Disulfide Bonding State Prediction with SVM Based on Protein Types*

Chih-Ying Lin, Chang-Biau Yang†, Chiou-Yi Hor and Kuo-Si Huang
Department of Computer Science and Engineering
National Sun Yat-sen University
Kaohsiung 80424, Taiwan

***Abstract*—Disulfide bonds play the key role for predicting the three-dimensional structure and the function of a protein. In this paper, we propose an algorithm for predicting the disulfide bonding state of each cysteine in a protein sequence. This method is based on the multi-stage framework and the multi-classifier of the support vector machine. We also design a new training strategy to increase the prediction accuracy. It appends the probabilities to the existing features and then starts a new training procedure repeatedly to improve performance. We perform the experiments on the data set derived from the well-known database Protein Data Bank (PDB). We get 94.2% accuracy for predicting disulfide bonding state, which gets improvement 3.5% compared with the previous best result 90.7%.**

***Index Terms*—disulfide bond; bioinfomatics; support vector machine; cysteine state prediction;**

I. Introduction

Protein is essential for a living thing. The primary structure of a protein is a sequence composed of twenty kinds of standard amino acids. It is usually believed that the protein function is mainly determined by its tertiary (three-dimensional) structure [4]. In recent years, many research topics have drawn much attention, such as the protein functions, protein structures and family classification [1], [9], [17].

In a protein, a *disulfide bond*, which is bonded by two *cysteines*, is very important to form the folding and to maintain the stability of a protein, because the force of disulfide bond is much stronger than others, such as van der Waals force or hydrogen bond. Besides, a disulfide bond might maintain the secondary structure, such as *alpha helix* or *beta sheet*, by resisting the attack of water molecules on hydrogen bonds. It resists water molecules by forming the hydrophobic core of the folded protein through condensing hydrophobic residues around itself. Therefore, the disulfide bond might have significant information for protein structure and function.

Cheng *et al.* [8] defined the disulfide bond prediction problem as the classification problem of four different levels. First, a protein may have several chains. Researchers may want to know which protein chains contain disulfide bond and it is called *chain classification*. Second, a chain containing disulfide bond does not mean all cysteines are oxidized in this chain. Thus the *state classification* problem is to decide whether a cysteine is oxidized or reduced in the chain. And third, given a pair of cysteines, the *bond classification* problem is to predict whether they are bonded together. Finally, given all cysteines, the *connectivity prediction* problem is to predict the corresponding cysteines for each bonded pair. In this paper, we focus on the study of the cysteine state (disulfide bonding state) classification problem.

Several algorithms have been proposed for the state classification problem. Martelli *et al.* [14] proposed the method based on the hidden Markov model (HMM) [16] and the neural networks (NN) [12], [19]. Chen *et al.* [7] applied the *support vector machine* (SVM) [11], [6] and an adjustment method, called *cysteine state sequence* method, to predict the cysteine state. In addition, some studies applied the the multi-classifier to deal with this problem [5], [18].

In this paper, we develop an algorithm to improve the accuracy of cysteine state classification.

*This research work was partially supported by the National Science Council of Taiwan under contract NSC98-2221-E-110-062.

†Corresponding author: cbyang@cse.nsysu.edu.tw

978-1-4244-6439-5/10/$26.00 ©2010 IEEE

This algorithm adopts the multi-stage framework of multi-classifier, and it gets good result. We test our algorithm on dataset PDB4136, and the result of our algorithm achieves 94.2% accuracy in cysteine state prediction. It improves 3.5% accuracy compared with the previous best result 90.7% [10].

The rest of this paper is organized as follows. We will introduce some preliminary knowledge in Section II. And next, Section III presents our algorithm in detail. Section IV shows our experimental results and gives the comparison with some previous results. Finally, we give the conclusion of this paper in Section V.

II. Preliminaries

This section introduces some preliminary knowledge used by in this paper, including the position specific scoring matrix (PSSM), the support vector machine (SVM) and two related algorithms, the two-stage system [5] and the simple tune method [10]. Our algorithm is based on the latter two algorithms.

