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

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

Paolo Frasconi
Dept. of Electrical and Electronic Engineering
University of Cagliari
09123 Cagliari, Italy

Gianluca Pollastri, Giovanni Soda
Dept. of Informatics and Systems
University of Florence
50139 Florence, Italy

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 exceeding 75% 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: E-mail: *docbazz@tin.it*, *pfbaldi@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 and Brunak, 1998) for a detailed review]. In particular, many different feedforward neural network (NN) architectures have been applied to this task (Qian and Sejnowski, 1988; Rost and Sander, 1994; Riis and 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 $C_\alpha = 0.41$, $C_\beta = 0.31$, and $C_c = 0.41$. Throughout this article, we use the standard performance measures reviewed in (Baldi *et al.*, 1999a). Prediction of SS based on single sequences and local windows seem to be limited to < 65–67% 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 and Sejnowski, 1988), for a long time the best SS prediction performance has been achieved by the PHD server (Rost and Sander,

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

1993; 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 and 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 and Mitchinson, 1995). In spite of a considerable amount of architectural design, the final performance with multiple alignments is practically identical to (Rost and Sander, 1994) with an overall accuracy of $Q_3 = 71.3\%$, and correlations $C_\alpha = 0.59$, $C_\beta = 0.50$, and $C_c = 0.41$.

At the 1998 CASP3 competition, the best results were obtained by one of the two programs entered by D. Jones, using a relatively simple 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. While the methods and the results have not yet been published, the improvement seem to result in part from the type of alignments used, characterized by a different set of homology requirements.

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 and 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) (both with first (Cleeremans, 1993) and second order (Giles *et al.*, 1992) connections) and input-output hidden Markov models (IOHMM) (Bengio and Frasconi, 1996; Baldi and Chauvin, 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, i.e. 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 (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 and 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. The first high quality data used in this study was extracted from the Brookhaven Protein Data Bank (PDB) (Bernstein and et al., 1977) release 77 (July 1996) 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 and 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 and Miller, 1988; Pearson, 1990) using the pam120 matrix, with gap penalties -12, -4. Finally, 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 and Sejnowski, 1988). We also used 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. Thus our experiments, are based on one of the largest available corpora of non-redundant data covering 1122 sequences and 267,123 combined amino acid positions. 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

The core of the model is an adaptive network based on the following vector dynamics equations:

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

where F_t and B_t are two state vectors in $\mathbb{R}^n$ and $\mathbb{R}^m$, respectively, and t is a discrete time index ranging from 1 to T, the total length of the protein sequence being examined. 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. The vector $I_t \in \mathbb{R}^k$ encodes the external input at time t (using one-hot encoding in the case of amino acids, we have $k = 20$). Thus, $\mathcal{N}_\phi$ and $\mathcal{N}_\beta$ are fed by $n + k$ and $m + k$ inputs, respectively. Larger input windows extending over several amino acids are also possible. Since eq. 1 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 and Carrasco, 1995) for standard RNNs. Note that, since the model does not employ an input window, it is not necessary to include an extra input symbol to represent the terminal portions of the protein. 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. The output prediction is computed as:

$$O_t = \eta(F_t, B_t, I_t) \qquad (2)$$

where $O_t \in [0, 1]^3$ encodes the predicted class and η is realized by a third neural network $\mathcal{N}_\eta$ (see center and top connections in Figure 2). As a result, O_t depends on the input I_t and on the contextual information, from the *entire* protein, summarized into the pair (F_t, B_t). The overall model can also be viewed as a probabilistic model where the SS class probabilities provided by the outputs of network $\mathcal{N}_\eta$ are parameterized using recurrent neural networks. To guarantee a consistent probabilistic

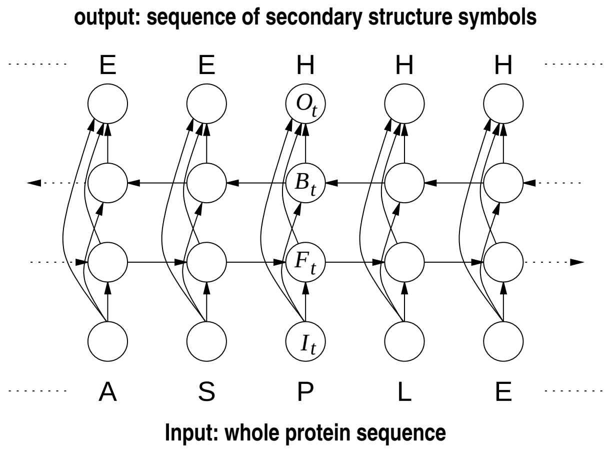

Figure 1: Direct dependencies among the variables involved in a bidirectional BRNN.

