \sum_{t=1}^{T} z_{i,t} \log o_{i,t} \tag{3}$$

with $z_{i,t} = 1$ if the SS at position t is i, and $z_{i,t} = 0$ otherwise. The optimization problem can be solved by gradient ascent. The only difference with respect to standard RNNs is that gradients must be computed by taking into account non-causal temporal dependencies. Because the unrolled network is acyclic, the generalized backpropagation algorithm can be derived as a special case of the backpropagation through structure algorithm (Frasconi *et al.*, 1998). Intuitively, the error signal, is first injected into the leaf nodes, corresponding to the output variables O_t. The error is then propagated through time in both directions, by following any reverse topological sort of the unrolled network (see Figure 1). Obviously, this step also involves backpropagation through the hidden layers of the NNs. Since the model is stationary, weights are shared among the different replicas of the NNs at different time steps. Hence, the total gradient is simply obtained by summing all the contributions associated with different time steps.

To speed-up convergence, we found it convenient to adopt an on-line weight updating strategy. Once gradients relative to a single protein have been computed, weights are immediately updated. This scheme was enriched also with a heuristic adaptive learning rate algorithm that progressively reduces the learning rate if the average error reduction within a fixed number of epochs falls below a given threshold.

2.3 Long-ranged dependencies

One of the principal difficulties when training standard RNNs is the problem of vanishing gradients (Bengio *et al.*, 1994). Intuitively, in order to contribute to the output at position or time t, the input signal at time $t-\tau$ must be propagated in the forward chain through τ replicas of the NN that implements the state transition function. However, during gradient computation, error signals must be propagated backward along the same path. Each propagation can be interpreted as the product between the error vector and the Jacobian matrix associated with the transition function. Unfortunately, when the dynamics develop attractors that allow the system to reliably store past information, the norm of the Jacobian is < 1. Hence, when τ is large, gradients of the error at time t with respect to inputs at time $t-\tau$ tend to vanish exponentially. Similarly, in the case of BRNNs, error propagation in both the forward and the backward chains is subject to exponential decay. Thus, although the model has in principle the capability of storing remote information, such information cannot be learnt effectively. Clearly, this is a theoretical argument and its practical impact needs to be evaluated on a per case basis.

In practice, in the case of proteins, the BRNN can reliably utilize input information located within about ± 15 amino acids (i.e., the total effective window size is about 31). This was empirically evaluated by feeding the model with increasingly long protein fragments. We observed that the average predictions at the central residues did not significantly change if fragments were extended beyond 41 amino acids. This is an improvement over standard NNs with input window sizes ranging from 11 to 17 amino acids (Rost & Sander, 1994; Riis & Krogh, 1996). Yet, there is presumably relevant information located at longer distances that our models have not been able to discover so far.

To limit this problem, we propose a remedy motivated by recent studies (Lin *et al.*, 1996) suggesting that the vanishing gradients problem can be mitigated by the use of an explicit delay line applied to the output, which provides shorter paths for the effective propagation of error signals. Unfortunately, this idea cannot be applied directly to BRNNs since output feedback, combined with bidirectional propagation, would generate cycles in the unrolled network. A similar mechanism, however, can be implemented using the following modified dynamics:

$$\begin{aligned} F_t &= \phi(F_{t-1}, F_{t-2}, \ldots, F_{t-s}, I_t) \\ B_t &= \beta(B_{t+1}, B_{t+2}, \ldots, B_{t+s}, I_t). \end{aligned} \tag{4}$$

The explicit dependence on forward or backward states introduces *shortcut* connections in the graphical model, forming shorter paths along which gradients can be propagated. This is akin to introducing higher order Markov chains in the probabilistic version. However, unlike Markov chains where the number of parameters would grow exponentially with s, in the present case the number of parameters grows only linearly with s. To reduce the number of parameters, a simplified version of Eq. 4 limits the dependencies to state vectors located s residues apart from t:

$$\begin{aligned} F_t &= \phi(F_{t-1}, F_{t-s}, I_t) \\ B_t &= \beta(B_{t+1}, B_{t+s}, I_t). \end{aligned} \tag{5}$$

Another variant of the basic architecture which also allows to increase the effective window size consists in feeding the output networks with a window in the forward and backward state chains. In this case, the prediction is computed as

$$O_t = \eta(F_{t-s}, \ldots, F_{t+s}, B_{t-s}, \ldots, B_{t+s}, I_t). \tag{6}$$

Notice that the window can extend in the past and the future of t on both vectors F_t and B_t.