A. Position Specific Scoring Matrix

The *position specific scoring matrix* (PSSM), proposed by Altschul *et al.* [3], is a particular scoring matrix which is used in *basic local alignment search tool* (BLAST) [2] and *position-specific iterated-BLAST* (PSI-BLAST) [3]. Given a primary (target) protein sequence, BLAST can answer its similar sequences based on the scoring matrix. PSI-BLAST uses the result produced by BLAST to create the PSSM. Then PSI-BLAST uses the PSSM as the input to iterate BLAST. Hence, PSI-BLAST is more sensitive than BLAST in searching distant homologous sequences.

The scores of PSSM represent the occurrence probabilities over background probability. If some residues are more significant in homologous sequences of the target sequence, their scores will be higher than others in each row. Table I shows a partial PSSM of the sequence with PDBID `1AHL`, whose original length is 49.

Chuang *et al.* [9] studied the relationship between disulfide bond and protein structure, and their result shows that they are highly related. In addition, Rubinstein and Fiser [17] analyzed the disulfide bond connectivity in proteins by correlated mutations. It shows that the probability of the two cysteines of a disulfide bond simultaneously mutated is 99% in homologous proteins. These studies reflect the relationship between disulfide bonds and protein structures or functions in homologous proteins. Therefore we presume that PSSM could be useful in cysteine state classification.

B. Support Vector Machine

The *support vector machine* (SVM) program was developed by Chang and Lin [6], and it is widely used in many research areas. Let $\mathbf{x_i}$ be the feature vector extracted from data element i and y_i be the class label of i, where each $\mathbf{x_i}$ is in the d-dimensional space. The main function of SVM is to compute the hyperplane for splitting these feature vectors ($\mathbf{x_i}$) into two clusters consistent with their label (y_i). With this hyperplane, we can predict and classify one unknown target data element into its class. For more general applications, these feature vectors may not be linearly separable. The SVM solves this problem by providing different *kernel functions* to transform the original feature space into higher dimensional space, or using the *soft margin* which allows few data elements being classified into a wrong class.

C. The Two-stage System

The two-stage system for predicting cysteine states was proposed by Ceroni *et al.* [5]. This algorithm performs different classification in each stage. In the first stage, it classifies all proteins into three types, including *none*, *mix*, and *all*, which correspond to that the cysteine states of a protein are *all reduced* , *some oxidized* and *all oxidized*, respectively. In the second stage, it predicts each cysteine state in mixed proteins by SVM. In addition, the cysteine states of none or all are clearly to be set as all reduced or all oxidized, respectively. In summary, it uses both global and local evolutionary information, which are the features extracted from whole protein and from context of multiple sequence alignment near a specific cysteine, respectively, to enhance the prediction accuracy of cysteine state. Figure 1 illustrates the concept of the two-stage system.

TABLE I
A PARTIAL PSSM OF PDBID 1AHL.

		A	R	N	D	C	Q	E	G	H	I	L	K	M	F	P	S	T	W	Y	V
1	G	0	-3	-1	-2	-3	-2	-3	6	-3	-4	-4	-2	-3	-4	-3	-1	-2	-3	-4	-4
2	V	2	-3	-3	-3	-1	-2	-2	-2	-3	3	0	-2	0	1	-2	-1	-1	-3	-1	3
3	S	2	-2	-2	-2	-2	-1	-1	-1	-2	-3	-3	-1	-2	-4	7	1	-1	-4	-3	-2
4	C	-1	-4	-4	-4	10	-4	-5	-3	-4	-2	-2	-4	-2	-3	-4	-2	-2	-3	-3	-2
5	L	0	3	-2	-3	-2	-1	-1	-3	-2	0	3	1	1	-1	-3	-2	-1	-3	-2	-1
6	C	-1	-4	-4	-4	10	-4	-5	-3	-4	-2	-2	-4	-2	-3	-4	-2	-2	-3	-3	-2
7	D	0	-2	1	6	-3	-1	1	-2	-2	-2	-3	0	-3	-4	-2	-1	-1	-5	-3	-1
8	S	0	-1	2	2	-2	0	0	-1	-1	-3	-3	-1	-2	-3	-1	5	1	-4	-2	-2
9	D	-2	-2	1	7	-4	-1	1	-2	-2	-4	-4	-1	-4	-4	-2	1	-1	-5	-4	-4
10	G	0	-3	-1	-2	-3	-2	-3	6	-3	-5	-4	-2	-3	-4	-3	-1	-2	-3	-4	-4
11	P	-1	-3	-3	-2	-4	-2	-2	-3	-3	-4	-4	-2	-3	-5	8	-1	-2	-5	-4	-3
12	S	0	0	3	3	-2	0	0	-1	2	-3	-3	0	-2	-3	-2	3	2	-4	-2	-2
13	V	0	-2	-2	-3	-1	-2	-2	-3	-3	2	0	-2	0	-1	0	0	2	-3	-2	3

