Journal of VLSI Signal Processing 35, 287–295, 2003

Predicting the Disulfide Bonding State of Cysteines with Combinations of Kernel Machines*

ALESSIO CERONI, PAOLO FRASCONI† , ANDREA PASSERINI AND ALESSANDRO VULLO
Dipartimento di Sistemi e Informatica, Università di Firenze, Italy

Received November 15, 2002; Revised February 10, 2003

Abstract. Cysteines may form covalent bonds, known as disulfide bridges, that have an important role in stabilizing the native conformation of proteins. Several methods have been proposed for predicting the bonding state of cysteines, either using local context or using global protein descriptors. In this paper we introduce an SVM based predictor that operates in two stages. The first stage is a multi-class classifier that operates at the protein level, using either standard Gaussian or spectrum kernels. The second stage is a binary classifier that refines the prediction by exploiting local context enriched with evolutionary information in the form of multiple alignment profiles. At both stages, we enriched profile encoding with information about cysteine conservation. The prediction accuracy of the system is 85% measured by 5-fold cross validation, on a set of 716 proteins from the September 2001 PDB Select dataset.

Keywords: bonding state of cysteines, kernel machines, machine learning, structural genomics

1. Introduction

The prediction of protein three-dimensional structure from sequence is one of the most important and unsolved problem in computational molecular biology. Current sequencing technologies allow us to know entire genomes for an increasing number of organisms. The central dogma of molecular biology (DNA makes RNA makes proteins) implies that from genomes we also know protein sequences, but the biological function carried out by proteins cannot be understood from their sequence alone. Unfortunately, the experimental determination of structures is costly, very time consuming and, perhaps more importantly, often impossible or impractical. The sequence (or primary structure) of a protein can be seen as a string on a twenty letters alphabet, where each letter correspond to one amino acid. An ideal prediction tool would take as input such a string and produce as its output the list of all the atoms coordinates. This ab-initio prediction problem is very challenging and is complicated by translational and rotational invariance. Therefore, it makes sense to solve intermediate steps by predicting invariant structural features, such as secondary structure or pairwise distances between amino acids. These features can either be used to help solving the structure or be useful per se to characterize the biological behavior of the protein. In this paper we focus on one specific prediction problem in this scenario, namely the prediction of the bonding state of cysteines, one of the twenty amino acids that constitute proteins.

The oxidized form of cysteines plays a fundamental role in the stabilization process of the native conformation of proteins. The covalent bonds formed by cysteines, known as disulfide bridges, may connect very distant portion of the sequence. The pattern formed by disulfide bridges can help understanding structural properties of the protein and to identify which family the protein belongs to, giving important insights about its biological function. Moreover, the location of these bonds is a very informative constraint on the

*Based on "A Two-Stage SVM Architecture for Predicting the Disulfide Bonding State of Cysteines" by P. Frasconi, A. Passerini and A. Vullo, which appeared in the Proceedings of the 2002 IEEE International Workshop on Neural Networks for Signal Processing. © 2002 IEEE.

†Present address: Department of Systems & Computer Science, University of Florence, Via di Santa Marta 3, I-50139 Firenze, Italy.

conformational space and the associated information represents a significant step towards folding. Prediction of disulfide bridges from sequence is thus one of the important (and difficult) tasks in structural genomics.

Recent works related to the prediction of disulfide bridges suggest methodologies based on two steps. First, the disulfide-bonding state of each cysteine is predicted (a binary classification problem) [1–3]. Subsequently, once candidate cystines are known, other algorithms can be used to predict the actual location of disulfide bridges [4]. In this paper we are interested in the first step. Currently available predictors are all based on neural network approaches.

The program CYSPRED developed by Fariselli et al. [1] (accessible at `http://gpcr.biocomp.unibo.it/predictors/cyspred/`), uses a neural network with no hidden units, fed by a window of $2k+1$ residues, centered around the target cysteine. Each element of the window is a vector of 20 components (one for each amino acid) obtained from multiple alignment profiles. This method achieved 79% accuracy (correct assignment of the bonding state) measured by 20-fold cross validation and using a non-redundant set of 640 high quality proteins from PDB Select [5] of October 1997. Accuracy was boosted to 81% using a jury of six networks.

