

Disulfide Connectivity Prediction using Recursive Neural Networks and Evolutionary Information

Alessandro Vullo

Paolo Frasconi

Department of Systems and Computer Science, Università di Firenze
Via di S. Marta 3, 50139-I Firenze, Italy

Email: {vullo,paolo}@dsi.unifi.it

Phone: +39 055 4796 362, Fax: +39 055 4796 363

Abstract

We focus on the prediction of disulfide bridges in proteins starting from their amino acid sequence and from the knowledge of the disulfide bonding state of each cysteine. The location of disulfide bridges is a structural feature that conveys important information about the protein main chain conformation and it can therefore help towards the solution of the folding problem. Existing approaches based on weighted graph matching algorithms do not take advantage of evolutionary information. Recursive neural networks (RNN), on the other hand, can handle in a natural way complex data structures such as graphs whose vertices are labeled by real vectors, allowing us to incorporate multiple alignment profiles in the graphical representation of disulfide connectivity patterns.

The core of the method is the use of machine learning tools to rank alternative disulfide connectivity patterns. We develop an ad-hoc RNN architecture for scoring labeled undirected graphs that represent connectivity patterns. In order to compare our algorithm with the previous method, we report experimental results on the SWISS-PROT 39 data set. We found that using multiple alignment profiles allows us to obtain improvements of prediction accuracy, clearly demonstrating the important role played by evolutionary information.

Keywords. Prediction of disulfide bridges, machine learning, recursive neural networks.

1 Introduction

The sequence of proteins which contain cysteine residues is subject to post-translational covalent modifications and cysteines can occur either in oxidized or thiol form. Two oxidized cysteines

uniquely pair to form a covalent bond, known as disulfide bridge. As reported by experiments in protein engineering (Matsumura et al., 1989), disulfide bridges can increase the thermodynamic stability of the native state, because they contribute to a reduction of the number of the unfolded conformations, thus of the entropic cost of folding a polypeptide chain into its native state (Harrison & Sternberg, 1994; Wedemeyer, Welkner, Narayan, & Scheraga, 2000). Depending on their number and location, these bonds may connect very distant portion of the sequence. Therefore, they add strong structural constraints that can be very helpful towards the ab-initio prediction of the 3D structure.

In the absence of an experimentally determined structure, sequence archives do not report reliable information relating either the oxidized form of cysteines or disulfide bridges location. The prediction task can thus be conveniently decomposed in two separate steps. First, the disulfide-bonding state of each cysteine is predicted from sequence, a binary classification problem that has been solved using several machine learning algorithms such as neural networks, (Fariselli, Riccobelli, & Casadio, 1999; Fiser & Simon, 2000), support vector machines (Frasconi, Passerini, & Vullo, 2002) and hidden Markov models (Martelli, Fariselli, Malaguti, & Casadio, 2002). Second, the location of disulfide bridges is predicted starting from knowledge of bonded cysteines. This paper focuses on the second task, that has received relatively scarce attention in the literature. To the best of our knowledge, the only published method (Fariselli & Casadio, 2001) is based on a weighted graph representation of disulfide bridges, where vertices are oxidized cysteines and undirected edges are labeled by the strength of interaction (contact potential) in the associated pair of cysteines. First, constrained optimization is used to find an optimal set of weights. After a complete labeled graph is obtained, candidate bridges are located by finding the maximum weight perfect matching¹. The problem can be solved in polynomial time using linear programming. Nevertheless, the computation of contact potentials is a time consuming process. In a subsequent improvement (Fariselli, Martelli, & Casadio, 2002), neural network predictions were used for labeling edges with contact potentials, increasing the predictive power and concomitantly reducing the training time. This method achieves satisfactory results for the simplest cases (4 and occasionally 6 oxidized cysteines).