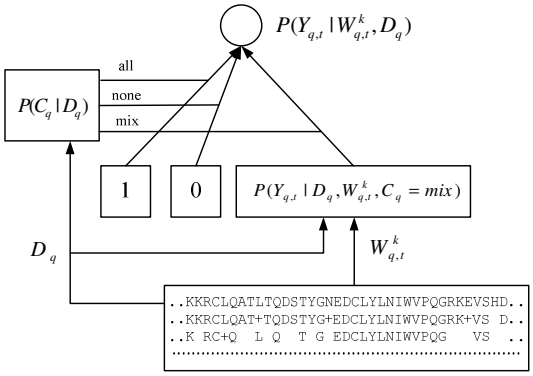

Fig. 1. The two-stage system [5].

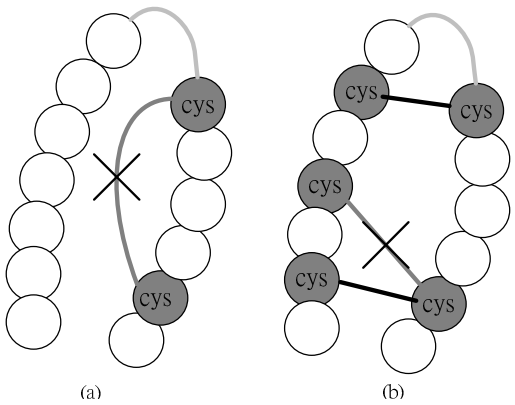

Fig. 2. The illustrations of assumptions of infeasible bonds. (a) A bond within a peptide-chain. (b) The mal-aligned bond between two peptide-chains.

D. The Simple Tune Method

In 2009, Chung *et al.* [10] proposed an algorithm using PSSM as features of SVM to predict the cysteine state. For every cysteine in a protein, this algorithm adopts features $Q_i, L, PSSM_{k,i}$. Feature Q_i is the order of the cysteine in a protein, and its normalization is done with dividing it by the number of cysteines in this protein. Variable L indicates the normalized scale of protein length, which is divided by the longest length of proteins. $PSSM_{k,i}$ means the scoring submatrix of $(2k+1)$-size window centering at cysteine i, which contains $(2k+1)\times 20$ elements. In order to normalize $PSSM_{k,i}$, their algorithm uses the minimum and maximum of every scoring matrix to guarantee that all features are of values between zero and one. Note that every protein has different minimum and maximum.

In addition, the algorithm uses the probability produced by SVM to perform the effective tuning method called *simple tune*. This simple tune method is based on two assumptions. First, it tends to be infeasible if two cysteines forming a disulfide bond in one peptide-chain locate in the same beta sheet, as shown in Figure 2(a). Second, if there are several disulfide bonds on two peptide-chains of a beta sheet, these bonds tend to align in parallel to preserve the structure, as shown in Figure 2(b). The simple tune method consists of four adjustment steps, including boundary adjustment, oxidized inversion, reduced inversion and odd-even revision. This adjustment can improve the accuracy of cysteine state prediction.