The program CYSREDOX, later developed by Fiser and Simon [2] (accessible at `http://pipe.rockefeller.edu/cysredox/cysredox.html`) achieves state-of-the-art performance by exploiting the observation that cysteines and half cystines[1] rarely co-occur in the same protein. The important criterion in [2] is that if a larger fraction of cysteines are classified as belonging to one oxidation state, then all the remaining cysteines are predicted in the same state. The accuracy of this method is as high as 82%, measured by a jack-knife procedure (leave-one-out applied at the level of proteins) on a set of 81 protein alignments.

More recently, Mucchielli-Giorgi et al. [3] have proposed a predictor that exploits both local context and global protein descriptors (normalized statistics based on amino acid frequencies, protein size, and number of cysteines). One interesting finding in [3] is that prediction of covalent state based on global descriptors is more accurate (77.7%) than prediction based on local descriptors alone (67.3%). This is not surprising in the light of the results presented in [2] because when using global descriptors all the cysteines in a given protein are deemed to be assigned to the same state. Thus a good method for classifying proteins in two classes is also a good method for predicting the bonding state of each cysteine. The effect of local context however is not negligible: results in [3] show that 79.3% accuracy can be achieved by using an input vector joining global and local descriptors (results in this case are measured by 5-fold cross-validation on a set of 559 proteins from Culled PDB). Although results are not directly comparable because different datasets are used the performance levels attained in [1] and [2] suggest that multiple alignment profiles are more discriminative than frequency-based descriptors when prediction is based on a local window only.

Starting from the above observations, in this paper we propose a novel approach for exploiting the key fact that cysteines and half cystines rarely co-occur. Classification is achieved in two stages. The first classifier predicts the type of protein based on the whole sequence. Classes in this case are "all", "none", or "mix", depending whether all, none, or some of the cysteines in the protein are involved in disulfide bridges. The second binary classifier is then trained to selectively predict the state of cysteines for proteins assigned to class "mix", using as input a local window with multiple alignment profiles together with the protein global descriptor. The overall model is implemented as a probabilistic combination of support vector machines, as detailed in the remainder of the paper.

Furthermore, we study two extensions for improving the three-state classifier in the above architecture. First, we use a kernel machine based on the spectrum kernel [6] that exploits the whole protein sequence as input. Second, we introduce evolutionary information in the form of cysteine conservation in multiple alignments. These modifications allow to improve prediction accuracy to 85%.

2. Two-Stage Classification of Cysteines

Let $Y_{i,t}$ be a binary random variable associated with the bonding state of cysteine at position t in protein i. By W_t^k we denote the context of cysteine t (a window of size $2k+1$ centered around position t) enriched with evolutionary information in the form of multiple alignment profiles. Moreover, let D_i denote a global set of attributes (descriptors) for protein i. We are interested in building a model for $P(Y_{i,t} = 1 \mid D_i, W_t^k)$.

For each protein, let C_i be a three-state variable that represents the propensity of the protein to form disulfide bridges. The possible states for C_i are "all",

"none", and "mix", depending whether all, none, or some of the cysteines in the protein are involved in disulfide bridges. After introducing C_i, the model can be decomposed as follows:

$$P\left(Y_{i,t}, D_i, W_t^k\right) = \sum_{C_i} P\left(Y_{i,t} \mid D_i, W_t^k, C_i\right) P\left(C_i \mid D_i, W_t^k\right). \quad (1)$$

We can simplify the above model by introducing some conditional independence assumptions. First, we assume that the type of protein C_i depends only on its descriptor: $P(C_i \mid D_i, W_t^k) = P(C_i \mid D_i)$. Second, we simplify Eq. (1) by remembering the semantics of C_i:

$$\begin{aligned} P\left(Y_{i,t} = 1 \mid D_i, W_t^k, C_i = \text{all}\right) &= 1 \\ P\left(Y_{i,t} = 1 \mid D_i, W_t^k, C_i = \text{none}\right) &= 0 \end{aligned} \quad (2)$$

(this can be seen as a particular form of context-specific independence [7]). As a result, the model in Eq. (1) can be implemented by a cascade of two classifiers. Intuitively, we start with a multi-class classifier for computing $P(C_i \mid D_i)$. If this classifier predicts one of the classes "all" or "none", then all the cysteines of the protein should be classified as disulfide-bonded or nondisulfide-bonded, respectively. If instead the protein is in class "mix", we refine the prediction using a second (binary) classifier for computing $P(Y_{i,t} \mid D_i, W_t^k, C_i = \text{mix})$. Thus the prediction is obtained as follows (see also Fig. 1):

$$\begin{aligned} P\left(Y_{i,t} = 1 \mid D_i, W_t^k\right) &= P\left(Y_{i,t} = 1 \mid D_i, W_t^k, C_i = \text{mix}\right) \\ &\quad \times P(C_i = \text{mix} \mid D_i) + P(C_i = \text{all} \mid D_i) \end{aligned} \quad (3)$$

By comparison, note that the method in [2] cannot assign different bonding states to cysteine residues in the same sequence.

