Available online at www.sciencedirect.com
SCIENCE DIRECT
Neural Networks 18 (2005) 1029–1039
Neural Networks
www.elsevier.com/locate/neunet

2005 Special Issue

Learning protein secondary structure from sequential and relational data

Alessio Ceroni[a,*], Paolo Frasconi[a], Gianluca Pollastri[b]

[a]*Dipartimento di Sistemi e Informatica, Università degli Studi di Firenze Via Santa Marta, Firenze, Italy*
[b]*Department of Computer Science, University College Dublin Belfield, Dublin 4, Ireland*

Abstract

We propose a method for sequential supervised learning that exploits explicit knowledge of short- and long-range dependencies. The architecture consists of a recursive and bi-directional neural network that takes as input a sequence along with an associated interaction graph. The interaction graph models (partial) knowledge about long-range dependency relations.

We tested the method on the prediction of protein secondary structure, a task in which relations due to beta-strand pairings and other spatial proximities are known to have a significant effect on the prediction accuracy. In this particular task, interactions can be derived from knowledge of protein contact maps at the residue level. Our results show that prediction accuracy can be significantly boosted by the integration of interaction graphs.

Keywords: Recursive neural networks; Relational learning; Protein secondary structure prediction; Protein contact maps

1. Introduction

We consider the sequential supervised learning problem, sometimes also referred to as *sequence translation* (see e.g. Dietterich, 2002, for a review). In this setting, each example is a pair of input–output sequences $(\boldsymbol{x}, \boldsymbol{y})$. The objective of learning consists of approximating the probabilistic dependency between sequence pairs by a sequential transduction, i.e. a function f that maps an input sequence $\boldsymbol{x}$ into a corresponding output sequence $f(\boldsymbol{x})$. The sequential nature of $f(\boldsymbol{x})$ is a major difference with respect to other sequence learning problems like sequence classification or sequence regression, in which $f(\boldsymbol{x})$ is a scalar. In molecular biology, several one-dimensional prediction tasks (e.g. protein secondary structure and solvent accessibility) naturally fit this formulation.

Solving the sequential supervised learning problem can be difficult when sequences are long. Symbolic methods that attempt to induce translating automata cannot easily handle uncertainty and real-valued information. The study of kernel machines for this kind of problems is quite recent and their practical applicability has not been completely elucidated from a broad and realistic perspective. In the framework of learning vector-valued functions (Micchelli & Pontil, 2005) a method has been suggested based on searching the space of admissible output sequences for the minimizer of an evaluation function. For long sequences, this search may turn out to be prohibitive unless efficient algorithms are specifically designed. A related idea based on the definition of a feature mapping on the joint space of input and output sequences is presented by Altun, Tsochantaridis, and Hofmann (2003). In this case, sparseness in the problem can be exploited to design efficient algorithms. Interesting applications to problems involving sequences of moderate length have been presented.

In some application domains probabilistic graphical models (Bengio & Frasconi, 1996; Lafferty, McCallum, & Pereira, 2001; McCallum, Freitag, & Pereira, 2000) and recurrent neural networks (RNN) (see e.g. Kolen & Kremer, 2001) might offer an effective solution to sequential supervised learning problems. In particular, the bi-directional version of RNN (BRNN) has been successfully applied to various problems involving biological sequences (Baldi, Brunak, Frasconi, Soda, & Pollastri, 1999; Pollastri, Baldi, Fariselli, & Casadio, 2002b; Hawkins & Bodén, in press). The main difficulty with this class of neural networks is due to the lack of generally efficient algorithms for solving numerical optimization. In particular, error minimization is known to fail in the presence of long-range dependencies (Bengio, Simard, & Frasconi, 1994;

* Corresponding author.

0893-6080/$ - see front matter © 2005 Elsevier Ltd. All rights reserved.
doi:10.1016/j.neunet.2005.07.001

Hochreiter, Schmidhuber, Frasconi, & Bengio, 2001). Interesting remedies to vanishing gradients have been suggested in the literature (Gers, Schraudolph, & Schmidhuber, 2002; Hochreiter & Schmidhuber, 1997) but their effectiveness in realistic large scale supervised learning tasks has not been elucidated so far. The severity of this problem clearly depends on the specific application domain. For some tasks, dependencies are *mainly* local and sub-optimal solutions can be still quite useful in practice; however, full exploitation of long-range dependencies could lead to significantly better accuracy. Note that the difficulty of dealing with long-range dependencies is not necessarily inherently associated with neural networks. Other learning algorithms may equally suffer computational complexity problems.

In this paper, we start from a simple but plausible general assumption: in many problems learning efficiently in the presence of long-range dependencies is not possible because of missing information about which remote sequence positions interact. The learner is only given a set of inputs and a serial order relation on them and must solve a difficult credit assignment problem in order to identify the unknown interacting positions. On the other hand, if the learner had explicit access to information about interacting position pairs its task would be greatly simplified and it could possibly succeed. Interactions can be conveniently represented in the form of an undirected graph whose vertices are sequence positions and whose edges are interacting pairs. In order to operate in this extended setting, a learning algorithm should be able to exploit relational information other than the serial order.

We propose a neural network solution to the relational sequential supervised learning problem outlined above. We start from a class of recursive neural networks that can solve the supervised learning problem assuming that relational information is expressed in the form of a directed acyclic graph (DAG) (Frasconi, Gori, & Sperduti, 1998). These networks have been applied to a variety of supervised learning problems in which the input portion of the data is a labeled DAG while the output is a scalar (Bianucci, Micheli, Sperduti, & Starita, 2000; Sturt, Costa, Lombardo, & Frasconi, 2003). However, since sequential interaction is a symmetric relation, we need networks that operate on *undirected* graphs whose vertices are serially ordered. Similar ideas were applied in (Vullo & Frasconi, 2004) for scoring candidate graphs representing alternative disulfide connectivity patterns in proteins. In that case the output was a single real number. Here, we extend this family of networks to solve interaction enriched sequential supervised learning problems. We call the resulting architecture interaction enriched BRNN (IEBRNN).

In order to test the proposed method we study its application to protein secondary structure (SS) prediction. In this problem we are given as input the amino acid sequence (also known as primary structure) of a protein and we are interested in learning the folding regularities maintained by hydrogen bonds that are formed at a local level. These regularities are traditionally divided into three main classes: α-helices, β-strands, and coils. A supervised learning problem is then formulated as the association between an input sequence (representing the protein primary structure) and an output string that contains the SS at each residue. This is an important domain in which it is well-known that the above outlined scenario holds: dependencies are mainly local but long-range interactions might significantly help towards accuracy improvements. Several available SS prediction methods use feedforward neural networks whose input is the multiple alignment profile in a sliding window centered around the target position (Cuff & Barton, 2000; Jones, 1999; Riis & Krogh, 1996; Rost & Sander, 1993). By construction, predictions obtained with these methods are local. Long-range dependencies, on the other hand, clearly play an important role in this problem. For example, a β-sheet is composed of two or more strands that are held together by hydrogen bonds but that can be situated very far apart in the primary structure of the protein. In this case, residues that are close in space occupy distant positions in sequence. A similar effect is observed for cysteines that are linked by disulfide bonds.