III. Our Algorithm

Chung *et al.* [10] for disulfide bond state prediction has got good result in related studies. For improving the accuracy of whole proteins for cysteine state prediction, based on the concept of the two-stage system proposed by Ceroni *et al.* [5], we develop an algorithm with specific useful features. Here the accuracy of whole proteins is measured by the number of correctly predicted proteins that all cysteines in one protein are correctly predicted.

A. Overview

The first stage of the two-stage system [5] is to classify proteins into three types none, mix and all. We call the first stage of the two-stage system as *type classification*. For improving the prediction accuracy of the two-stage system, we use various features to perform type classification. We use the output probabilities of SVM as the new features, append them to the existing features and start a new training iteration repeatedly. The detailed steps of *probability feedback* will be discussed in Section III-B. With the probability feedback, the accuracy can be improved with some fixed parameters of SVM, cost and gamma. After the type classification stage, we predict the cysteine state of a mixed protein with Chung's algorithm [10]. In addition, we add several extra features to SVM and use another normalization technique as described below. We call this stage *mix classification*. Figure 3 shows the flowchart of this algorithm.

B. Probability Feedback

The probabilities output from SVM are crucial to the subsequent prediction. We append the probabilities to the existing features in the type classification. We find that it enhance the accuracy for some fixed parameter. In other words, we append the classification probability of data element i at the end of $\mathbf{x_i}$. Then, we start a new cross-validation iteration with the same parameters, cost and gamma of SVM. This procedure is performed repeatedly until the result converges. Note the feature set grows up in every iteration. The convergent condition is that the difference of correct classification between two successive iterations is less than three and it happens twice in a row.

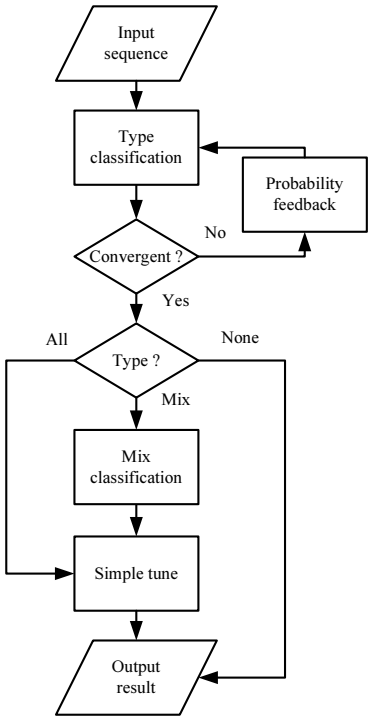

Fig. 3. The flowchart of our algorithm.

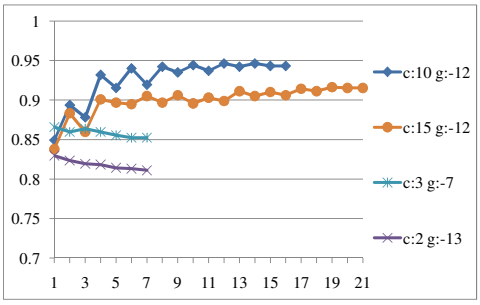

Fig. 4. The accuracies on various parameters in each iteration. The X-axis and Y-axis represent iteration and accuracy, respectively.

In most cases, these accuracies increase and decrease alternately, but the trends of accuracies rise up. However, in some cases, the trends of accuracies always decrease. Figure 4 shows four results of different parameters on each iteration. In the legend of Figure 4, c and g indicate SVM parameters cost and gamma, respectively, where the values on the right side of c and g mean the logarithms of cost and gamma to the base 2.

TABLE II
HYDROPHOBIC COEFFICIENTS [13].