2.4 Multiple Alignments and Mixture of Estimators

Multiple alignments and mixture of estimators are two algorithmic ideas which have been shown to be very effective in the protein SS prediction task. Both of them have been incorporated in our BRNN-based system. Multiple predictors can be obtained by varying the size of the BRNN, as controlled by the dimensions of the state vectors and the number of hidden units. Moreover, different predictors can be obtained using profiles, or multiple alignments, in input mode (Rost & Sander, 1994), or in output mode (Riis & Krogh, 1996). In the first case, instead of a code for the current amino acid, the input I_t contains the relative frequencies of amino acids at position t in the protein family. In the second case, a separate sequence of SS predictions is obtained for each aligned protein, and then all the predictions are averaged in each column k possibly in combination with a weighting scheme. Since the two methods give different prediction errors—the input mode, for instance, yields slightly more accurate beta-sheet predictions—it is reasonable to build ensembles containing both types of predictors.

The setup developed in (Lund *et al.*, 1997) has also been used to build profiles for sequences not present in the HSSP data base. The setup contains methods similar to the ones applied earlier by Sander and Schneider (1991), where the key parameter is the similarity threshold (in terms of identical residues in a particular pairwise alignment). In (Lund *et al.*, 1997) the similarity threshold, dividing sequences with structural homology from those without, had the form $I < \frac{290}{\sqrt{L}}$, where L is the length of the alignment. The threshold was then used to build a profile for all the relevant PDB entries from the matches found in the SWISS-PROT database. We have also derived profiles by running the BLAST program, with default parameters, against the major publicly available protein databases

3 Implementation and Results

We experimented with different DSSP class assignments to the three SS classes (see below). When we do not specify otherwise, the default mapping we use is as follows: α is formed by DSSP class H, β by E, and γ by everything else (including DSSP classes G, S, T, B, I, and "."). Otherwise we use the "harder" assignment of the CASP competition (Moult *et al.*, 1997; CASP3, 1998), where α contains H, and G, while β contains E, and B. Other assignments used in the literature include (Riis & Krogh, 1996), where α contains DSSP classes H, G, and I. A study of the effect of various assignments on the prediction performance can be found in (Cuff & Barton, 1999).

We carried out many experiments to tune up and evaluate the prediction system. The main results are summarized in Tables 1-4. In a first set of experiments, basedon the 824 sequences, we reserved 2/3 of the available data for training, and 1/3 for testing, chosen at random. We trained several BRNNs of different sizes and different architectural details. Training was stopped using a fixed threshold on the reduction in error, rather than other adaptive criteria such as early stopping. In all experiments, we set $n = m$ and tried different values for n and s (see Eq. 6). The number of free parameters varied from about 1400 to 2900. Qualitatively we observed that using $s > 0$ can improve accuracy, but increasing n beyond 12 does not help because of overfitting. Results for this method, without using profiles, are summarized in the first rows of Table 1. By comparison, we also trained several feedforward NNs on the same data. The best feedforward NN achieved Q_3 =67.2% accuracy using a window of 13 amino acids. By enriching the feedforward architecture with adaptive input encoding and output filtering, as in (Riis & Krogh, 1996), 68.5% accuracy was achieved (output filtering actually increases the length of the input window). Hence, the best BRNN outperforms our best feedforward network, even when additional architectural design is included.

Subsequent experiments included the use of profiles. Table 1 reports the best results obtained by using multiple alignments, both at the input and output levels. Profiles at the input level consistently yielded better results. The best feedforward networks trained in the same conditions achieve $Q_3 = 73.0\%$ and 72.3%, respectively.

In a second set of experiments based on the 824 sequences, we combined several BRNNs, trained with 1/3-2/3 data splitting, to form an ensemble, as in (Krogh & Vedelsby, 1995), using a simple averaging scheme. Different networks were obtained by varying architectural details such as n, s, and the number of hidden units. Combining the 6 architectures at the bottom of Table 1, using profiles at the input level only, we obtained the best accuracy $Q_3 = 75.1\%$, measured in this case using 7-fold cross validation. We also tried to include in the ensemble a set of 4 BRNNs using profiles at the output level, but performance in this way slightly decreased to 75.0%. We also combined 8 different BRNNs, 6 with input profiles and 2 with output profiles. The percentage of correctly predicted residues on the test set of 59,865 residues is 75.07%. Because of the absence of any improvements from including output profiles, in the rest of this work we focus on ensembles made up of the 6 architectures at the bottom of Table 1.