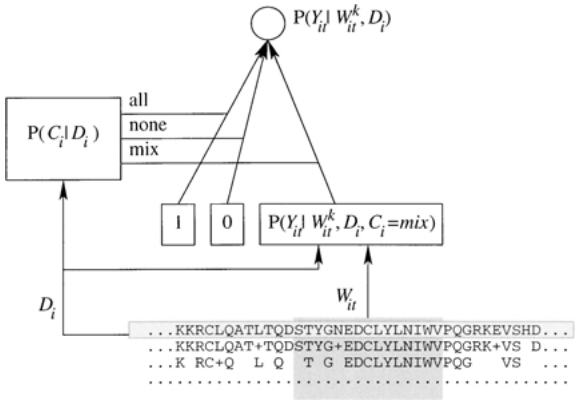


Figure 1. The two-stage system. The protein classifier on the left uses a global descriptor based on amino acid frequencies. The local context classifier is fed by profiles derived from multiple alignments together with protein global descriptor.

3. Implementation Using Probabilistic SVM

Kernel machines, and in particular support vector machines (SVM), are motivated by Vapnik's principle of structural risk minimization in statistical learning theory [8]. In the simplest case, the SVM training algorithm starts from a vector-based representation of data points and searches a separating hyperplane that has maximum distance from the dataset, a quantity that is know as the margin. More in general, when examples are not linearly separable vectors, the algorithm maps them into a high dimensional space, called *feature space* where they are almost linearly separable. This is typically achieved via a kernel function that computes the dot product of the images of two examples in the feature space. The popularity of SVM is due to the existence of theoretical results guaranteeing that the hypothesis obtained from training data minimizes a bound on the error associated with (future) test data.

The decision function associated with an SVM is based on the sign of the distance from the separating hyperplane:

$$f(\mathbf{x}) = \sum_{i=1}^{N} y_i \alpha_i K(\mathbf{x}_1, \mathbf{x}_i) \quad (4)$$

where $\mathbf{x}$ is the input vector, $\{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ is the set of support vectors, $K(\cdot, \cdot)$ is the kernel function, and y_i is the class of the i-th support vector ($+1$ or -1 for positive and negative examples, respectively).

3.1. *Probabilistic Outputs in SVM*

In their standard formulation SVMs output hard decisions rather than conditional probabilities. However, margins can be converted into conditional probabilities in different ways both in the case of binary classification [9, 10] and in the case of multi-class classification [11]. The method used in this paper extends the algorithm presented in [10], where margins in Eq. (4) are

mapped into conditional probabilities using a logistic function, parameterized by an offset B and a slope A:

$$P(C_i = 1 \mid \mathbf{x}) = \frac{1}{1 + \exp(-Af(\mathbf{x}) - B)} \quad (5)$$

In [10], parameters A and B are adjusted according to the maximum likelihood principle, assuming a Bernoulli model for the class variable. This is extended here to the multi-class case by assuming a multinomial model and replacing the logistic function by a softmax function [12]. More precisely, assuming Q classes, we train Q binary classifiers, according to the one-against-all output coding strategy. In this way, for each point $\mathbf{x}$, we obtain a vector $[f_1(\mathbf{x}), \ldots, f_Q(\mathbf{x})]$ of margins, that can be transformed into a vector of probabilities using the softmax function as follows:

$$g_q(\mathbf{x}) = P(C = q \mid \mathbf{x}) = \frac{e^{A_q f_q(\mathbf{x}) + B_q}}{\sum_{r=1}^{Q} e^{A_r f_r(\mathbf{x}) + B_r}} \quad (6)$$

The softmax parameters A_q, B_q are determined as follows. First, we introduce a new dataset $\{(f_1(\mathbf{x}_i), \ldots, f_Q(\mathbf{x}_i), \mathbf{z}_i), i = 1, \ldots, m\}$ of examples whose input portion is a vector of Q margins and output portion is a vector $\mathbf{z}$ of indicator variables encoding (in one hot) one of Q classes. As suggested in [10] for the two classes case, this dataset should be obtained either using a hold-out strategy, or a k-fold cross validation procedure.[2] Second we derive the (log) likelihood function under a multinomial model, and search the parameters A_q and B_q that maximize

$$\ell = \sum_i \sum_{q=1}^{Q} z_{q,i} \log g_q(\mathbf{x}_i) \quad (7)$$

where $z_{q,i} = 1$ if the i-th training example belongs to class q and $z_{q,i} = 0$ otherwise. Optimization can be carried out by simple gradient descent.