R	-4.5	K	-3.9	N	-3.5	D	-3.5
Q	-3.5	E	-3.5	H	-3.2	P	-1.6
Y	-1.3	W	-0.9	S	-0.8	T	-0.7
G	-0.4	A	1.8	M	1.9	C	2.5
F	2.8	L	3.8	V	4.2	I	4.5

C. Feature Extraction of Type Classification

In the type classification, we include eleven sets of features. Let L be the length of protein, F_i be the occurrence of amino acid i in a protein, $H(i)$ be the hydrophobic coefficient of residue i (Table II), $P_{i,j}$ be the position of the jth residue i, $A(j)$ be the residue at position j, and $S_{m,n}$ be the score of row m and column n in PSSM.

1) Protein length: L.
2) Cysteine count: F_C.
3) Cysteine odd-even index: $F_C \bmod 2$.
4) Amino acid composition: The composition of residue i is $\frac{F_i}{L}$.
5) Average cysteine position: $\frac{\sum_{j=1}^{F_C} P_{C,j}}{F_C}$.
6) Average distance of every two cysteines: $\frac{\sum_{x=1}^{F_C}\sum_{y=1}^{F_C}|P_{C,x}-P_{C,y}|}{L\times F_C}$.
7) Average hydrophobicity: $\frac{\sum_{i=1}^{20} F_i\times H(i)}{L}$.
8) Average hydrophobicity around cysteine: The 2γ values of average hydrophobicity around all cysteines are defined as $\frac{\sum_{j=1}^{F_C} H(A(P_{C,j}+k))}{F_C}$ for $-\gamma \le k \le \gamma$, but $k \ne 0$. Here we set $\gamma = 7$.
9) Cysteine position distribution: For $1 \le d \le \rho$, the dth cysteine position distribution is $|\{\alpha = \frac{P_{C,j}}{L} \mid \frac{(d-1)\times L}{n} < \alpha \le \frac{(d)\times L}{n}, 1 \le j \le F_C\}|$. We set $\rho = 5$.
10) Cysteine distance distribution: For $1 \le d \le \delta$, the dth cysteine distance distribution is $|\{\beta = \frac{|P_{C,x}-P_{C,y}|}{L} \mid \frac{(d-1)\times L}{n} < \beta \le \frac{(d)\times L}{n}, 1 \le x \le y \le F_C\}|$. We set $\delta = 5$.
11) Average PSSM of amino acid: The average PSSM of residue i is $\frac{\sum_{m=1}^{L} S_{m,i}}{L}$.

D. Feature Extraction of Mix and State Classification

In the mix and state classification, the involved features are similar to the algorithm of Chung *et al.* [10]. We only add several extra features to the existing feature set, but we have different normalization. The features of cysteine j in a protein are described as follows.

1) Cysteine index: $\frac{P_{C,j}}{F_C}$.
2) Cysteine location: $\frac{P_{C,j}}{L}$.
3) Hydrophobicity around cysteine: The hydrophobicity of 2γ residues around a cysteine is $H(A(P_{C,j}+k))$, for $-\gamma \le k \le \gamma$, but $k \ne 0$. We set $\gamma = 12$.
4) PSSM around cysteine: $PSSM_{k,j}$ (see Section II-D). The k is set to be 12. We do not use the minimum and maximum to normalize this feature set.

We also add some feature sets used in the type classification, including average cysteine position, average cysteine distance, cysteine odd-even index and average hydrophobicity, to this classifier.

E. Normalization of Features

The purpose that we divide the features by their protein lengths (such as amino acid composition) or cysteine count (such as cysteine index) is to make them more significant. That is, we decrease the effect of the variance of sequence length or cysteine count.

The normalization is to make the distribution of feature values more evenly, and to make SVM separate the feature vectors more easily. Let N_j and M_j be the average and the standard deviation of the values in the jth dimension of all $\mathbf{x_i}$'s. The normalized value of each feature is $\frac{x_{i,j}-N_j}{M_j}$.

IV. EXPERIMENTAL RESULTS

The data set PDB4136, used by Martelli *et al.* [14], was got from the *PAPIA system* [15] to select non-homologous protein chains from PDB, with identity less than 25%. And this data set has also been used by many researchers.

We analyze all proteins in PDB released on March 18, 2010, which contains 61924 proteins and 149773 chains in total. As the summarization of protein types shown in Table III, the distribution of protein types in PDB is similar to PDB4136.