The solution proposed by Baldi et al. (1999) and further developed by Pollastri, Przybylski, Rost, and Baldi (2002c) demonstrates that local dependencies can be also captured by a (bidirectional) recurrent neural network (BRNN) without any a-priori commitment about the size of the sliding window. The architecture in this case allows us to process the sequence as a whole. As a result, the output at any position in a BRNN depends on the entire input sequence and thus a BRNN might in principle exploit long-range information. Unfortunately, well-known problems of vanishing gradients (Bengio et al., 1994) do not allow us to *learn* these dependencies. In practice, even BRNNs do not clearly outperform the best predictors based on feedforward networks.

In this paper, we apply IEBRNNs in a realistic SS prediction environment. The use of this new architecture may help towards improving the present state-of-the-art and elucidate in a quantitative way the importance of long-range information in the prediction of SS. In a first set of experiments we show that high-quality interaction graphs (derived from known protein structures) lead to very significant improvements in SS prediction accuracy. Although not useful in practice for constructing a prediction tool, these results show that the IEBRNN can effectively exploit relational information and provide an empirical upper bound on the prediction accuracy that could be reached in this task by a learner that can access complete long-range information about the data. In a second set of experiments we use interaction graphs derived from predicted contact maps, showing that the proposed learning framework may be successfully exploited in a realistic setting.

The paper is organized as follows. In Section 2 we illustrate the learning problem setup and the details of IEBRNN. In Section 3 we briefly review the problem of secondary structure prediction and define our setting for learning from sequences and interaction graphs. In Section 4 we describe the data used in the experiments and in Section 5 we report and analyze our results.

2. Interaction-enriched BRNNs

2.1. Sequential supervised learning

We start from two sets x and y and we form pairs of sequences

$$(\boldsymbol{x}, \boldsymbol{y}) = (\langle x[1], x[2], \ldots, x[N]\rangle, \langle y[1], y[2], \ldots, y[M]\rangle)$$

with $x[t] \in x$, $t = 1, \ldots, N$, $y[s] \in y$, $s = 1, \ldots, M$, by sampling from a fixed and unknown distribution $p(\boldsymbol{x}, \boldsymbol{y})$. For example, in the application to protein secondary structure prediction described in this paper, sequences of real vectors are mapped into sequences of integers as in multiclass classification, i.e. $\mathcal{X} = \mathbb{R}^n$ and $y = \{1, 2, \ldots, Q\}$.

In standard supervised sequence learning we are given a dataset of i.i.d. input–output pairs $D_m = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^m$ where each $\boldsymbol{x_i}$ is an input sequence of length N_i and $\boldsymbol{y_i}$ is the corresponding target output sequence of length M_i. The objective is to learn a function f that maps an input sequence $\boldsymbol{x}$ into a corresponding output sequence $f(\boldsymbol{x})$ in a way that approximates well the probabilistic relation between $\boldsymbol{x}$ and $\boldsymbol{y}$. In the following, we assume that sequence lengths are preserved, i.e. $N_i = M_i$ for all input–output pairs and we shall denote by $f(\boldsymbol{x_i})[t]$ the t-th element of $f(\boldsymbol{x_i})$ for $t = 1, \ldots, N_i$.

2.2. Interaction enriched sequential supervised learning

In the extended setting inputs are enriched with the interaction relations. In particular, the input portion of the data is now described by the pair $(\boldsymbol{x}, G)$ where $\boldsymbol{x}$ is an input sequence as before and $G = (V, E)$ is an undirected graph whose vertex set is $V = \{1, 2, \ldots, N\}$ and whose edges represent interactions, i.e. $\{t, s\} \in E$ if and only if positions t and s interact. In this case we are interested in learning a function f that can exploit both sources of information: the sequence $\boldsymbol{x}$ and the additional relational information consisting of the interaction graph G. Output sequences are then approximated as $f(\boldsymbol{x}, G)$.

Note that the input object $(\boldsymbol{x}, G)$ can also be interpreted as a labeled undirected graph. In particular, $x[t]$ can be regarded as the label attached to vertex t (see Fig. 1).

Edge labels (i.e. attributes associated with the interactions) can also be accommodated in this framework. They could be useful for example to model interaction strengths. It is possible to see that under the assumption of bounded connectivity and thanks to the total order on vertices, the label of an edge can be always moved into each of the two adjacent vertices without losing any information. For example, in the case of real-valued labels, assuming each sequence position interacts with at most K other positions, labeled edges can be simply dealt with by extending the input set from $\mathcal{X}$ to $\mathcal{X}' = \mathcal{X} \times \mathbb{R}^K$.

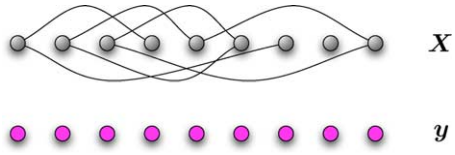

Fig. 1. Input and output portion of the data.

2.3. Architecture

The architecture presented here is an extension of the recursive neural networks (RNN) described by Frasconi et al. (1998) for directed acyclic graphs and the bidirectional recurrent neural network (BRNN) used by Baldi et al. (1999) for non-causal processing of protein sequences. In all these architectures memories are realized with (hidden) state variables and the input–output mapping is obtained as the composition of a state transition function and an output function. RNNs can solve IO-isomorph structural transduction problems, i.e. mapping a labeled input graph $\boldsymbol{x}$ into a corresponding output graph $\boldsymbol{y}$ having the same topology as $\boldsymbol{x}$. Intuitively, an RNN maps the label of each input into a corresponding label of the associated output vertex and the mapping depends on contextual information found in other vertices. Edges in RNNs are interpreted as *causal* dependencies and therefore structures are restricted to directed and acyclic graphs. Causality in this context means that prediction at a given vertex v, denoted as $f(\boldsymbol{x})[v]$, only depend on the input labels found on vertices of $\boldsymbol{x}$ that can be reached from v. BRNNs, on the other hand, are based on a factorization of the state space and use two transition functions that process the input sequence in both directions. BRNNs do not make any causality assumption in the data[1] but cannot deal with graphs. Like in BRNNs, the solution suggested here uses state space factorization.