The method we propose in this paper is based on extended recursive neural networks (RNN) (Frasconi, Gori, & Sperduti, 1998), a connectionist model which allows us to formulate classification and regression tasks on structured data, like the graphs representing disulfide connectivity patterns. The network is trained to score candidate graphs according to a similarity metric with respect to the correct graph. During prediction, the score computed by the network is used to exhaustively explore the space of candidate graphs. We show how the method can easily incorporate and effectively exploit evolutionary information and how it can efficiently cover with a broad spectrum of sequences for the disulfide bridge prediction problem.

¹A perfect matching of a graph (V, E) is a subset $E' \subseteq E$ such that each vertex $v \in V$ is met by only one vertex.

Table 1: Number of chains in the two data sets used in the experiments, grouped by number of disulfide bridges (B).

Num Bonds	Num Seqs
2	156
3	146
4	99
5	45
Total	446

2 System and Methods

2.1 The data sets of protein sequences

In order to compare our method to the alternative algorithm described in (Fariselli & Casadio, 2001; Fariselli et al., 2002), we replicated the same experimental setting. In particular, we extracted the same set of sequences from the SWISS-PROT database release no. 39 (SP39), October 2000 (Bairoch & Apweiler, 2000) and we applied the same filtering procedure in order to include only high quality and experimentally verified intra-chain disulfide bridge annotations. Also, the experiments were carried out by excluding from the data sets all the chains having more than 10 oxidized cysteines. Less than 20% of SWISS-PROT sequences have more than five disulfide bridges (see Figure 1).

Table 1 reports the number of sequences used in our experiments, grouped by data set and by number of disulfide bonds.

2.2 Prediction of the location of disulfide bridges

A disulfide connectivity pattern has a simple representation in terms of an undirected graph $G = (V, E)$. The vertex set V represents the set of bonded cysteines and an edge $e \in E$ corresponds to a disulfide bridge between its adjacent cysteines. Admissible vertex and edge sets are constrained because an even number of intra-chain bonded cysteines is required and a cysteine can only be bridged to one and only one different cysteine. Thus, we have $|V| = 2B$, $|E| = B$ and $degree(v) = 1$ for any $v \in V$ (perfect matching), where B denotes the number of disulfide bonds in a chain.

We introduce a simple formulation for the problem of predicting the correct connectivity pattern for a given disulfide bonded chain: find the best possible candidate as given by a suitable scoring function. This function maps undirected graphs to real numbers. Let $G^* = (V, E^*)$ denote

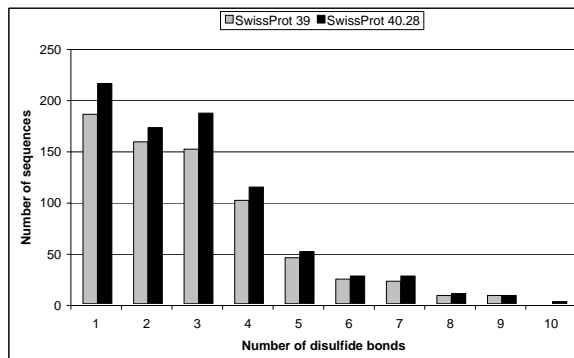


Figure 1: Distribution of cysteine-rich sequences in the SWISS-PROT data sets. Chains are grouped according to the number of disulfide bonds.

the target connectivity pattern. Let \mathcal{G} be the set of candidate solutions and let $s(E, V) : \mathcal{G} \mapsto [0, 1]$ be a scoring function mapping $G \in \mathcal{G}$ into $[0, 1]$ and satisfying the following assumptions:

1. $s(E, V) = 1$ iff $E = E^*$;
2. for every pair (E_1, E_2) of edge sets,

$$|E_1 \cap E^*| \geq |E_2 \cap E^*| \Rightarrow s(E_1, V) \geq s(E_2, V)$$

The function $s(G)$ induces a partial order relation over the set of candidate pattern graphs sharing the same vertex set V . In other words, if $s(G) > s(G')$ then $G \succ G'$. Given $s(E, V)$ or an approximation of this function, a pattern can be predicted by a simple procedure enumerating all possible candidates and giving as output one graph with maximal score. The predicted pattern $\tilde{G} = (V, \tilde{E})$ is formally computed as:

$$\tilde{E} = \arg \max_{E \in \mathcal{E}} s(E, V) \tag{1}$$

where \mathcal{E} is the set of possible edge sets satisfying the given perfect matching constraints. It can be easily shown that the function

$$s^*(E) = \frac{|E \cap E^*|}{|E|} \tag{2}$$

satisfies the required assumptions, thus can be effectively used in the procedure of Equation 1. The given scoring function represents the fraction of correct pairs in the candidate solution. The definition of Equation 2 implies that $s : \mathcal{G} \mapsto [0, 1]$ is neither injective nor onto $[0, 1]$. The codomain is the finite discrete set $\{s_0, \dots, s_{B-2}, s_{B-1}\}$, with $s_i = i/B$, $i = 0 \dots B - 2$ and

$s_{B-1} = 1$. In this case, many different candidates project into the same value, but still we have that $G = G^*$ if $s_{B-1} = 1$. Therefore, given the true or a nearly perfect function, the search guarantees to find the target solution.

To analyze the computational complexity involved, first observe that we need to generate the whole set \mathcal{E} of possible solutions. Given a chain with $|V| = 2B = n$ cysteines, the size of this set is

$$(n-1)!! = \prod_{i \leq n/2} (2i-1) = \frac{n!}{2^{n/2}(n/2)!}$$

By the last equality, it holds that

$$\left(\frac{n}{4}\right)^{n/2} < (n-1)!! < \left(\frac{n}{2}\right)^{n/2},$$

thus $\Omega\left(\left(\frac{\sqrt{n}}{2}\right)^n\right)$ steps are necessary to compute \mathcal{E} . Each step requires the evaluation of the function s for the current candidate. Assuming s to be linear in $|V|$, it follows that the algorithm takes time $\Omega\left(n\left(\frac{\sqrt{n}}{2}\right)^n\right)$. This computational complexity limits the application of the algorithm only to chains with few bridges (1 to 5 bridges). As previously noted by (Fariselli & Casadio, 2001), this is not a severe problem, since the constraint $1 \leq B \leq 5$ is satisfied for most of the chains of interest (see Fig. 1). In the following we propose a connectionist model capable of learning $s(E, V)$ from examples of known disulfide bond patterns.

3 Algorithm

The apparent simplicity for finding G^* with the above procedure is due to the existence of an oracle that is able to compute $s(E, V)$ for every candidate graph: this requires the knowledge of E^* , a kind of information which is obviously unknown at prediction time. In realistic situations, the scoring function for the algorithm of Equation 1 must rely only on V to assign scores to candidate solutions.

3.1 Bi-recursive neural network architecture

A labeled graph on a set \mathcal{X} is a pair $G = (V, E)$, with $E \subset V \times V$, and with a function $x : V \rightarrow \mathcal{X}$ that associates a label x_v to each vertex $v \in V$. The set of all finite size labeled graphs on \mathcal{X} is denoted $\mathcal{X}^\#$. Classification or regression tasks for which the input portion consists of a labeled graph can be formulated as a mapping from $\mathcal{X}^\#$ to a set of categories (classification) or to real numbers (regression). Since graphs have variable size, regression in this case need to be represented as the composition of two functions, a mapping $F : \mathcal{X}^\# \rightarrow \Phi$ that transforms input

graphs to an intermediate representation in a vector space Φ , and a mapping $g : \Phi \rightarrow \mathbb{R}$ from vectors to real numbers.

The solution used in this paper is based on a generalization of recursive neural networks (RNN). In this class of models, the mapping $F : \mathcal{X}^\# \rightarrow \Phi$ is adaptive and Φ is a low-dimensional space. This is in contrast to alternative approaches based for example on convolution kernels (Haussler, 1999), where Φ is a high-dimensional (or infinite dimensional) feature space and F is a fixed map.