In order to study the capabilities of the model to capture long-ranged information, we performed the following experiments For each protein and for each amino acid

Table 1: Experimental results using a single BRNN and 1/3 of the data as test set. n is the length of the context vectors F_t and $B_t (n = m)$. s is the number of additional consecutive context vectors in the future and the past of both F_t and B_t that are used in the prediction. h_ϕ, h_β and h_η are the number of hidden units for the transition networks $\mathcal{N}_\phi$, $\mathcal{N}_\beta$, and the output network $\mathcal{N}_\eta$, respectively. We always set $h_\phi = h_\beta$. W is the number of parameters.

Profiles	n	s	h_ϕ	h_η	W	Accuracy (Q_3)
No	7	2	8	11	1611	68.7%
No	9	2	8	11	1899	68.8%
No	7	3	8	11	1919	68.6%
No	8	3	9	11	2181	68.8%
No	20	0	17	11	2821	67.7%
Output	9	2	8	11	1899	72.6%
Output	8	3	9	11	2181	72.7%
Input	9	2	8	11	1899	73.3%
Input	8	3	9	11	2181	73.4%
Input	12	3	9	10	2757	73.6%
Input	7	3	8	11	1919	73.4%
Input	8	3	9	10	2045	73.4%
Input	12	3	9	11	2949	73.2%

position t, we fed the BRNN mixture described above with a sequence obtained by replacing all inputs outside the range $[t - \tau, t + \tau]$ with null values. The experiment was repeated for different values of τ from 0 to 23. Figure 3 shows the results. Each diagram is a normalized row of the test set confusion table, for the semi-window size τ ranging from 0 to 23. So for example the line labeled $H \to C$ in the first diagram is the percentage of helices classified as coils, as a function of τ. The curves are almost stable for $\tau > 15$. Although the model is not sensitive to *very* distant information, it should be remarked that typical feedforward nets reported in the literature do not exploit information beyond $\tau = 8$.

To further explore the long-ranged information problem we conducted another set of experiments using BRNNs with simplified shortcuts (see eq. 5). In this case, as for the results reported in Table 1, we used a single model (rather than a mixture) and the test set method (1/3 of the available data) for measuring accuracy. We tried all values of s from 1 to 10, but in no case could we observe a significant performance improvement on the test set. Interestingly, our experiments showed that using shortcuts reduces the convergence difficulties associated with vanishing gradients (see Sec. 2.3): accuracy on the training set increased from 75.7% using no shortcuts to 76.9% with $s = 3$. On the other hand, the gap between training set and test set performance also increased. Thus overfitting offsets the convergence improvement, probably because long-ranged information is too sparse and noisy.

We run another experiment by training our system on

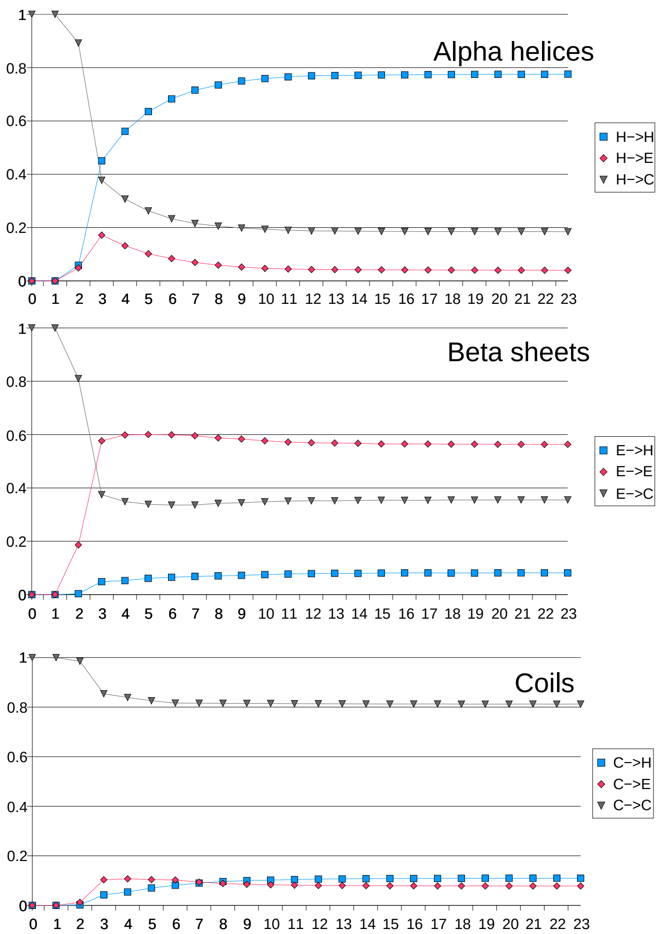

Figure 3: Distant information exploited by the BRNN. The horizontal axis represents τ, the distance from a given position beyond which all entries are set to null values. Each curve represents a normalized row of the test-set confusion matrix.