We perform 20-fold cross-validations on PDB4136. The k-fold cross-validation means that it splits a data set D into k pieces $D = \{d_1, d_2,$

TABLE III
SUMMARIZATION OF PDB AND DATA SET PDB4136. THE PERCENTAGE VALUES (%) ARE SHOWN IN THE PARENTHESES.

Dataset	total	no cysteine	none	mix	all
PDB protein	61924	13043	34514(0.706)	5299(0.108)	9068(0.186)
PDB chain	149773	37645	84306(0.725)	11691(0.104)	16131(0.144)
PDB4136	969	-	691(0.714)	75(0.077)	203(0.209)

$\cdots, d_k\}$, then it trains the model based on $D - d_i$ and tests the accuracy with d_i for $1 \leq i \leq k$. This procedure runs k times since each piece d_i has to be tested.

For evaluating the prediction performance, we apply the conventional measurements *specificity*, *sensitivity* and *Matthews correlation coefficient* (MCC), which are shown in the following.

$$\text{Specificity} = \frac{TP}{TP + FP}. \tag{1}$$

$$\text{Sensitivity} = \frac{TP}{TP + FN}. \tag{2}$$

$$\text{MCC} = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FN)(TP + FP)(TN + FP)(TN + FN)}}. \tag{3}$$

Besides, we use Q_c for evaluating cysteine-based performance.

$$Q_c = \frac{P_c}{N_c}. \tag{4}$$

In Equation 4, P_c means the number of cysteines with correct prediction in all test folds and N_c represents total number of cysteines. And we use Q_p for evaluating protein-based performance.

$$Q_p = \frac{P_p}{N_p}. \tag{5}$$

In Equation 5, P_p means the number of proteins whose cysteines are all correctly predicted and N_p means total number of proteins.

Table IV shows the experimental result comparison of previous studies and our algorithm. The hidden neural network (HNN) [14] is a hybrid system of hidden Markov model and neural networks. This system uses the specific state transitions to guarantee that oxidized cysteines are even. The MultipleSVM+CSS [7] is based on the cysteine state sequences (CSS). A CSS represents the transition path for all cysteine states of a protein, and it produces the average probability of transition path by MultipleSVM. The APTK+DISULFIND [18] is a method combined by all-pairs decomposition kernel and DISULFIND (a prediction web server). Finally, the Multi-phase Approach [10] was already introduced in Section II-D.

As shown in Table IV, the best previous study is the Multi-phase Approach [10], and our result has about 3.5% and 5.8% improvements in Q_c and Q_p, respectively. Besides, MCC is increased by 0.08, and both specificity and sensitivity (no matter in oxidized or reduced state) are all improved.

V. CONCLUSION

In this paper, we develop an algorithm to improve the accuracy of cysteine state classification, and we get better results than previous studies. We also propose a training strategy called probability feedback to improve the accuracy, and it gets great increase in type classification. Note that in the simple tune method, while the protein-based performance is enhanced, the cysteine-based performance always decreases after the adjustment. It shows that when features extracted from sequences, we lose some information which may be useful in simple tune. For further improving the accuracy, one may find other useful features or find the tradeoff between cysteine-based and protein-based performances.

REFERENCES

[1] V. I. Abkevich and E. I. Shakhnovich, "What can disulfide bonds tell us about protein energetics, function and folding: Simulations and bioninformatics analysis," *Journal of Molecular Biology*, vol. 300, pp. 975–985, 2000.

[2] S. F. Altschul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipmanl, "Basic local alignment search tool," *Journal of Molecular Biology*, vol. 215, no. 3, pp. 403–410, 1990.

[3] S. F. Altschul, T. L. Madden, A. A. Schaffer, J. Zhang, Z. Zhang, W. Miller, and D. J. Lipman, "Gapped BLAST and PSI-BLAST: a new generation of protein database search programs," *Nucleic Acids Research*, vol. 25, no. 17, pp. 3389–3402, 1997.