The theory developed in (Frasconi et al., 1998), briefly summarized below, holds in the case of directed ordered acyclic graphs (DOAG) with bounded connectivity and that possess a supersource. Ordered in this context refers to the existence of a total order defined on the set of children of each vertex. A supersource r is a vertex having the property that for every other vertex $v \in V$ there exists a directed path from r to v . Under these assumptions, F can be written recursively as follows. For each vertex v , we introduce a representation vector $\phi_v \in \mathbb{R}^n$ recursively computed as:

$$\phi_v = \begin{cases} 0 & \text{if } \text{ch}_v = \emptyset \quad (\text{Base step}) \\ f(x_v, \phi_{\text{ch}_v}) & \text{otherwise} \quad (\text{Induction}) \end{cases} \quad (3)$$

where $\phi_{\text{ch}_v} = (\phi_{\text{ch}_v^1}, \dots, \phi_{\text{ch}_v^k})$ is the k -tuple of labels of v 's children, ch_v^i denotes the i -th child of v and k is the maximum outdegree in the class of graphs being considered. In RNNs, functions $f(\cdot)$ and $g(\cdot)$ are implemented by adaptive neural networks with connection weights ϑ and θ , respectively. We assume x_v , the label of vertex v , to be encoded by a real vector in \mathbb{R}^m . Thus, the network of $f(\cdot)$ has $m + kn$ input units and n output units. The computation in Equation 3 proceeds in a bottom-up fashion, from 'leaf' vertices to the supersource. Since we assume that G is acyclic, propagation order can be obtained by sorting topologically the vertex set. The adaptive mapping from graphs to features is simply accomplished by taking the label at the supersource: $F(G) = \phi_r$. It turns out that for each vertex v , vector ϕ_v is the encoding of the subgraph induced by v and all its descendants.

Training is performed using a set of pairs $\{(G_i, y_i), i = 1, \dots, N\}$ where $y_i \in [0, 1]$ is the desired output for graph G_i and $g(F(G_i)) = g(\phi_r^i)$ is its network global output. Parameters θ and ϑ are adjusted by minimizing the error function

$$C(\theta, \vartheta) = \frac{1}{2} \sum_{i=1}^N \left(y_i - g(\phi_r^i) \right)^2 \quad (4)$$

As shown in (Frasconi et al., 1998), a gradient descent algorithm can be obtained by propagating errors backward in structure (i.e. in a top-down fashion, starting from the supersource and following a reverse topological sort of G).

Graphs describing disulfide connectivity do not match the above framework since they are undirected, disconnected and unordered. However, we can introduce a suitable transformation that converts a disulfide graph $G = (V, E)$ into a DOAG G' having a supersource. First, note that the

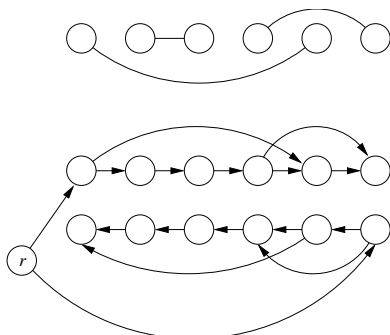


Figure 2: Transforming a disulfide graph (top) into a supersource DOAG (bottom).

set of vertices V can be ordered reading the protein sequence from left to right. Edges can be thus oriented so that the source vertex precedes in sequence the target vertex. In this way, G is converted into a directed graph. Moreover, additional sequential edges can be added to connect vertices that are adjacent in sequence. After doing so G is connected and has a supersource. In practice, if we use a model like the one described by Equation 3, the role played by sequential links is to propagate information from left to right. More precisely, the feature vector ϕ_v associated with each half-cystine v would summarize information about all the upstream half-cystines and candidate bridges. Note, however, that dependencies in biological sequences are not unidirectional from left to right. To propagate information in both direction we can duplicate G and transpose the edge set of the copy, as shown in Figure 2. A similar approach already proved to be effective for the prediction of protein secondary structure (Baldi, Brunak, Frasconi, Soda, & Pollastri, 1999). In our implementation, the recursive computation described by Equation 3 is actually piecewise stationary: $\vartheta_v = \vartheta_u$ if and only if v and u are both in the original graph or in the transposed copy. This means that we have three sets of adjustable weights for function F : one for vertices linked upstream to downstream, one for vertices linked downstream to upstream, and a distinguished set of weights for the supersource. It turns out that ϕ_v ($v \neq r$), is now represented by coupling the outputs of two functions $f(\cdot)$ and $b(\cdot)$ computed by two neural networks respectively with the up-to-downstream and down-to-upstream connection weights.