To simplify notation, in the following we omit the integer subscripts that index training examples and we focus on a single input–output pair $((\boldsymbol{x}, G), \boldsymbol{y})$. For each position t, we introduce two real vectors $\boldsymbol{\varphi}(\boldsymbol{x})[t] \in \mathbb{R}^d$ and $\boldsymbol{\beta}(\boldsymbol{x})[t] \in \mathbb{R}^d$ that we call the *forward* state and the *backward* state, respectively. For simplicity, the dependency on $\boldsymbol{x}$ will be also omitted from the notation when clear from the context.

Intuitively, the forward and the backward states at t are compressed vector representations of the 'past' substring

[1] This makes sense, for example, in the case of protein data: it is not true that SS at position t depends only on amino acids situated *before* t or only on those situated *after* t.

$x[1],\ldots,x[t]$ and the 'future' substring $x[t],\ldots,x[N]$, respectively. These two vectors are expected to contain all the information about the input sequence that is needed to make a prediction at position t. A standard BRNN can be interpreted as the combination of two discrete-time dynamical systems defined by two recursive update equations in which $\boldsymbol{\phi}[t]$ explicitly depends on $\boldsymbol{\phi}[t-1]$ and $\boldsymbol{\beta}[t]$ on $\boldsymbol{\beta}[t+1]$. In an IEBRNN we introduce additional 'shortcut' dependencies so that states $\boldsymbol{\phi}[t]$ and $\boldsymbol{\beta}[t]$ depend explicitly also on states at interacting positions, as specified by G.

In order to construct the generalized dynamics for an IEBRNN we begin by forming two directed graphs from G as follows. Let

$$E_{\mathrm{F}} \doteq \{(s,t) : \{s,t\} \in E, s < t\} \tag{1}$$

$$E_{\mathrm{B}} \doteq \{(s,t) : \{s,t\} \in E, s > t\}. \tag{2}$$

We now have a predecessor graph $G_{\mathrm{F}}=(V, E_{\mathrm{F}})$ with forward oriented edges and a successor graph $G_{\mathrm{B}}=(V, E_{\mathrm{F}})$ with backward oriented edges. It is immediate to see that G_F and G_{B} are acyclic. Moreover, for each given vertex t, the set $\{(t, s): s\in V\}$ of edges incident on t can be sorted in increasing order of s. Also, we assume that all graphs have degree smaller than a fixed constant K. Define

$$\ell_{t,j} = \begin{cases} j\text{th vertex in the sorted adjacency list of } t \text{ in } G_{\mathrm{F}} & \text{if } t \text{ has at least } j \text{ parents in } G_{\mathrm{F_C}} \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

and

$$r_{t,j} = \begin{cases} j\text{th vertex in the sorted adjacency list of } t \text{ in } G_B & \text{if } t \text{ has at least } j \text{ parents in } G_B \\ N+1 & \text{otherwise} \end{cases} \tag{4}$$

The IEBRNN is then based on the following non-causal dynamics:

$$\varphi[t] = \mathcal{T}_{\mathrm{F}}(x[t], \varphi[t-1], \varphi[\ell_{t,1}], \varphi[\ell_{t,2}], \ldots, \varphi[\ell_{t,K}]; \boldsymbol{\theta}_{\mathrm{F}}) \tag{5}$$

$$\boldsymbol{\beta}[t] = \mathcal{T}_{\mathrm{B}}(x[t], \boldsymbol{\beta}[t+1], \boldsymbol{\beta}[r_{t,1}], \boldsymbol{\beta}[r_{t,2}], \ldots, \boldsymbol{\beta}[r_{t,K}]; \boldsymbol{\theta}_{\mathrm{B}}) \tag{6}$$

with boundary conditions

$$\varphi[0] = \beta[N+1] = 0. \tag{7}$$

In the above recursions, $\mathcal{T}_{\mathrm{F}}$ and $\mathcal{T}_{\mathrm{B}}$ are the forward and backward state transition function, respectively. They are parametric functions with adjustable parameters $\boldsymbol{\theta}_{\mathrm{F}}$ and $\boldsymbol{\theta}_{\mathrm{B}}$ that will be determined by learning. We assume here that they are realized by feedforward neural networks with $n+(K+1)d$ inputs and d sigmoidal outputs (with no internal hidden layer) as shown in Fig. 3.

Outputs are then computed as follows:

$$f(\boldsymbol{x})[t] = \eta(\varphi(\boldsymbol{x})[t], \boldsymbol{\beta}(\boldsymbol{x})[t]; \boldsymbol{\theta}_Y); \tag{8}$$

where η is also a parametric function with adjustable parameters $\boldsymbol{\theta}_{\boldsymbol{Y}}$. Sequential translation with discrete outputs assigned to each position is similar to multiclass classification. Assuming Q classes, we define

$$a_q[t] = \langle [\varphi[t]\boldsymbol{\beta}[t]]', \theta_{Y,q} \rangle \quad q = 1, \ldots, Q \tag{9}$$

where $\boldsymbol{\theta}_{\mathrm{Y},q} \in \mathbb{R}^{2d}$ is a parameter vector and $\langle \cdot, \cdot \rangle$ denotes the standard dot product. We then compute

$$f_q[t] = \frac{\mathrm{e}^{a_q[t]}}{\sum_{r=1}^{Q} \mathrm{e}^{a_r[t]}}. \tag{10}$$

and

$$f(\boldsymbol{x})[t] = \arg\max_q f_q[t]$$

The use of normalized exponentials allows us to interpret $f_{\boldsymbol{q}}[t]$ as the conditional probability $\Pr(Y[t]=q|\boldsymbol{x})$ that the class at position t is q, given the input $\boldsymbol{x}$.

The computations described by Eqs. (5), (6), and (8) can be graphically described as shown in Fig. 2. Nodes in the diagram represent input, state, and output vectors at different sequence positions. Arcs represent arguments to transition and output functions. Dotted arcs represent the first argument of functions $\mathcal{T}_{\mathrm{F}}$, $\mathcal{T}_{\mathrm{B}}$. Solid arcs going left-to-right and right-to-left represent the second argument of functions $\mathcal{T}_{\mathrm{F}}$, $\mathcal{T}_{\mathrm{B}}$ and are also found in the computation of standard BRNNs. Thin arcs are shortcut inherited from the interaction graphs associated with the input sequence. Finally, dashed arcs represent the arguments of the output function η.

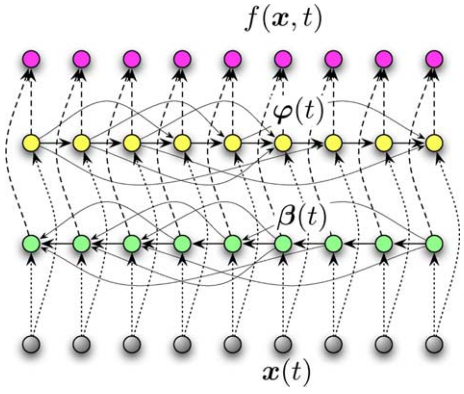

Fig. 2. Graphical representation of the message passing in the IEBRNN architecture.