the 824 sequences and using the official test sequences used at the 1998 CASP3 competition, this time using the CASP assignment. The profiles used for testing here were not obtained from the HSSP data base but by the method described above. The CASP3 competition was won by one of the two programs entered by D. Jones, which selected 23 out of 35 proteins obtaining a performance of $Q_3 = 77.6\%$ per protein, or $Q_3 = 75.5\%$ per residue. We evaluated that system on the whole set of 35 proteins by using Jones' prediction server at `http://insulin.bio.warwick.ac.uk/psiform.html`. It achieved 76.2% per protein and $Q_3 = 74.3\%$ per residue. On the same 35 sequences our system achieved 74.6% per protein and $Q_3 = 73.0\%$ per residue. A test set of 35 proteins is relatively small for drawing general conclusions. Still, we believe that this result confirms the effectiveness of the proposed model, especially in consideration of the fact that Jones' system builds upon more recent profiles based on iterative search in the TrEMBL database (Bairoch & Apweiler, 1999). These

Table 2: First confusion matrix derived with an ensemble of 6 BRNNs with 2/3-1/3 data splitting. First row provides percentages of predicted helices, sheets, and coils within (DSSP-assigned) helices.

	pred α	pred β	pred γ
α	80.03%	2.88%	17.09%
β	4.68%	62.01%	33.31%
γ	10.60%	9.62%	79.78%

profiles contain many more sequences than our training profiles which are based on the older HSSP profile approach, leaving room for further improvements of our system.

To further compare our system with other predictors, as in (Cuff & Barton, 1999), we also trained an ensemble of BRNNs using the 126 sequences in the Rost and Sander data set. The performance on the 396 test sequences prepared by Cuff and Barton is $Q_3 = 72.0\%$. This is slightly better than the 71.9% score for the single best predictor (PHD) amongst (DSC, PHD, NNSSP, and PREDATOR) reported in (Cuff & Barton, 1999). This result is also achieved with the CASP class assignment.

Finally, we also trained an ensemble of 6 BRNNs using the set containing 826 sequences with less than 25% identity to the 126 sequences of Rost and Sander. When tested on the 126 sequences, the system achieves $Q_3 = 74.7\%$ per residue, with correlation coefficients $C_\alpha = 0.692$, $C_\beta = 0.571$, and $C_\gamma = 0.544$. This is again achieved with the harder CASP assignment. In contrast, the $Q_3 = 75.1\%$ described above and obtained by 7 fold cross-validation on 824 sequences was obtained with the easier class assignment (H$\to \alpha$, E$\to \beta$, the rest $\to \gamma$). The same experiment was performed using the larger training set of 1,180 sequences having also less than 25% identity with the 126 sequences of Rost and Sander, but with a less stringent redundancy reduction requirement. In this case, and with the same hard assignment, the results are $Q_3 = 75.3\%$ with correlation coefficients $C_\alpha = 0.704$, $C_\beta = 0.583$, and $C_\gamma = 0.550$. The corresponding confusion matrices are given in Tables 2 and 3. We also confirm the observation made by Cuff and Barton (1999) concerning the impact of the class assignment: the same system tested with the default class assignment achieves a performance of $Q_3 = 77.3\%$.

4 Discussion

Given the large number of protein sequences available through genome and other sequencing projects, even small percentage improvements in SS prediction can be significant. The system presented here achieves an overall performance close to 76% correct classification, at

Table 3: Same as above. First row provides percentages of (DSSP-assigned) helices, sheets, and coils within the predicted helices.

	α	β	γ
pred α	81.29%	3.46%	15.25%
pred β	4.68%	73.20%	22.12%
pred γ	11.07%	15.69%	73.24%

Table 4: Summary of main performance results.