4 Implementation

4.1 Performance Measures

Let \mathcal{D} denote a set of proteins and let $G_i = (V_i, E_i)$ and $G_i^* = (V_i, E_i^*)$ denote the predicted and correct connectivity patterns for the i -th protein in the set \mathcal{D} , respectively. Prediction indices are defined as

$$Q_p = \frac{\sum_{i=1}^{|\mathcal{D}|} \delta(E_i, E_i^*)}{|\mathcal{D}|}, \quad Q_c = \frac{\sum_{i=1}^{|\mathcal{D}|} |E_i \cap E_i^*|}{\sum_{i=1}^{|\mathcal{D}|} |E_i|} \quad (5)$$

where $\delta(x, y) = 1$ if $x = y$ and 0 otherwise. The prediction index Q_p is pattern-based and measures the fraction of correctly assigned connectivity patterns. It estimates predictive performance at the protein level, namely the probability that a whole prediction is correct. Note that Q_p can be decreased by proteins having large values of B and yet the location of several bridges could have been predicted correctly. For this reason we also use the complementary couple-based prediction index Q_c , defined as the fraction of correctly predicted disulfide bridges.

4.2 Data set generation and input data

For each chain in the data set we generated connectivity graphs by including all the possible B !! connectivity patterns. We obtained 55,683 undirected graphs for the SP39 data set. Training is performed by assigning a target output score to each connectivity graph. More formally, if $G_i^* = (V_i, E_i^*)$ is the i -th instance for a data set \mathcal{D} with N chains and $\mathcal{E}_i = \{G_{i_1}, \dots, G_{i_{|\mathcal{E}_i|}}\}$ is the set of candidate patterns for chain i , we used the set of pairs $\{(G_{i_j}, y_{i_j}), i = 1, \dots, N, j = 1, \dots, |\mathcal{E}_i|\}$, where y_{i_j} represents the similarity between the candidate G_{i_j} and the true pattern G_i^* , as computed by Equation 2.

Each vertex in a connectivity graph contains information describing the local environment of the corresponding bonded cysteine. More precisely, we used a window of size $2k + 1$ amino acids centered around the bonded cysteine and we encoded each amino acid position by a vector of 20 components. By comparison, (Fariselli & Casadio, 2001) used edge-labeled graphs where each edge was annotated by the contact potential between two adjacent vertices. As pointed out in the statistical analysis of (Harrison & Sternberg, 1994), sequence separation between bonded cysteines and sequence length correlate with specific connectivity patterns. To take advantage of this information we enriched label vectors with two additional features: the normalized sequence position t/L (where t is the cysteine position along the chain and L is the chain length in residues) and the relative sequence length L/L_{max} (where L_{max} is the maximum chain length observed in the database).

We consider two different encodings for the positions along the window: single-sequence and profile based. In the single-sequence case, an all-zero-but-one binary vector identifies the residue

type at a given window position. In this case the cysteine in the center of window is not taken into account, being it always present and carrying no information. In profile-based encoding, each amino acid position is described by the vector of multiple sequence alignments profile. In this case the central position corresponding to the cysteine is retained. In all our experiments we used a window of size $k = 2$ resulting in labels vectors of dimension 82 (single-sequence) or 102 (profile-based). The position-specific scoring matrix of each chain in the filtered SP39 data set was created by running two iterations of the PSI-BLAST program (Altschul, Madden, Schaffer, Zhang, Miller, & Lipman, 1997) against the non-redundant SWISS-PROT+TrEmbl data set of sequences and using default arguments.