2.4. Learning

Discriminant training is carried out following the maximum likelihood approach: a likelihood function of the parameters and the training set is obtained thanks to the probabilistic interpretation of the outputs. Assuming i.i.d sequences and denoting by $\boldsymbol{\theta}$ the whole set of parameters we have

$$\ell(D_m|\boldsymbol{\theta}) \doteq \log p(D_m|\boldsymbol{\theta}) \propto \sum_{i=1}^{m}\sum_{t=1}^{N_i} \log p(y_i[t]|\boldsymbol{x}_i, \boldsymbol{\theta}) \tag{11}$$

that can be rewritten as

$$\ell(D_m|\theta) = \sum_{i=1}^{m}\sum_{t=1}^{N_i}\sum_{q=1}^{Q} z_{i,q}[t]\log f_q(\boldsymbol{x}_i;\boldsymbol{\theta})[t]. \tag{12}$$

where $z_{i,q}[t]=1$ if $y_i[t]=q$ and 0 otherwise.

Gradient computation for likelihood optimization can be carried out analytically using backpropagation on the unfolded architecture (see Fig. 2) and taking into account the fact that parameters of the transition functions and output function are shared across different sequence positions. Additional details on gradient calculation in this network given in Appendix A.

3. Prediction of protein secondary structure

3.1. The folding problem

A protein chain is a polymer formed by amino acids linked by peptide bonds. Each amino acid in a chain is also called a *residue*. Since there are 20 amino acids in nature, the sequence can be conveniently represented by a string in a 20-letter alphabet. The biological function of a protein depends on its fold or *tertiary* structure (i.e. the coordinates of all its atoms). The experimental determination of structures is a costly process that is presently carried out either by X-ray crystallography or by nuclear magnetic resonance methods. As a consequence of past and ongoing genome sequencing projects, at the time of writing roughly 2 million non-redundant protein sequences are known, compared to roughly 30,000 resolved 3D structures deposited in the Protein Data Bank (PDB) (Berman et al., 2000)—a 60:1 ratio, that continues to grow. According to Anfinsen's thermodynamic hypothesis (Anfinsen, 1973), a protein's native fold is the one having lowest free energy and thus it depends on the sequence and the external environment (Berg, Tymoczko, Stryer, & Clarke, 2002). Unfortunately, computing and minimizing the potential function from first principles is not feasible for present computers and therefore predictive methods are a very important approach in order to obtain an approximation of the 3D structures associated with known sequences.

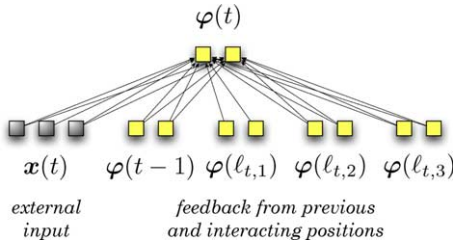

Fig. 3. Realization of state transition function $\mathcal{T}_{\mathrm{F}}$ by a feedforward network. Here $n=3$, $d=2$ and $K=3$-see Eq. (5).

In addition, prediction approaches may be useful in cases where the lowest energy state cannot be reached (Fig. 3).

Prediction methods are divided into three main classes: comparative modeling (that can be applied if the target protein has sufficient sequence similarity with existing structures), fold recognition/threading (that requires similarity at the level of fold category), and ab initio (for new folds). Machine learning is a promising approach in the last case (Baldi & Brunak, 2001). While comparative modeling methods are currently fairly accurate, especially when sequence similarity to existing structures is above 50%, fold-recognition/threading and especially ab initio methods are rarely so (Baker & Sali, 2001). For instance, the best ab initio methods are capable of satisfactorily reconstructing stretches of 80–100 residues (about a third of the average protein) in less than 20% of the cases (Bradley et al., 2003).

3.2. State-of-the-art in SS prediction

Predictors based on neural networks have been pioneered by Qian and Sejnowski (1988) and subsequently refined during the 1990s. Rost and Sander (1993) have incorporated evolutionary information in the form of multiple alignment profiles, a technique that significantly boosted prediction accuracy. For each position in a protein sequence, the multiple alignment profile consists of the frequencies of every possible amino-acid as calculated from the multiple alignment in this position.

Subsequently, Riis and Krogh (1996) have introduced a number of architectural improvements, including the use of adaptive encoding of amino acids and the use of a structure-to-structure network trained as a postprocessor to filter out local prediction errors. Jones (1999) suggested the use of position specific matrices for incorporating evolutionary information, and obtained favorable results in terms of prediction accuracy. Predictors based on HMMs (Karplus, Barrett, & Hughey, 1998) and bidirectional RNNs (Baldi et al., 1999; Pollastri & McLysaght, in press; Pollastri et al., 2002c) have also been introduced, leading to state-of-the-art results, but no radical breakthroughs.

A widely used measure of performance is Q_3, defined as the fraction of residues whose SS is assigned correctly by the predictor. The best currently available methods report a Q_3 level between 75 and 79% in the three-state SS prediction. A complementary measure that quantifies the

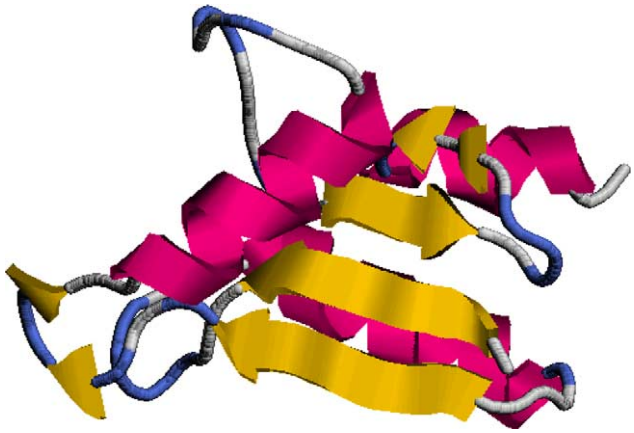

Fig. 4. 3D structure of Glutaredoxin (PDB code 1ABA).

capability of the classifier to correctly predict entire segments of SS is segment overlap (SOV)—see (Zemla, Venclovas, Fidelis, & Rost, 1999) for details.

3.3. Prediction of SS from sequence and interaction graphs

In the case of SS prediction, a reasonable source of information about long-range interaction can be obtained from contact maps, a graphical representation of the spatial neighborhood relation among amino acids. An edge $\{t, s\}$ in a contact map indicates that the distance between the C_α atoms of the residues at positions t and s is lower than a predefined threshold (see Figs. 4 and 5).