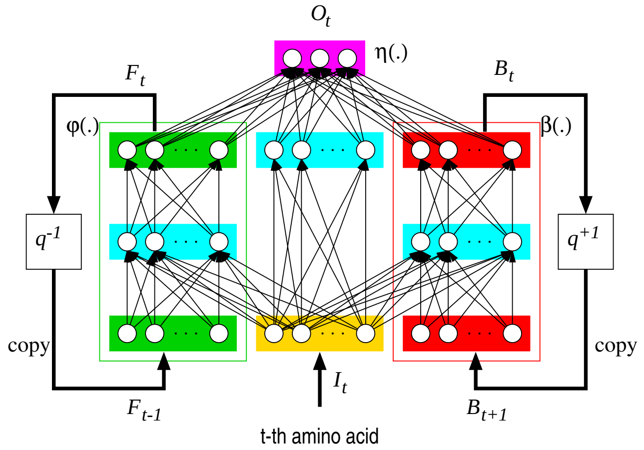

Figure 2: A BRNN architecture.

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. In this way $o_{i,t}$ represents the probability of SS class i at time t, conditioned on the entire protein. In contrast, in a conventional NN approach this probability depends only on a relatively short subsequence of amino acids.

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. 1 and 2. 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.*, 1999b).

An architecture resulting from Eqs. 1 and 2 is shown 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 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. 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.*, 1999b).

Learning can be formulated as a maximum likelihood estimation problem, where the log likelihood is essentially the cross-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 SS at site 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 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 and Sander, 1994; Riis and Krogh, 1996). Yet, there is presumably relevant information located at longer distances that our model 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:

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

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

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

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 and Sander, 1994), or in output mode (Riis and 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. 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 select the profiles. 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.

3 IMPLEMENTATION AND RESULTS

We carried out several preliminary experiments to tune up and evaluate the prediction system. DSSP classes were assigned to three secondary structure classes α,β, and γ as follows: α is formed by DSSP class H, β by E, and γ by everything else (including DSSP classes F,S,T,B, and I). This assignment is slightly different from other assignments reported in the literature. For example, in (Riis and Krogh, 1996), α contains DSSP classes H, G, and I. In the CASP competition (Moult and et al., 1997; CASP3, 1998) α contains H, and G, while β contains E, and B.

In a first set of experiments, we reserved 2/3 of the available data for training, and 1/3 for testing. We trained several BRNNs of different sizes and different architectural details. In all experiments, we set $n = m$ and we tried different values for n and k (see Eq. 6). The number of free parameters varied from about 1400 to 2600. Qualitatively we observed that using $k > 0$ 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 and 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.

Table 1: Experimental results using a single BRNN and 1/3 of the data as test set. 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 $h_\phi = h_\beta$.

Profiles	n	k	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	2601	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	2565	73.6%

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, we combined several BRNNs to form an ensemble, as in (Krogh and Vedelsby, 1995), using a simple averaging scheme. Different networks were obtained by varying architectural details such as n, k, and the number of hidden units. Combining 6 networks using profiles at the input level 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%. In order to study the capabilities of the model in capturing long ranged information we performed the following experiment. For each protein and for each amino acid 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 infor-

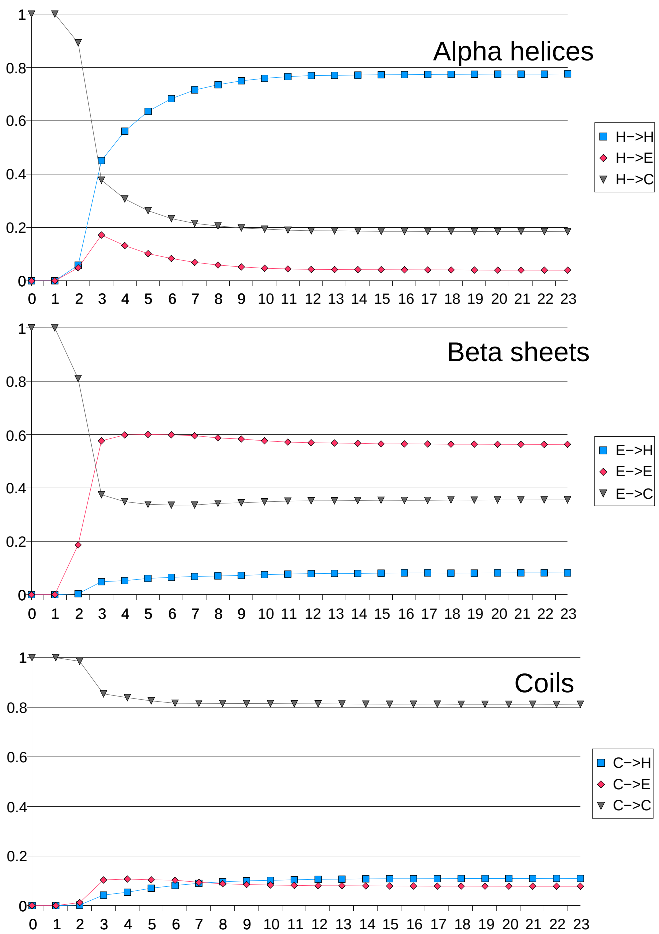

Figure 3: Distant information exploited by the BRNN. See text for explanation.

mation, it should be remarked that typical feedforward nets reported in the literature do not exploit information beyond $\tau = 8$.