TABLE IV
ACCURACY COMPARISON OF DATA SET PDB4136 WITH PREVIOUS STUDIES.

Method	Q_C	MCC	Oxidized		Reduced		Q_p
			Specificity	Sensitivity	Specificity	Sensitivity	
HNN [14]	88.0	0.73	78.1	93.3	86.3	88.8	84.0
MultipleSVM+CSS [7]	90.0	0.77	91.0	77.0	89.0	90.0	-
APTK+DISULFIND [18]	90.3	-	82.1	89.2	-	-	-
Multi-phase Approach [10]	90.7	0.79	88.4	84.4	91.8	94.1	86.0
Our Algorithm	94.2	0.87	91.0	92.3	95.9	95.2	91.8

[4] M. K. Campbell and S. O. Farrell, *Biochemistry*, 4th ed. Thomson-Brooks/Cole, 2003.

[5] A. Ceroni, P. Frasconi, A. Passerini, and A. Vullo, "Predicting the disulfide bonding state of cysteines with combinations of kernel machines," *Journal of VLSI Signal Processing*, vol. 35, pp. 287–295, 2003.

[6] C.-C. Chang and C.-J. Lin, "LIBSVM: a library for support vector machines," 2001, software available at http://www.csie.ntu.edu.tw/∼cjlin/libsvm.

[7] Y.-C. Chen, Y.-S. Lin, C.-J. Lin, and J.-K. Hwang, "Prediction of the bonding states of cysteines using the support vector machines based on multiple feature vectors and cysteine state sequences," *Proteins: Structure, Function, and Bioinformatics*, vol. 55, pp. 1036–1042, 2004.

[8] J. Cheng, H. Saigo, and P. Baldi, "Large-scale prediction of disulphide bridges using kernel methods, two-dimensional recursive neural networks, and weighted graph matching," *Proteins: Structure, Function, and Bioinformatics*, vol. 62, pp. 617–629, 2006.

[9] C.-C. Chuang, C.-Y. Chen, J.-M. Yang, P.-C. Lyu, and J.-K. Hwang, "Relationship between protein structures and disulfide-bonding patterns," *Proteins: Structure, Function, and Bioinformatics*, vol. 53, pp. 1–5, 2003.

[10] W.-C. Chung, C.-B. Yang, and C.-Y. Hor, "An effective tuning method for cysteine state classification," in *Proc. of National Computer Symposium, Workshop on Algorithms and Bioinformatics*, Taipei, Taiwan, Nov. 27-28 2009.

[11] N. Cristianini and J. Shawe-Taylor, *An Introduction to Support Vector Machines and other kernel-based learning methods*. Cambridge University Press, 2000.

[12] K. Gurney and K. N. Gurney, *An introduction to neural networks*. MIT Press, 1995.

[13] J. Kyte and R. F. Doolittle, "A simple method for displaying the hydropathic character of a protein," *Journal of Molecular Biology*, vol. 157, pp. 105–132, 1982.

[14] P. L. Martelli, P. Fariselli, L. Malaguti, and R. Casadio, "Prediction of the disulfide-bonding state of cysteines in proteins at 88% accuracy," *Protein Science*, vol. 11, pp. 2735–2739, 2002.

[15] T. Noguchi, H. Matsuda, and Y. Akiyama, "PDB-REPRDB: a database of representative protein chains from the protein data bank (PDB)," *Nucleic Acids Research*, vol. 29, no. 1, pp. 219–220, 2001.

[16] L. R. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[17] R. Rubinstein and A. Fiser, "Predicting disulfide bond connectivity in proteins by correlated mutations analysis," *Bioinformatics*, vol. 24, no. 4, pp. 498–504, 2008.

[18] M. Vincent, A. Passerini, M. Labbe, and P. Frasconi, "A simplified approach to disulfide connectivity prediction from protein sequences," *BMC Bioinformatics*, vol. 9, no. 1, p. 20, 2008.

[19] J. Zurada, *Introduction to artificial neural systems*. St. Paul, MN, USA: West Publishing Co., 1992.