Contact maps at a coarser (e.g. secondary structure segment (Baldi & Pollastri, 2003)) or finer (e.g. atomic) level are also possible. In this work we choose residue contact maps because they provide an accurate enough picture to yield correct 3D structures (Vendruscolo, Kussel, & Domany, 1997), yet remain computationally tractable.

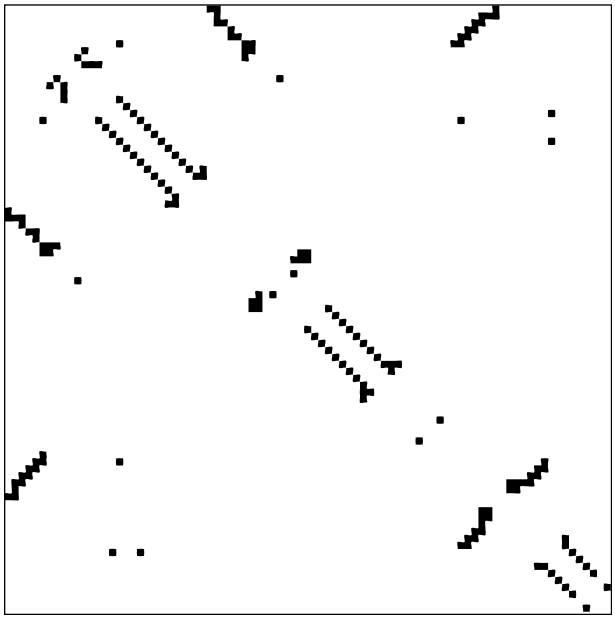

Fig. 5. Contact map of Glutaredoxin (PDB code 1ABA). Contact resolution in this example is 6 Å. Contacts between residues that are closer than 3 positions in sequences are omitted.

There are various reasons why using contact maps information in order to predict SS is worth investigation:

Algorithms that reconstruct structures from contact maps are based on the definition of a potential energy function whose global minimization is not straightforward and requires stochastic optimization techniques to escape local minima (Vendruscolo et al., 1997). Thus it is not clear that a supervised learning algorithm can actually *learn* to recover SS from contact maps.

Contact maps can be predicted from sequence (Fariselli, Olmea, Valencia, & Casadio, 2001; Pollastri & Baldi, 2002; Pollastri, Baldi, Vullo, & Frasconi, 2003) or can be obtained from structures predicted by ab initio methods such as Rosetta (Baker & Sali, 2001). Although accuracy of present methods is not yet sufficient to provide a satisfactory solution to the folding problem, predicted maps may still contain useful information to improve the prediction of lower order properties such as the SS.

Even if contact maps are given, the design of a learning algorithm that can fully exploit their information content is not straightforward. For example, Meiler and Baker (2003) have shown that SS prediction can be improved by using information about inter-residue distances. Their architecture is a feedforward network fed by *average* property profiles associated with amino acids that are near in space to the target position. In this way, relative ordering among neighbors in the contact map is lost.

4. Data used in the experiments

4.1. Protein data

The experiments have been performed using a representative set of non-homologous chains from the Protein Data Bank (PDB Select (Hobohm & Sander, 1994)). We extracted the sequences from the December 2002 release, listing 1,950 chains with a percentage of homology lower than 25%. From this set we retained only high quality proteins on which the DSSP program does not crash, determined only by X-ray diffraction, without any physical chain breaks and resolution threshold lower than 2.5 Å. The final dataset contained 811 chains (137,926 residues) split in a training set R of 374 chains (68,074 residues), a validation set V of 197 chains (35,181 residues) and a test set T of 240 chains (34,671 residues).

In all the experiments, the input sequences consisted of profiles derived from multiple alignments. Multiple alignments were generated using PSI-BLAST (Altschul et al., 1997) applied to the Swiss-Prot+TrEMBL non-redundant database (Bairoch & Apweiler, 1996). Each sequence

position t contained a real vector $x[t] \in \mathbb{R}^{20}$ whose ith component was the frequency of occurrence for the ith amino acid in the tth column of the alignment. SS labels were assigned using the DSSP program (Kabsch & Sander, 1983). In case of ambiguities in the PDB files, the coordinates of the C_α atoms used to calculate the contact map were selected according to the same strategy used by DSSP. We reduced DSSP's eight classes into the three main classes by mapping H into α (helices), E into β (strands), and the rest (B, C, G, I, S, T) into γ (coils). Contacts were defined using a threshold of 6 Å, a value, which comprises all the hydrogen bonded pairs and few side-chain contacts. Amino-acids spaced less than 3 positions in the sequence were not considered, because their C_α atoms are always at a distance lower than 6 Å.

4.2. Interaction graphs

The second set of experiments reported below aims to estimate the generalization of the system in SS prediction from sequence and *predicted* interactions. These predicted interactions are obtained from a predictor of contacts at 6 Å from the CMAPpro suite (Baldi & Pollastri, 2003; Pollastri & Baldi, 2002). This predictor is based on an ensemble of 3 Recursive Neural Networks trained on a non-redundant set of protein contact maps. The RNNs are given as inputs the protein sequence, evolutionary information obtained from multiple sequence alignments, and predicted secondary structure and relative solvent accessibility. Solvent accessibility predictions are obtained from the ACCpro program (Pollastri, Baldi, Fariselli, & Casadio, 2002a), while secondary structure predictions are derived from the baseline BRNN predictor described in Section 5.1. ACCpro and CMAPpro were originally trained using different sets of chains (Baldi & Pollastri, 2003; Pollastri & Baldi, 2002; Pollastri et al., 2002a) whose union consisted of 1432 sequences. In order to guarantee rigorous independence between train and test data in the present experiments, the previously defined set T has been chosen so that no chain had significant remote homology (calculated doing a BLAST search with e-value of 10^{-2}) with the 1432 sequences. By doing so we ensure that none of the stages contributing to the prediction of contact interactions was trained on sequences homologous to those in the test set T.

5. Prediction of secondary structure from sequence and contact maps

5.1. Prediction from sequence alone

A first set of experiments was performed to obtain baseline prediction accuracy for the SS problem on this dataset. In this paper we are interested in highlighting the contributions of long-range information to the prediction of SS rather than in obtaining state-of-the-art performances from the classifier. For this reason a 'bare bones' BRNN classifier with multiple alignment profiles as input was used for this purpose. All common tricks that can be used to boost accuracy (in particular large training sets, shortcuts connections between non-consecutive states, cascaded predictors, computation of outputs at each position in the sequence using a window of states, and ensembles of independent classifiers, as in (Baldi et al., 1999; Pollastri & McLysaght, in press; Pollastri et al., 2002c) were omitted.