4.3 Experimental protocol

Networks were trained by optimizing the cost function 4 using gradient descent — see (Frasconi et al., 1998) for details on backpropagation in recursive neural networks. We used the online stochastic approximation, updating weights after the presentation of each graph.

Training and testing of the recursive neural model was performed according to the same 4-fold cross validation procedure as used in (Fariselli & Casadio, 2001). In order to automatically stop the four training phases and to control overfitting, we adopted two strategies. In a first one, for each fold we used a validation set containing 20% of randomly chosen sequences from the original train set. In a second setting, we regularized the neural network using weight-decay and we used the entire training data in each fold for computing gradients.

Preliminary experiments were carried out testing different choices of the adjustable parameters. More precisely, model performances were evaluated varying the architecture of the neural networks implementing the forward and backward transition function ($f(\cdot)$ and $b(\cdot)$, respectively) and the global output function $g(\cdot)$. All the results showed in the next section are relative to transition networks with 20 output units, no hidden layers and a single sigmoid unit representing the global output function (the score of a graph). During preliminary experiments we found that training separate networks for the sets of proteins having the same number of bridges helps improving generalization. In all the subsequent experiments we therefore used four separate networks for predicting the connectivity of proteins having $B = 2, 3, 4$ and 5 , respectively. On average, a cross-validation procedure applied to the SP39 data set took about 12 hours on a single Pentium IV running at 1GHz.

5 Discussion

Table 2 summarizes the results obtained by running the experiments with the B -specialized networks, as it was described in the previous section. We report the estimated pattern-based and

	2		3		4		5	
Method	Q_p	Q_c	Q_p	Q_c	Q_p	Q_c	Q_p	Q_c
Frequency	.58	.58	.29	.37	.01	.10	.00	.23
MC graph-matching	.56	.56	.21	.36	.17	.37	.02	.21
NN graph-matching	.68	.68	.22	.37	.20	.37	.02	.26
brnn1-sequence	.59	.59	.17	.30	.10	.22	.04	.18
brnn1-profiles	.65	.65	.46	.56	.24	.32	.08	.27
brnn2-sequence	.59	.59	.22	.34	.18	.30	.08	.24
brnn2-profiles	.73	.73	.41	.51	.24	.37	.13	.30

Table 2: Prediction accuracies of different disulfide connectivity predictors estimated by 4-fold cross validation on the SwissProt 39 dataset. The first row is the baseline accuracy obtained by predicting the most frequent pattern. The next two rows are previous results based on the contact potential method reported in (Fariselli & Casadio, 2001) and (Fariselli, et al., 2002), respectively. The remaining four rows refer to the methods described in this paper.

couple-based prediction indexes Q_p and Q_c for each of the group of chains having the same number of bonds.

Rows from the third to the last one report prediction results as obtained by the algorithms indicated under column labeled Method. “Frequency” is a trivial method consisting of predicting always the most frequent pattern observed in the train set. We use it to compare our algorithm against baseline performances. For the sake of comparison, on the next two rows we also report the already published results obtained respectively in (Fariselli & Casadio, 2001) and (Fariselli et al., 2002). The next two rows report performances obtained by the bi-recursive neural network trained with validation sets and using as input respectively only the information encoded in the sequence (brnn1-sequence) and multiple alignment profiles (brnn1-profile). The last two rows show results of the same model trained using weight-decay as stopping procedure.

Surprisingly, the baseline method performs reasonable quite well in the case of 2 and 3 bonds. Excluded the case of 4 bonds, the method MC graph-matching and the sequence-based recursive neural networks (brnn1-sequence and brnn2-sequence) cannot significantly outperform the trivial algorithm or obtain worse results. NN graph-matching performs better than single-sequence RNNs: the first method uses significantly richer information compared to the input processed by RNNs.