To further explore the long-range 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 we could 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 offset the convergence improvement, probably because long-range information is too sparse and noisy.

Finally, we run experiments by training on all the available data and using the official test sequences used at the 1998 CASP3 competition. In this case we adopted a slightly different class assignment for training (DSSP classes H, G, and I were merged together). 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://137.205.156.147/psiform.html`. It achieved $Q_3 = 74.3\%$ per residue and 76.2% per protein. On the same 35 sequences our system achieved $Q_3 = 73.0\%$ per residue and 74.6 per protein. 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 from TrEMBL database (Bairoch and Apweiler, 1999). These profiles contain many more sequences than our profiles which are based on the older HSSP database, leaving room for further improvements of our system.

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 of over 75% correct classification, at 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 at this time, and details about his method remain unpublished. By selecting our own subset of sequences from the 35 CASP3 sequences according to some criteria, we can match Jones results. Such a comparison, however, is not 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-range 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 architectural variations such as the use of larger input windows in the BRNN architecture, 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 is also worth noting that using multi-layered perceptrons for implementing $\beta()$ and $\phi()$ is just one of the available options. For example, a generalizations 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 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 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

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

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

P. Baldi and S. Brunak. *Bioinformatics: The Machine Learning Approach.* MIT Press, Cambridge, MA, 1998.

P. Baldi and Y. Chauvin. Hybrid modeling, HMM/NN architectures, and protein applications. *Neural Computation*, 8(7):1541–1565, 1996.

P. Baldi, S. Brunak, Y. Chauvin, and H. Nielsen. Assessing the accuracy of prediction algorithms for classification: an overview. Submitted for publication, 1999.

P. Baldi, S. Brunak, P. Frasconi, G. Pollastri, and G. Soda. Bidirectional dynamics for protein secondary structure prediction. IJCAI-99 workshop on neural, symbolic, and reinforcement methods for sequence learning, 1999.

Y. Bengio and P. Frasconi. Input-output HMM's for sequence processing. *IEEE Trans. on Neural Networks*, 7(5):1231–1249, 1996.

Y. Bengio, P. Simard, and P. Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5(2):157–166, 1994.

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

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

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

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

M.L. Forcada and R.C. Carrasco. Learning the initial state of a second-order recurrent neural network during regular-language inference. *Neural Computation*, 7(5):923–930, 1995.

P. Frasconi, M. Gori, and A. Sperduti. A general framework for adaptive processing of data structures. *IEEE Trans. on Neural Networks*, 9(5):768–786, 1998.

C. L. Giles, C. B. Miller, D. Chen, H. H. Chen, G. Z. Sun, and Y. C. Lee. Learning and extracting finite state automata with second-order recurrent neural networks. *Neural Computation*, 4(3):393–405, 1992.

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

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

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

A. Krogh and G. Mitchinson. Maximum entropy weighting of aligned sequences of proteins of DNA. In C. Rawlings *et al.*, editor, *Proc. 3rd Int. Conf. on Intelligent Systems for Molecular Biology*, pages 215–221. Menlo Park, CA, 1995.

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

T. Lin, B. G. Horne, P. Tiño, and C. L. Giles. Learning long-term dependencies in NARX recurrent neural networks. *IEEE Transactions on Neural Networks*, 7(6):1329–1338, November 1996.

O. Lund, K. Frimand, J. Gorodkin, H. Bohr, J. Bohr, J. Hansen, and S. Brunak. Protein distance constraints predicted by neural networks and probability density functions. To appear., 1997.

John Moult and et al. Critical assessment of methods of protein structure prediction (CASP): Round II. *Proteins*, 29(S1):2–6, 1997. Supplement 1.

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

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

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

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

B. Rost and C. Sander. Improved prediction of protein secondary structure by use of sequence profiles and neural networks. *Proc. Natl. Acad. Sci. USA*, 90(16):7558–7562, 1993.

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

C. Sander and R. Schneider. Database of homology-derived protein structures and the structural meaning of sequence alignment. *Proteins*, 9:56–68, 1991.

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