The network size, together with the parameters of the learning algorithm, were chosen using the validation set: an architecture with $d = 20$ neurons for each recursive state was then selected, the learning rate was set fixed to 1×10^{-4} and an early stopping procedure was used to avoid over-fitting. The same parameters and sizes were used in all the experiments reported here and in the following. The validation set was also used for the early-stopping procedure.

Results of the baseline experiment are reported in the upper-left corner of Table 1. The corresponding confusion matrix is shown in the upper-left corner of Table 2. In this matrix, entry at row i and column j is the number of residues that are assigned to class i and belong to class j, divided by the number of residues assigned to class i. Precision for each class is marked in boldface and recall is reported separately in the last row.

5.2. Prediction from contact maps alone

Here, we want to estimate the amount of information about the SS the IEBRNN architecture is capable to learn from contacts alone. Therefore, in this experiment null inputs were used, i.e. $x(t) = 0$ for each position t. Information on contacts was inserted in the form of an interaction graph, as explained in Section 2.

Table 1
Prediction accuracy (Q_3) and segment overlap (SOV) along with 95% confidence intervals. Interaction graphs are obtained in this case from true protein structures

BRNN			IEBRNN		
Profiles only (baseline)	Q_3	74.6 ± 0.4%	Interactions only	Q_3	79.9 ± 0.4%
	SOV	66.7 ± 1.8%		SOV	73.3 ± 1.6%
Profiles + context	Q_3	82.5 ± 0.4%	Profiles + interactions	Q_3	84.6 ± 0.3%
	SOV	77.4 ± 1.5%		SOV	79.3 ± 1.5%
Profiles + context (no γ)	Q_3	95.9 ± 0.2%	Profiles + interactions (no γ)	Q_3	97.9 ± 0.1%
	SOV	94.6 ± 1.0%		SOV	95.5 ± 0.8%

Table 2
Confusion matrices for the different methods described in the paper. Interaction graphs are obtained from true protein structures

BRNN				IEBRNN		
Profiles only (baseline)				Interactions only		
	α (%)	β (%)	γ (%)	α (%)	β (%)	γ (%)
α	**77.8**	5.8	16.4	**83.8**	0.8	15.5
β	8.2	**70.9**	20.9	1.1	**76.5**	22.4
γ	11.6	14.5	**74.0**	9.8	11.4	**78.8**
Recall	77.4	58.9	80.2	85.2	75.3	78.4
Profiles + context				Profiles + interactions		
	α (%)	β (%)	γ (%)	α (%)	β (%)	γ (%)
α	**85.1**	2.6	12.2	**88.6**	0.7	10.7
β	4.5	**76.0**	19.5	1.2	**81.9**	16.9
γ	6.7	9.6	**83.7**	8.0	8.9	**83.1**
Recall	87.6	76.5	81.8	87.6	80.1	84.6
Profiles + context (no γ)				Profiles + interactions (no γ)		
	α (%)	β (%)	γ (%)	α (%)	β (%)	γ (%)
α	**93.4**	6.5	0.1	**98.3**	1.6	0.1
β	6.4	**93.4**	0.1	2.4	**97.5**	0.1
γ	0.2	0.9	**98.9**	1.2	0.9	**97.9**
Recall	95.5	88.3	99.9	96.6	95.8	99.9

As reported in Table 1, we obtained $Q_3 \approx 80\%$ and SOV $\approx 73\%$. These results are interesting considered that the classifier has no knowledge about the three-dimensional conformation of the hydrogen bonded atoms and the physics behind the formation of SS.

Comparing the predictions of this classifier and those of the classifier described above in Section 5.1 we found that they overlap only for the 69% of the residues in the test set. Moreover, 92% of the times at least one of the two classifiers makes the correct prediction. It is then arguable that BRNNs trained on profile sequences and IEBRNNs trained on contact maps alone capture rather different regularities in the data. This suggests that training the IEBRNN with both inputs should allow us to obtain improved accuracy.

5.3. Prediction from profiles and contacts together

In this experiment, we trained the IEBRNN with both profiles and contacts as input.

For sake of comparison with the results of Meiler and Baker (2003) we also trained a standard BRNN with the same kind of inputs they used. In particular, the spatial *context* of each residue was computed by averaging the profile of the amino-acids in a sphere of 6 Å centered on the residue itself. This additional input was then given to the standard BRNN together with the usual profile. This method can be seen as a simplified version of the IEBRNN in which all contacts give the same contribution to a given position. In particular, order among contacts cannot be distinguished.

As can be seen from Table 1, the information about contact ordering is efficiently exploited by the IEBRNN, which appreciably outperforms the solution based on the average context. Interestingly, prediction accuracy improves for helices and strands but not for coils (see confusion matrices in Table 2).

5.4. Effects of interaction robustness

The results of the above experiments show us that IEBRNNs can effectively exploit the information contained in the contact map to improve prediction accuracy. However, there is still about 15% residual error rate that would be interesting to explain. We conjecture that the reliability of the interactions that were injected as an additional input may play a significant role. In facts, edges in a contact map express spatial proximity but do not necessarily imply dependencies between the two close residues. This may be particularly true in the case of contacts that involve coil residues. Instead, contacts between residues that both belong to helices or strands can be expected to encode interactions in a more robust way since they are often maintained by hydrogen bonds.

In order to evaluate the effect of interaction robustness we repeated our experiments using more sparse contact maps where edges only connect residues that belong to helices or strands. In so doing, we removed about 60% of the edges from interaction graphs. Results are reported in the last row of Table 1 both for the standard BRNN (fed by profiles and context) and for the IEBRNN. The error reduction obtained in this way is dramatic. The residual error is comparable to the disagreement between different SS assignment programs (i.e. DSSP, STRIDE, and DEFINE). These experiments could indicate that part of the information contained in the contact map can be misleading, especially for the shorter segments.

5.5. Integration with contact map predictor

We performed a final set of experiments to estimate the accuracy of the SS predictor on a realistic prediction setting in which only the primary structure is known. In this case, the interaction graphs were derived from the contact map

Table 3
Prediction accuracy (Q_3) and segment overlap (SOV) along with 95% confidence intervals

IEBRNN					
Profiles + predicted interactions	Q_3	76.0 ± 0.5%	Profiles + near predicted interactions	Q_3	75.7 ± 0.5%
	SOV	71.0 ± 1.8%		SOV	71.5 ± 1.8%
Profiles + true interactions	Q_3	77.4 ± 0.5%	Profiles + near true interactions	Q_3	79.4 ± 0.4%
	SOV	72.6 ± 1.7%		SOV	73.6 ± 1.6%

Networks are trained on interaction graphs obtained from predicted contact maps. Trained networks are tested both on predicted and true interaction graphs. Results in the right column were obtained by keeping only local interactions (distance between 3 and 7 positions).

predicted by the system described in Section 4. The contact map predictor was fed with the secondary structure information given by the BRNN architecture with profiles only as input described in Section 5.1. Therefore, the composition of BRNN, contact map predictor, and IEBRNN can be thought of as a single secondary structure predictor working on primary structure only as input. The data used to test the IEBRNN are independent from those used to train the BRNN and the contact map predictor, to avoid overestimation of the generalization performances of the IEBRNN.