3.2. A Fully-Observed Mixture of SVM Experts

While the above method yields multiclass conditional probabilities it does not yet implement the model specified by Eq. (3). We now discuss the following general model, that can be seen as a variant of the mixture-of-experts architecture [13]:

$$P(Y = 1 \mid \mathbf{x}) = \sum_{q=1}^{Q} P(C = q \mid \mathbf{x}) P(Y = 1 \mid C = q, \mathbf{x}) \quad (8)$$

In the above equation, $P(C = q \mid \mathbf{x})$ is the probability that q is the expert for data point $\mathbf{x}$, and $P(Y = 1 \mid C = q, \mathbf{x})$ is the probability that $\mathbf{x}$ is a positive instance, according to the q-th expert. Collobert et al. [14] have recently proposed a different SVM embodiment of the mixture-of-experts architecture, the main focus in their case being on the computational efficiency gained by problem decomposition. Our present proposal for cysteines is actually a simplified case since the discrete variable C associated with the gating network is not hidden.[3] Under this assumption there is no credit assignment problem and a simplified training procedure for the model in Eq. (8) can be derived as follows.

Let $f'_q(\mathbf{x})$ denote the margin associated with the q-th expert. We may obtain estimates of $P(Y = 1 \mid C = q, \mathbf{x})$ using a logistic function as follows:

$$p_q(\mathbf{x}) = P(Y = 1 \mid C = q, \mathbf{x}) = \frac{1}{1 + \exp(A'_q f'_q(\mathbf{x}) + B'_q)}. \quad (9)$$

Plugging Eqs. (6) and (9) into Eq. (8), we obtain the overall output probability as a function of $4Q$ parameters: A_q, B_q, A'_q, and B'_q. These parameters can be estimated by maximizing the following likelihood function

$$\ell = \sum_{i=1}^{m} \frac{1 - y_i}{2} \log\left(\sum_{q=1}^{Q} g_q(\mathbf{x}_i) p_q(\mathbf{x}_i) \right) \quad (10)$$

The margins to be used for maximum likelihood estimation are collected by partitioning the training set into k subsets. On each iteration all the $2Q$ SVMs are trained on $k - 1$ subsets and the margins computed on the held-out subset. Repeating k times we obtain as many margins vectors $(f_1(\mathbf{x}), \ldots, f_Q(\mathbf{x}), f'_1(\mathbf{x}), \ldots, f'_Q(\mathbf{x}))$ as training examples. These vectors are used to fit the parameters A_q, B_q, A'_q, and B'_q on Eq. (10). Finally, the $2Q$ machines are re-trained on the whole training set.

3.3. Spectrum Kernel

The ultimate goal of predicting the bonding state of cysteines is the location of disulfide bonds, a structural feature which depends on the properties of possibly very distant portions of the sequence. Therefore, it might be useful to adopt computational approaches which can exploit the whole sequence as input. Standard kernels assume a vectorial representation of the input data and require a prior processing step where a fixed set of features is extracted from each sequence.

On the other hand, convolution kernels [15] allow to process structured data directly. The spectrum kernel [6] is a convolution kernel specialized for string comparison problems. Given all the strings of length k in a certain alphabet A, the k-spectrum of a sequence s is the vector $\Phi_k(s)$ whose components count the occurrences of each substring of s. The k-spectrum kernel of s and s' is then defined as the dot product

$$K(s, s') = \Phi_k(s)^\top \Phi_k(s')$$

and can be computed efficiently using suffix trees, a data structure that has been employed for solving several problems related to string matching [16].

A suffix tree for a given string s of length m is a tree with exactly m leaves, where each path from the root to a leaf is a suffix of s. Ukkonen's algorithm constructs suffix trees in time $O(m)$ [17]. In our case, a suffix tree can be used to identify all the substrings of length k contained in the input sequence by following all the possible paths of length k that start from the root of the tree. Moreover, the occurrences of each substring of s can be obtained by counting the number of leaves in the subtree rooted at the end of the corresponding path. Since the number of leaves equals the length of the string, we have a linear-time method to calculate the k-spectrum of a string.