Training RNNs with multiple alignment profiles (brnn1-profiles and brnn2-profiles) allows us to obtain a significant performance improvement for all groups of chains. In brnn2-profiles, we use the same amount of training data as in NN graph-matching and we are able to consistently outperform this method. Despite being the best reported ones, we still obtain low-level results in the

case of 4 and 5 bonds, confirming the difficulties of choosing the right pattern among respectively 105 and 945 possible candidates.

Overall, the combination of our RNNs and multiple alignment profiles has not yet a general applicability, but can be very informative in a real prediction context, especially in the case of 2 and 3 bonds.

6 Conclusions

We have proposed and tested a novel machine learning method for predicting the disulfide connectivity patterns in cysteine-rich proteins. Performance is comparable or better with respect to other existing algorithms in literature. Our method can easily incorporate and process evolutionary information in the form of multiple alignment profiles. Experimental studies give evidence of the benefit in using this type of information.

One obvious direction for further study is to combine cysteine bonding state predictors to a pairing algorithm like the one presented in this paper, in order to build a complete predictor of disulfide bridges. In this perspective, the use of neural networks for solving the pairing problem is potentially advantageous as it allows global optimization of the recursive network together with the parameters of the bonding state predictor. Disulfide bridges can also be seen as a special (and important) case of residue contact. Therefore it may be important in the future to compare and combine predictors of disulfide bridges with predictors of contact maps whose performance is improving but still appears to be not satisfactory for long ranged interactions.

References

- Altschul, S., Madden, T., Schaffer, A., Zhang, A., Miller, W., & Lipman, D. (1997). Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Res.*, *25*, 3389–3402.
- Bairoch, A., & Apweiler, R. (2000). The SWISS-PROT protein sequence database and its supplement TrEMBL. *Nucleic Acids Res.*, *28*, 45–48.
- Baldi, P., Brunak, S., Frasconi, P., Soda, G., & Pollastri, G. (1999). Exploiting the past and the future in protein secondary structure prediction. *Bioinformatics*, *15*, 937–946.
- Fariselli, P., & Casadio, R. (2001). Prediction of disulfide connectivity in proteins. *Bioinformatics*, *17*, 957–964.
- Fariselli, P., Martelli, P. L., & Casadio, R. (2002). A neural network-based method for predicting the disulfide connectivity in proteins. 6th KBASS Conference.
- Fariselli, P., Riccobelli, P., & Casadio, R. (1999). Role of evolutionary information in predicting the disulfide-bonding state of cysteine in proteins. *Proteins*, *36*, 340–346.
- Fiser, A., & Simon, I. (2000). Predicting the oxidation state of cysteines by multiple sequence alignment. *Bioinformatics*, *3*, 251–256.

- Frasconi, P., Gori, M., & Sperduti, A. (1998). A general framework for adaptive processing of data structures. *IEEE Trans. on Neural Networks*, 9(5), 768–786.
- Frasconi, P., Passerini, A., & Vullo, A. (2002). A two stage SVM architecture for predicting the disulfide bonding state of cysteines. In *Proceedings of IEEE Neural Network for signal processing conference*. IEEE Press.
- Harrison, P. M., & Sternberg, M. (1994). Analysis and classification of disulfide connectivity in proteins. *J. Mol. Biol.*, 244, 448–463.
- Haussler, D. (1999). Convolutional kernels on discrete structures. Tech. rep. UCSC-CRL-99-10, University of California at Santa Cruz.
- Martelli, P., Fariselli, P., Malaguti, L., & Casadio, R. (2002). Prediction of the disulfide-bonding state of cysteines in proteins at 88% accuracy. *Protein Sci*, 11(11), 2735–9.
- Matsumura, M., et al. (1989). Substantial increase of protein stability by multiple disulfide bonds. *Nature*, 342, 291–293.
- Wedemeyer, W., Welkner, E., Narayan, M., & Scheraga, H. (2000). Disulfide bonds and protein-folding. *Biochemistry*, 39, 4207–4216.