Results are presented in Tables 3 and 4 and should be compared to the corresponding values in Tables 1 and 2. As expected, the accuracy obtained from sequences and predicted interaction graphs degrades compared to the results obtained when using true contact maps. Nonetheless, compared to prediction from sequence alone a statistically significant increase can be observed.

In Tables 3 and 4 we also present the value of accuracy of the same networks trained on predicted interaction graphs when tested using true contact maps. Results demonstrate that the IEBRNN can effectively exploit even uncertain contacts information during training and could perform even better if the quality of predicted contact maps improved.

Finally, we carried out a set of experiments in which distant contacts were removed. The right columns of Tables 3 and 4 report accuracies and confusion matrices obtained by keeping only predicted near interactions (distance in sequence between 3 and 7 positions). Unexpectedly, the performances increased, probably due to a higher confidence of the network on near contacts, which usually are more accurately predicted.

6. Conclusions

We have introduced IEBRNN as a method for simplifying sequence learning tasks with neural networks by incorporating explicit knowledge about interactions between sequence elements. Empirical results show that this approach can greatly improve prediction of protein secondary structure when interaction knowledge is available. In addition, in this problem IEBRNNs are able to capture the serial order relation within interactions, a property that leads to better predictions than simply averaging the input at interacting positions in a sequence, as proposed in Meiler and Baker (2003).

The method described here does not yet advance the state-of-the-art in SS prediction but enlightens possible future directions of investigation. Our findings provide further evidence that the knowledge of a relatively small number of reliable interactions (such as contacts between residues in helices and strands) is sufficient with the currently available data to obtain high quality predictions. In order to advance this study we are investigating

Table 4
Confusion matrices obtained from networks trained on predicted interaction graphs

IEBRNN						
Profiles + predicted interactions				Profiles + near pred. interactions		
	α (%)	β (%)	γ (%)	α (%)	β (%)	γ (%)
α	**80.9**	5.5	13.6	**80.2**	6.2	13.6
β	6.3	**76.5**	7.2	7.5	**74.3**	18.2
γ	11.5	15.6	**72.9**	10.9	15.7	**73.4**
Recall	77.8	57.6	84.1	78.2	57.0	83.5
Profiles + true interactions				Profiles + near true interactions.		
	α (%)	β (%)	γ (%)	α (%)	β (%)	γ (%)
α	**80.5**	5.5	14.0	**85.8**	1.4	12.8
β	4.6	**78.7**	16.7	3.4	**78.5**	18.1
γ	9.9	15.0	**75.1**	7.7	16.7	**76.6**
Recall	81.7	59.7	83.5	86.0	61.9	83.7

Trained networks are tested both on predicted and true interaction graphs. Results in the matrices on the right were obtained by keeping only local interactions (distance between 3 and 7 positions).

improvements on the IEBRNN architecture to match the characteristics and performances of the latest more complex SS predictors. A closed loop procedure, which iterates the SS/Contact-Map prediction process, will also be tested.

Acknowledgements

We thank Andrea Passerini and Alessandro Vullo for useful discussions. The work of AC and PF was partially supported by the Italian Department of Education, University, and Research (MIUR) under grants no. 2002093941 and 2003091149_002 and by EU NoE BIOPATTERN (project no. 508803). The work of GP is supported by Science Foundation Ireland grants 04/BR/CS0353 and 05/RFP/CMS0029, and a UCD President's Award 2004.

Appendix A

We highlight here the gradient computation in the IEBRNN. For simplicity we only focus on the forward states $\varphi[t] \in \mathbb{R}^d$ (see Eq. (5)) since the computation for backward states (Eq. (6)) follows a similar scheme. Also, to avoid a cumbersome notation we just show calculations on a single sequence. For a generic component $c=1,\ldots,d$ of the forward state vector let

$$\alpha_c[t] = \sum_{j=1}^{n} \theta^{\text{in}}_{F,c,j} x_j[t] + \sum_{j=1}^{d} \theta^{\text{so}}_{F,c,j} \varphi_j[t-1] + \sum_{k=1}^{K} \sum_{j=1}^{d} \theta^{\text{ie}}_{F,c,j} \varphi_j[\ell_{t,k}] \quad \text{(A.1)}$$

where $\theta^{\text{in}}_{F,c,j}$ are weights connecting input units to forward state units, $\theta^{\text{so}}_{F,c,j}$ are weights connecting previous to present states (serial order), and $\theta^{\text{ie}}_{F,c,j}$ are weights associated with connections defined by interacting states (the F subscript stands for 'forward' weights). Then forward state components are obtained as

$$\varphi_c[t] = \sigma(\alpha_c[t])$$

where σ is a sigmoidal function, e.g. $\sigma(\alpha) = \tan h(\alpha)$. Define

$$\delta_c[t] \doteq \frac{\partial \log p(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{\theta})}{\partial \alpha_c[t]}$$

where $p(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{\theta})$ is the likelihood of the generic sequence in the data set given the entire vector of network weights $\boldsymbol{\theta}$. Then we have

$$\delta_c[t] = \sigma'(\alpha_c[t])\Big(\sum_{l=1}^{d} \theta^{\text{so}}_{F,l,c} \delta_l[t+1] + \sum_{\tau,h:\ell_{\tau,h}=t} \sum_{l=1}^{d} \theta^{\text{ie}}_{F,l,c} \delta_l[\tau] + \sum_{q=1}^{Q} \theta_{Y,q,c}(z_q[t] - f_q[t])\Big)$$

where $\boldsymbol{\theta}_{Y,q} = [\theta_{Y,q,1}, \ldots, \theta_{Y,q,d}]$, $z_q[t] = 1$ if $y[t] = q$ and $z_q[t] = 0$ otherwise. Gradients for the weight connecting two units are finally computed by multiplying the δ of the receiving unit by the activation of the sending unit. Thus for example

$$\frac{\partial \log p(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{\theta})}{\partial \theta^{\text{in}}_{F,c,j}} = \sum_t \delta_c[t] x_j[t]$$

where the summation over sequence indices t is needed to take into account weight sharing.

References

Altschul, S. F., Madden, T. L., Schaffer, A. A., Zhang, J., Zhang, Z., Miller, W., et al. (1997). Gapped BLAST and PSI-BLAST: A new generation of protein database search programs. *Nucleic Acids Research*, *25*, 3389–3402.

Altun, Y., Tsochantaridis, I., & Hofmann, T. (2003). Hidden markov support vector machines. In *Proceedings of international conference on machine learning*.