When computing the kernel, further modifications are needed to avoid explicit calculation of dot products. A generalized suffix tree is a suffix tree constructed using more than one string [16]. Given a set of strings, there exists a variant of Ukkonen's algorithm that can build the corresponding generalized suffix tree in time linear in the sum of the length of all the strings. A generalized suffix tree is eventually used to compute the k-spectrum kernel at once, just traversing the tree depth-first and counting the occurrences of every substring of length k.

Interestingly, descriptors based on amino acid frequencies as defined in [3], basically correspond to the use of a spectrum with $k = 1$. Augmenting the feature space by incorporating short subsequences increases the expressive power of the model and may improve prediction accuracy, if k is carefully chosen and enough training sequences are avilable. A more general form of the spectrum kernel can also be constructed summing the values of some k-spectrum kernels for certain values of k. No modifications of the presented algorithm are needed, given that all the k-spectrum kernel with different k can be calculated at once with a single traversal of the tree.

4. Data Preparation

All the experiments were carried out using a significant fraction of the current representative set of non homologous protein data bank chains (PDB Select [5]). We extracted the chains in the file 2001, Sep. 25 listing 1641 chains with percentage of homology identity less than 25%. From this set we retained only high quality proteins on which the DSSP program [18] does not crash, determined only by X-ray diffraction, without any physical chain breaks and resolution threshold less than 2.5 Å. The DSSP program was also used to identify disulfide bonds between cysteines. Proteins with inter-chain bonds were not included in the final dataset containing 716 proteins for a total of 4859 cysteines, 1820 of which in disulfide-bonded state and 3039 in nondisulfide-bonded state. In this dataset, 187 proteins are of type "all", 478 are of type "none", and 51 (i.e. only 7%) of type "mix". Evolutionary information is derived from multiple sequence alignments, obtained in our case from the HSSP database [19].

4.1. Input Encoding

The first stage classifier uses the descriptor D_i as an encoding of global characteristics of the protein chain. We have adopted two different kinds of descriptor representation. The first one is a real vector with 24 components, similar to the one used in [3], which we used in combination with standard kernel SVMs. The first 20 components are $\log(N_i^j / N^j)$, where N_i^j is the number of occurrences of amino acid type j in protein i and N^j is the number of occurrences of amino acid type j in the whole training set. The 21st component is $\log(N_i / N_{\text{avg}})$ where N_i is the length in residues of sequence i and N_{avg} is the average length of the proteins in the training set. The next two components are $N_i^{\text{cys}} / N_{\text{max}}^{\text{cys}}$ and N_i^{cys} / N_i where N_i^{cys} and $N_{\text{max}}^{\text{cys}}$ are respectively the number of cysteines in protein i and the maximum number of observed cysteines in the training set. The last component is a flag indicating whether the cysteine count is odd.

The second descriptor representation is the amino acid sequence itself and it is used in combination with spectrum kernel SVMs, as described in the previous section.

The local input window W_t^k used by the second stage classifier is represented as the set of multiple sequence profile vectors of the residues flanking cysteine at position t. In the experiments, we used a symmetrical

window centered at each cysteine varying the window size parameter k from 8 to 10. Note that although the central residue is always a cysteine, the corresponding feature is still taken into account since the profile in this case indicates the degree of conservation of the cysteine. For each of the $2k + 1$ positions we used a vector of 22 components, enriching the 20-components profile with relative entropy and conservation weight.

4.2. Cysteines Conservation

Cysteines tends to be conserved in multiple alignments when they form disulfide bridges. In the experiments reported below, we made available this information to the three state classifier (none-all-mix) in two alternative ways. First, we defined an extended descriptor with H additional components related to the conservation of cysteines. For $h = 0, \ldots, H - 1$, the h-th extra component is the fraction of cysteines in the sequence whose multiple alignment conservation falls in the bin $[h/H, (h + 1)/H]$. This global descriptor is also fed to the multi-classifier together with the local window. Second, we defined a special sequential representation of the proteins that incorporates evolutionary information. In this representation, a protein is a string in an extended alphabet having $19 + Z$ symbols, where occurrences of C (cysteine) are replaced by a special symbol that indicates the degree of conservation of the cysteines in the correspondent positions of multiple alignments. For example if $Z = 2$, one symbol encodes highly (>50%) conserved cysteines and another one encodes lowly conserved ones.