Training sequences	Test sequences	Class assignment	Performance per residue
824 (2/3)	824 (1/3)	default	$Q_3 = 75.1\%$
824	35	CASP	$Q_3 = 73.0\%$
126	396	CASP	$Q_3 = 72.0\%$
826	126	CASP	$Q_3 = 74.7\%$
1180	126	CASP	$Q_3 = 75.3\%$
1180	126	CASP	$Q_3 = 75.9\%$

least comparable to the best existing predictors, but using a different NN approach based on recurrent networks and bidirectional dynamics. A thorough comparison with D. Jones system (winner of CASP3) has not been carried out at this time, and details about his method remain unpublished. By selecting a subset of 24 sequences out of the 35 CASP3 sequences according to our own criteria (prediction confidence above a certain threshold), we can match Jones best results. Such a comparison, however, is not entirely fair since it does not satisfy the conditions of a blind prediction.

Interestingly, we have circumstantial evidence that the two methods behave in significantly different ways: there exist sequences for which our method achieves over 80% correct prediction, while Jones method is below 70%, and vice versa. Such differences require further study, and suggest that both methods could be combined to further improve the results. In particular, if the advantage of the method of Jones resides in the type of alignments used, similar alignments could be incorporated in the BRNN approach. While there is room for performance improvement, one should also not forget that 100% correct prediction, from the primary sequence alone, is probably unachievable if nothing else because a minority of proteins may not fold spontaneously, or because beta-sheet partner strands may be located on a different chain.

Most importantly, perhaps, we have developed here new algorithmic ideas that begin to address the problem of long-ranged dependencies. Unlike feedforward networks, BRNNs can possibly prove advantageous from this point of view and our preliminary experiments encourage further investigations. There are additional directions in which this work could be extended including many architectural variations. In addition to the use of larger input windows for I_t, one may consider non-symmetrical chains for the past and the future, and the use of priors on the parameters and/or the architecture together with a maximum a posteriori learning approach. It may also be advantageous to use an array of "wheels", instead of just two, of various memory capacity, rolling in different directions along the protein and possibly over shorter distances. It is also worth noting that using multi-layered perceptrons for implementing $\beta(.)$ and $\phi(.)$ is just one of the available options. For example, a generalization of second-order RNN (Giles *et al.*, 1992) is an easily conceivable alternative parameterization.

Finally, it is clear that the ideas introduced can be applied to other problems in bioinformatics, as well as other domains, where non-causal dynamical approaches are suitable. Obvious candidates for further tests of the general method include the prediction of beta-sheet partners, of DNA exon/intron boundaries and promoter regions, of RNA secondary structure, and of protein functional domains, such as signal peptides.

Acknowledgements

The work of PB has been initially supported by an NIH SBIR grant to Net-ID, Inc., and currently by a Laurel Wilkening Faculty Innovation award at UCI. The work of SB is supported by a grant from the Danish National Research Foundation.

References

Angluin, D. & Smith, C. (1987). Inductive inference. In *Encyclopedia of Artificial Intelligence*. J. Wiley and Sons, New York, pp. 409–418.

Bairoch, A. & Apweiler, R. (1999). The SWISS-PROT protein sequence data bank and its supplement TrEMBL in 1999. *Nucleic Acids Res*, **27**, 49–54.

Baldi, P. & Brunak, S. (1998). Bioinformatics: The Machine Learning Approach. MIT Press, Cambridge, MA.

Baldi, P., Brunak, S., Chauvin, Y. & Nielsen, H. (2000a). Assessing the accuracy of prediction algorithms for classification: an overview. *Bioinformatics*. In press.

Baldi, P., Brunak, S., Frasconi, P., Pollastri, G. & Soda, G. (2000b). Bidirectional dynamics for protein secondary structure prediction. In Sun, R. & Giles, L., (eds.) *Sequence Learning: Paradigms, Algorithms, and Applications*.

Baldi, P. & Chauvin, Y. (1996). Hybrid modeling, HMM/NN architectures, and protein applications. *Neural Computation*, **8**, 1541–1565.

Bengio, Y. & Frasconi, P. (1996). Input-output HMM's for sequence processing. *IEEE Trans. on Neural Networks*, **7**, 1231–1249.

Bengio, Y., Simard, P. & Frasconi, P. (1994). Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, **5**, 157–166.

Bernstein, F. C. & et al. (1977). The protein data bank: A computer based archival file for macromolecular structures. *J. Mol. Biol.*, **112**, 535–542.

CASP3 (1998). Third community wide experiment on the critical assessment of techniques for protein structure prediction. Unpublished results available in `http://predictioncenter.llnl.gov/casp3`.