Anfinsen, C. (1973). Principles that govern the folding of protein chains. *Science*, *181*, 223–230.

Bairoch, A., & Apweiler, R. (1996). The Swiss-Prot protein sequence data bank and its new supplement TrEMBL. *Nucleic Acids Research*, *24*, 21–25.

Baker, D., & Sali, A. (2001). Protein structure prediction and structural genomics. *Science*, *294*, 93–96.

Baldi, P., & Brunak, S. (2001). *Bioinformatics: The machine learning approach* (2nd ed.). Cambridge, MA: MIT Press.

Baldi, P., Brunak, S., Frasconi, P., Soda, G., & Pollastri, G. (1999). Exploiting the past and the future in protein secondary structure predicton. *Bioinformatics*, *15*, 937–946.

Baldi, P., & Pollastri, G. (2003). The principled design of large-scale recursive neural network architectures—dag-rnns and the protein structure prediction problem. *Journal of Machine Learning Research*, *4*(Sep), 575–602.

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

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

Berg, J., Tymoczko, J., Stryer, L., & Clarke, N. (2002). *Biochemistry* (5th ed.). San Francisco, CA: W.H. Freeman & Co..

Berman, H., Westbrook, J., Feng, Z., Gilliland, G., Bhat, T., Weissig, H., et al. (2000). The protein data bank. *Nucleic Acids Research*, *28*, 235–242.

Bianucci, A., Micheli, A., Sperduti, A., & Starita, A. (2000). Application of cascade correlation networks for structures to chemistry. *Applied Intelligence*, *12*, 117–146.

Bradley, P., Chivian, D., Meiler, J., Misura, K., Rohl, C., Schief, W., et al. (2003). Rosetta predictions in CASP5: Successes, failures, and prospects for complete automation. *Proteins*, *53*(S6), 457–468.

Cuff, J. A., & Barton, G. J. (2000). Application of multiple sequence alignment profiles to improve protein secondary structure prediction. *Proteins*, *40*, 502–511.

Dietterich, T. G. (2002). Machine learning for sequential data: A review. *Lecture Notes in Computer Science*, *2396*, 15–31.

Fariselli, P., Olmea, O., Valencia, A., & Casadio, R. (2001). Prediction of contact maps with neural networks and correlated mutations. *Protein Engineering*, *14*, 835–843.

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

Gers, F., Schraudolph, N., & Schmidhuber, J. (2002). Learning precise timing with LSTM recurrent networks. *Journal of Machine Learning Research*, *3*, 115–143.

Hawkins, J., & Bodén, M. (in press). The applicability of recurrent neural networks for biological sequence analysis. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*.

Hobohm, U., & Sander, C. (1994). Enlarged representative set of protein structures. *Protein Science*, *3*, 522–524.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, *9*, 1681–1726.

Hochreiter, S., Schmidhuber, J., Frasconi, P., & Bengio, Y. (2001). Gradient flow in recurrent nets: The difficulty of learning long-term dependencies. In J. Kolen, & S. C. Kremer (Eds.), *A field guide to dynamical recurrent networks*. New York: Wiley–IEEE Press.

Jones, D. (1999). Protein secondary structure prediction based on position-specific scoring matrices. *Journal of Molecular Biology*, *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.

Karplus, K., Barrett, C., & Hughey, R. (1998). Hidden markov models for detecting remote protein homologies. *Bioinformatics*, *14*, 846–856.

Kolen, J., & Kremer, S. C. (Eds.). (2001). *A field guide to dynamical recurrent networks*. New York: Wiley–IEEE Press.

Lafferty, J., McCallum, A., & Pereira, F. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of 18th international conference on machine learning* (pp. 282–289). San Francisco, CA: Morgan Kaufmann.

McCallum, A., Freitag, D., & Pereira, P. (2000). Maximum entropy Markov models for information extraction and segmentation. In *Proceedings of 17th international conference on machine learning* (pp. 591–598). San Francisco, CA: Morgan Kaufmann.

Meiler, J., & Baker, D. (2003). Coupled prediction of protein secondary and tertiary structure. *Proceedings of the National Academy of Sciences of the United States of America*, *100*(2), 12105–12110.

Micchelli, C., & Pontil, M. (2005). On learning vector-valued functions. *Neural Computation*, *17*, 177–204.

Pollastri, G., & Baldi, P. (2002). Prediction of contact maps by giohmms and recurrent neural networks using lateral propagation from all four cardinal corners. *Bioinformatics*, *18*(Suppl. 1), S62–S70.

Pollastri, G., Baldi, P., Fariselli, P., & Casadio, R. (2002a). Improved prediction of solvent accessibility and number of residue contacts in proteins. *Proteins: Structure, Function and Genetics*, *47*(2), 142–153.

Pollastri, G., Baldi, P., Fariselli, P., & Casadio, R. (2002b). Prediction of coordination number and relative solvent accessibility in proteins. *Proteins*, *47*, 142–153.

Pollastri, G., Baldi, P., Vullo, A., & Frasconi, P. (2003). Prediction of protein topologies using GIOHMMs and GRNNs. In S. Becker, S. Thrun, & K. Obermayer, *Advances in neural information processing systems* (vol. 15). Cambridge, MA: MIT Press.

Pollastri, G., & McLysaght, A. (in press). Porter: A new, accurate server for protein secondary structure prediction. *Bioinformatics*.

Pollastri, G., Przybylski, D., Rost, B., & Baldi, P. (2002c). Improving the prediction of protein secondary structure in three and eight classes using recurrent neural networks and profiles. *Proteins*, *47*(2), 228–235.

Qian, N., & Sejnowski, T. J. (1988). Predicting the secondary structure of globular proteins using neural network models. *Journal of Molecular Biology*, *202*, 865–884.

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

Rost, B., & Sander, C. (1993). Improved prediction of protein secondary structure by use of sequence profiles and neural networks. *Proceedings of the National Academy of Sciences of the United States of America*, *90*(16), 7558–7562.

Sturt, P., Costa, F., Lombardo, V., & Frasconi, P. (2003). Learning first-pass structural attachment preferences with dynamic grammars and recursive neural networks. *Cognition*, *88*(2), 133–169.

Vendruscolo, M., Kussel, E., & Domany, E. (1997). Recovery of protein structure from contact maps. *Folding & Design*, *2*, 295–306.

Vullo, A., & Frasconi, P. (2004). Disulfide connectivity prediction using recursive neural networks and evolutionary information. *Bioinformatics*, *20*(5), 653–659.

Zemla, A., Venclovas, C., Fidelis, K., & Rost, B. (1999). A modified definition of SOV, a segment-based measure for protein secondary structure prediction assessment. *Proteins*, *34*(2), 220–223.