5. Results

For each classifier we run a preliminary set of experiments to help the choice of the kernel type. In these experiments we used roughly 66% of the proteins for training and the remaining as a validation set. We tried linear, polynomial, and radial basis function (RBF) kernel types. The RBF kernels yielded the best results for the multi-class protein classifier, while binary classification of cysteines was more accurate when using a polynomial kernel of third degree. The protein classifier was additionally implemented and tested with the spectrum kernel.

Keeping fixed the type of kernel, we used a 5-fold cross-validation procedure to assess classification performance. The training procedure has been described in the implementation section, but on each fold we used the framework of algorithmic stability recently proposed in [20] as a tool for tuning kernel hyperparameters. In particular, in the case of RBF kernels, we selected the radii that minimized the generalization error bound based on the leave-one-out error. Softmax parameters (see Eqs. (6) and (9)) were estimated by 3-fold cross validation (inside each fold of the outer 5-fold cross-validation), *after* kernel parameter estimation.

5.1. Description of the Experiments

Table 1 reports the results obtained on the 716 proteins dataset. Each of the three major columns is relative to a different size k of the local window. Minor columns report classification accuracy A, precision P, and recall R. Accuracy (also denoted as Q_2 in other papers) is the fraction of correctly classified cysteines. Precision (or sensitivity) is the fraction of cysteines predicted in the disulfide-bonded state that are actually bonded. Recall (or specificity) is the fraction of disulfide-bonded cysteines that are correctly assigned to their state by the predictor.

Results are reported for five different classifiers. The first row (L) corresponds to a single classifier based on

Table 1. Summary of the experimental results.

	w17			w19			w21		
Method	A (%)	P (%)	R (%)	A (%)	P (%)	R (%)	A (%)	P (%)	R (%)
L	79.3	75.2	67.1	79.4	76.1	66.2	79.8	76.9	66.2
LG	83.3	82.4	70.8	83.0	82.5	69.4	82.8	82.0	69.7
LG + Desc24	84.0	82.6	73.0	83.6	82.1	72.2	83.2	81.6	71.5
LG + Desc29	84.1	85.1	70.3	84.1	85.3	70.0	84.3	85.4	70.4
LG + Spect24	84.8	82.7	75.5	84.9	83.7	74.5	85.1	83.6	75.1

a support vector machine with a 3-rd degree polynomial kernel, that only takes a local window of multiple alignments profiles as input. The following row (LG) reports the results for the same classifier with the inclusion of the 24 values protein descriptor (also used in the first global classifier) as input. In the other three experiments, the local classifier LG is combined with different global classifiers using the proposed two-stage architecture. LG + Desc24 uses a global SVM classifier (RBF kernel) taking as input the 24 values protein descriptors. In LG + Desc29 we extended the global descriptor adding 5 extra inputs encoding cysteine conservation, as explained in the previous section. Finally, in LG + Spect24 we employed the spectrum kernel for the global classifier, with substrings' dimensions from 2 to 5, using the extended alphabet with 24 symbols ($Z = 5$) to incorporate information about cysteine conservation.

5.2. *Discussion*

We performed the first two experiments as a base-line for our two-stages architecture. The results of the simplest local classifier (L) can be easily compared to [1], where the same kind of input is adopted. In order to exploit the idea that a cysteine redox state depend globally on the protein structure, the global descriptor (24 real values) was used as an additional input to the local classifier. This classifier, labeled LG, obtains an accuracy over 83%, comparable to the state-of-the-art architecture [2]. This classifier is also used in all the subsequent experiments to learn the local component $P(Y_{i,t} \mid D_i, W_t^k, C_i = \text{mix})$.

The main contribution of this work is the use of a separated global classifier to estimate the class of a protein, and the combination of this prediction to the output of each local classifier, as described in Section 2. The results of the experiments (LG + Desc24) show an increase in performance between the single-stage and the two-stages architecture. This is a demonstration of the ability of the latter architecture to build a richer model for the global dependencies of the cysteines redox state. A further improvement (LG + Desc29) is obtained by adding the evolutionary information given by the conservation of cysteines (Section 4.2), which proves to be a valuable information for the solution of this problem.