Cleeremans, A. (1993). Mechanisms of Implicit Learning. Connectionist Models of Sequence Processing. MIT Press.

Cuff, J. A. & Barton, G. J. (1999). Evaluation and improvement of multiple sequence methods for protein secondary structure prediction. *Proteins*, **34**, 508–519.

Eisenberg, D. (1997). Into the black night. *Nat. Struct. Biol.*, **4**, 95–97.

Forcada, M. & Carrasco, R. (1995). Learning the initial state of a second-order recurrent neural network during regular-language inference. *Neural Computation*, **7**, 923–930.

Frasconi, P., Gori, M. & Sperduti, A. (1998). A general framework for adaptive processing of data structures. *IEEE Trans. on Neural Networks*, **9**, 768–786.

Frishman, D. & Argos, P. (1995). Knowledge-based secondary structure assignment. *Proteins*, **23**, 566–579.

Giles, C. L., Miller, C. B., Chen, D., Chen, H. H., Sun, G. Z. & Lee, Y. C. (1992). Learning and extracting finite state automata with second-order recurrent neural networks. *Neural Computation*, **4**, 393–405.

Hobohm, U., Scharf, M., Schneider, R. & Sander, C. (1992). Selection of representative data sets. *Prot. Sci.*, **1**, 409–417.

Jensen, F. V., Lauritzen, S. L. & Olosen, K. G. (1990). Bayesian updating in recursive graphical models by local computations. *Comput. Stat. Quarterly*, **4**, 269–282.

Jones, D. T. (1999). Protein secondary structure prediction based on position-specific scoring matrices. *J. Mol. Biol.*, **292**, 195–202.

Kabsch, W. & Sander, C. (1983). Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features. *Biopolymers*, **22**, 2577–2637.

Krogh, A. & Mitchinson, G. (1995). Maximum entropy weighting of aligned sequences of proteins of DNA. In *et al.*, C. R., (ed.) *Proc. 3rd Int. Conf. on Intelligent Systems for Molecular Biology*. Menlo Park, CA, pp. 215–221.

Krogh, A. & Vedelsby, J. (1995). Neural network ensembles, cross validation, and active learning. In Tesauro, G., Touretzky, D. & Leen, T., (eds.) *NIPS 7*. The MIT Press, pp. 231–238.

Lin, T., Horne, B. G., Tiño, P. & Giles, C. L. (1996). Learning long-term dependencies in NARX recurrent neural networks. *IEEE Transactions on Neural Networks*, **7**, 1329–1338.

Lund, O., Frimand, K., Gorodkin, J., Bohr, H., Bohr, J., Hansen, J. & Brunak, S. (1997). Protein distance constraints predicted by neural networks and probability density functions. *Prot. Eng.*, **10**, 1241–1248.

Moult, J., Hubbard, T., Bryant, S. H., Fidelis, K. & Pedersen, J. T. (1997). Critical assessment of methods of protein structure prediction (CASP): Round II. *Proteins*, **29**, 2–6. Supplement 1.

Myers, E. W. & Miller, W. (1988). Optimal alignments in linear space. *Comput. Appl. Biosci.*, **4**, 11–7.

Pearson, W. R. (1990). Rapid and sensitive sequence comparison with FASTP and FASTA. *Meth. Enzymol.*, **183**, 63–98.

Qian, N. & Sejnowski, T. J. (1988). Predicting the secondary structure of glubular proteins using neural network models. *J. Mol. Biol.*, **202**, 865–884.

Richards, F. M. & Kundrot, C. E. (1988). Identification of structural motifs from protein coordinate data: secondary structure and first-level supersecondary structure. *Proteins*, **3**, 71–84.

Riis, S. K. & Krogh, A. (1996). Improving prediction of protein secondary structure using structured neural networks and multiple sequence alignments. *J. Comput. Biol.*, **3**, 163–183.

Rost, B. & Sander, C. (1993). Improved prediction of protein secondary structure by use of sequence profiles and neural networks. *Proc. Natl. Acad. Sci. USA*, **90**, 7558–7562.

Rost, B. & Sander, C. (1994). Combining evolutionary information and neural networks to predict protein secondary structure. *Proteins*, **19**, 55–72.

Schneider, R., de Daruvar, A. & Sander, C. (1997). The HSSP database of protein structure-sequence alignments. *Nucleic Acids Res.*, **25**, 226–230.