Finally, we employed the spectrum kernel for the global classifier (LG + Spect24). This choice proved to be successful, attaining the best performances, with an accuracy near to 85% for a window of 21 residues. The results demonstrate the capability of the spectrum kernel to effectively process the protein's primary structure as a whole to build its own internal representation. This opposes to the basic polynomial kernel, that needs an encoding of the global characteristics of the protein in a small descriptor, therefore loosing important informations that are written in the amino-acid sequence. Moreover, the use of the spectrum kernel allows us to feed the classifier with a representation of the conservation for every cysteine in the protein, and therefore to discriminate different patterns of conservation.

Generally speaking, when dealing with variable length structures, the use of a spectrum-based kernel (such as the kernel for strings used in this work) on a direct representation of such structures can lead to better results than standard classifiers working on features extracted from the data.

6. Conclusions

We have proposed a novel method for predicting the bonding state of cysteines, achieving state-of-the-art performance on the most recent set of non-redundant sequences from the Protein Data Bank. We have shown that global features extracted through a spectrum kernel can improve prediction accuracy compared to global descriptors based on amino acid frequencies. Furthermore, the encoding of cysteine conservation is valuable information and improves performance in all cases.

There are several obvious directions for further improving this method. First, we have seen that reliable detection of proteins that do not contain mixed types of cysteines is very important for the overall performance. In [3] it was shown that higher prediction accuracy is obtained by training and testing within groups of homogeneous proteins. This result suggests that a mixture-of-experts approach, where the gating network is in charge of determining the protein group, is also likely to yield improved performance. Second, it would be interesting to try the integration of different global approaches for capturing distant information. For instance, recursive neural networks [21] could be used to combine predictions over different cysteines in the same protein.

Acknowledgments

We thank Sauro Menchetti who implemented part of the SVM code used in the experiments.

Notes

1. A cystine is the dimer formed by a pair of disulfide-bonded cysteines.
2. The latter consists of splitting the data set in k disjoint subsets, training on $k-1$ subsets, and using the remaining set for collecting the samples used to estimate the margin distribution.
3. Actually the architecture in Fig. 1 for cysteines is even simpler since two of the experts output a constant prediction.

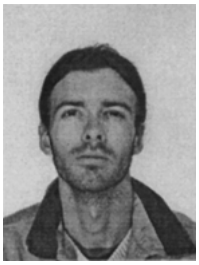
References

1. P. Fariselli, P. Riccobelli, and R. Casadio, "Role of Evolutionary Information in Predicting the Disulfide-Bonding State of Cysteine in Proteins," *Proteins*, vol. 36, 1999, pp. 340–346.
2. A. Fiser and I. Simon, "Predicting the Oxidation State of Cysteines by Multiple Sequence Alignment," *Bioinformatics*, vol. 16, no. 3, 2000, pp. 251–256.
3. M. Mucchielli-Giorgi, S. Hazout, and P. Tuffery, "Predicting the Disulfide Bonding State of Cysteines Using Protein Descriptors," *Proteins*, vol. 46, 2002, pp. 243–249.
4. P. Fariselli and R. Casadio, "Prediction of Disulfide Connectivity in Proteins," *Bioinformatics*, vol. 17, 2001, pp. 957–964.
5. U. Hobohm and C. Sander, "Enlarged Representative Set of Protein Structures," *Protein Science*, vol. 3, 1994, pp. 522–524.
6. C. Leslie, E. Eskin, and W. Noble, "The Spectrum Kernel: A String Kernel for SVM Protein Classification," in *Proc. Pacific Symposium on Biocomputing*, 2002, pp. 564–575.
7. C. Boutilier, N. Friedman, M. Goldszmidt, and D. Koller, "Context-Specific Independence in Bayesian Networks," in *Prof. 12th Conf. on Uncertainty in Artificial Intelligence*, Morgan Kaufmann, 1996, pp. 115–123.
8. V. Vapnik, *Statistical Learning Theory*, New York: John Wiley, 1998.
9. J. Kwok, "Moderating the Outputs of Support Vector Machine Classifiers," *IEEE Transactions on Neural Networks*, vol. 10, no. 5, 1999, pp. 1018–1031.
10. J. Platt, "Probabilistic Outputs for Support Vector Machines and Comparisons to Regularized Likelihood Methods," in *Advances in Large Margin Classifiers*, A. Smola, P. Bartlett, B. Scholkopf, and D. Schuurmans (Eds.), MIT Press, 2000.
11. A. Passerini, M. Pontil, and P. Frasconi, "From Margins to Probabilities in Multiclass Learning Problems," in *Proc. 15th European Conf. on Artificial Intelligence*, F. van Harmelen (Ed.), 2002.
12. J. Bridle, "Probabilistic Interpretation of Feedforward Classification Network Outputs, with Relationships to Statistical Pattern Recognition," in *Neuro-Computing: Algorithms, Architectures, and Applications*, F. Fogelman-Soulie and J. Hérault (Eds.), Springer-Verlag, 1989.
13. R. Jacobs, M. Jordan, S. Nowlan, and G.E. Hinton, "Adaptive Mixtures of Local Experts," *Neural Computation*, vol. 3, no. 1, 1991, pp. 79–87.
14. R. Collobert, S. Bengio, and Y. Bengio, "A Parallel Mixture of SVMs for Very Large Scale Problems," *Neural Computation*, vol. 14, no. 5, 2002.
15. D. Haussler, "Convolution Kernels on Discrete Structures," 1999.
16. D. Gusfield, *Algorithms on Strings, Trees, and Sequences: Computer Science and Computational Biology*, Cambridge University Press, 1997.
17. E. Ukkonen, "On-Line Construction of Suffix Trees," *Algorithmica*, vol. 14, no. 3, 1995, pp. 249–260.
18. W. Kabsch and C. Sander, "Dictionary of Protein Secondary Structure: Pattern Recognition of Hydrogen-Bonded and Geometrical Features," *Biopolymers*, vol. 22, 1983, pp. 2577–2637.
19. R. Schneider, A. de Daruvar, and C. Sander, "The HSSP Database of Protein Structure-Sequence Alignments," *Nucleic Acids Res.*, vol. 25, 1997, pp. 226–230.
20. O. Bousquet and A. Elisseeff, "Stability and Generalization," *Journal of Machine Learning Research*, vol. 2, 2002.
21. P. Frasconi, M. Gori, and A. Sperduti, "A General Framework for Adaptive Processing of Data Structures," *IEEE Trans. on Neural Networks*, vol. 9, 1998, pp. 768–786.

Alessio Ceroni is a Ph.D. student in Computer Engineering at the University of Firenze, Italy. He graduated (2002) in computer Engineering at the University of Firenze with best honours. He did his degree thesis under the supervision of Prof. D.E. Goldberg, while he was a visiting scholar (2001) at the Illinois Genetic Algorithm Laboratory, University of Illinois at Urbana-Champaign, USA. He worked for a few years (1999–2001) with Dr. J. Toro in the field of statistical methods for finance, in collaboration with a Chicago based trading firm. He is currently researching in the field of bioinformatics, focusing on protein structure prediction, with the Machine Learning and Neural Networks Group, at the University of Firenze.

Paolo Frasconi received the M.Sc. degree in Electronic Engineering in 1990, and the Ph.D. degree in Computer Science in 1994, both from the University of Florence, Italy, where he is presently an Associate Professor of Computer Science. He previously held positions at the University of Cagliari, Italy, at the University of Wollongong, Australia, and the Massachusetts Institute of Technology. His current research interests are in the area of machine learning using connectionist models and belief networks, with particular emphasis on problems involving learning about sequential and structured information.

Application fields of his interest include bioinformatics, natural language processing, pattern recognition, and document processing. Dr. Frasconi serves as an Associate Editor for the IEEE Transactions on Neural Networks, the IEEE Transactions on Knowledge and Data Engineering, and the ACM Transactions on Internet Technology. In 2001 he co-directed the NATO Advanced Studies Institute "Artificial Intelligence and Heuristic Methods for Bioinformatics." He is a member of the ACM, the IAPR, the IEEE, and the AI*IA.
http://www.dsi.unifi.it/neural
paolo@dsi.unifi.it



Andrea Passerini received the M.Sc. degree in Computer Engineering in 2000 from the University of Florence, Italy, where he is currently a Ph.D. student. His current research interests include web research and focused crawling, support vector machines and multiclass classification, machine learning applied to bioinformatics, especially protein tertiary structure prediction and gene expression analysis.

Alessandro Vullo received the M.Sc. degree in Computer Engineering in 2000 from the University of Florence, Italy. From 1997 to 2001 he was funder and technical staff member of an Internet company, NewNet Informatica. He is currently a Ph.D. candidate in Computer and Automation Engineering in the Department of Systems and Computer Science (DSI) at University of Florence and Visiting Scholar with the Department of Information and Computer Science (ICS) at University of California at Irvine. His research interests include connectionist models for learning with structured information with a focus in bioinformatics applications and text categorization problems.