An Efficient Nearest Neighbor Method for Protein Contact Prediction

Gualberto Asencio-Cortés(✉), Jesús S. Aguilar-Ruiz, and Alfonso E. Márquez-Chamorro

School of Engineering, Pablo de Olavide University, Sevilla, Spain
{guaasecor,aguilar,amarcha}@upo.es

Abstract. A variety of approaches for protein inter-residue contact prediction have been developed in recent years. However, this problem is far from being solved yet. In this article, we present an efficient nearest neighbor (NN) approach, called PKK-PCP, and an application for the protein inter-residue contact prediction. The great strength of using this approach is its adaptability to that problem. Furthermore, our method improves considerably the efficiency with regard to other NN approaches. Our NN-based method combines parallel execution with k-d tree as search algorithm. The input data used by our algorithm is based on structural features and physico-chemical properties of amino acids besides of evolutionary information. Results obtained show better efficiency rates, in terms of time and memory consumption, than other similar approaches.

Keywords: k-nearest neighbor · k-d tree · Protein inter-residue contact prediction

1 Introduction

Protein inter-residue contact prediction has been an important and relevant topic in bioinformatics and computational biology in last decades. The prediction of inter-residue contacts represents an important previous step to solve the protein structure prediction problem (PSP). Predicting the structure of a protein from its amino acid sequence is the main key to cure those diseases which are related with the anomalous formation of proteins, *e.g.* Alzheimer. Computational methods represent a faster and more economic way to solve the PSP problem, than experimental methods, *e.g.* X-ray crystallography.

The nearest neighbor (NN) algorithm is an adequate computational approach to address the problem of protein structure prediction, due to the knowledge that proteins which share a high degree of similarity in their sequences, should have similar 3D structures. In protein contact prediction problem, NN methods find the K-closest protein sequence profiles from a database of known protein structures, according to a distance measure, and predict the corresponding class (contact or non-contact).

Previously, nearest neighbor algorithms have been presented in the PSP literature. Abu-doleh *et al.* [1] predicts contact maps using an inference system

based on a fuzzy-neural network and nearest neighbor approach. The algorithm employs 5 windows of amino acids and performs an attribute selection according to negative matrix factorization. Colubri *et al.* [7] performs a homology-based method using torsion angles. The algorithm analyses propensities of the different types of amino acids and secondary structures. Similar structures search is carried out using a simulated annealing approach. Davies *et al.* [8] implements a case based reasoning method which predicts protein contact maps. The method obtains secondary structure from the contact maps and from the geometric knowledge about alpha-helix contacts. Finally, Glasgow and Davies [10] proposes a contact map predictor using sequence data. Case representation includes protein name, protein sequence, assignment of secondary structure to residues, structure class and protein contact map. The solution consists of a 3D backbone model of the protein structure computed from the contact map.

All these algorithms constitute a part of ensemble methods. In this article we have developed a method exclusively based on a NN approach for the contact prediction. A protein contact is established according to the geometrical distance of each pair of amino acids of a sequence. If this distance is lower than a determined threshold in angstroms, a contact is defined. We highlight that the performance efficiency of our method, in terms of execution time and computer memory used, is better than other NN implementations. A parallel implementation of the algorithm and the incorporation of k-d trees [5] as search algorithm contributes to the improvement of these measures. On the other hand, we have employed as encoding features, physico-chemical properties of residues (hydrophobicity, polarity and net charge), structural features, such as secondary structure and solvent accessibility, and evolutionary information, in the form of position specific scoring matrices (PSSM). This encoding shares the assumption that the prediction of the 3D structure of a protein can be based on characteristics of the amino acids [14].

The remainder of this paper is organized as follows. Section 2 introduces our methodology. Section 3 presents the experimentation and obtained results. Finally, Sect. 4, includes some conclusions and possible future works.

2 Methods

2.1 Overview

Our prediction system, named PKK-PCP, is able to predict contacts between amino acids in protein structures from protein sequences. Our system takes, as input data, the hydrophobicity, polarity and charge of residues. It also uses as input, external predictions for secondary structure and solvent accessibility. Furthermore, evolutionary profiles (PSSM) of residues are also employed as input.

Our system returns a contact probability value for each pair of amino acids. PKK-PCP is based on the classic nearest neighbour algorithm. The system uses k-d trees to optimize the neighbour search. Moreover, our system is implemented in C++, parallelized and optimised for 64 bits architecture.

The methodology carried out to build profiles for each residue pair is detailed in Sect. 2.2. Then we explain the prediction process based on nearest neighbour

approach in Sect. 2.3. In Sect. 2.4 we detail several important implementation notes of our predictor. Finally, we define the evaluation measures used for effectiveness and efficiency in Sect. 2.5.

2.2 Profile Construction

From input protein sequences, our system takes each pair of amino acids as usual in PSP methods [1,4]. Formally, we represent an amino acid sequence of length L as $s_1 \dots s_L$. Then we consider amino acid pairs (s_i, s_j) such that $1 \leq i < j \leq L$.

For each amino acid pair, PKK-PCP builds a profile that contains 50 attributes. We define the profile for each pair (s_i, s_j) as shown in Eq. 1.

$$[H^*_{i,j}, P^*_{i,j}, SS_i, SS_j, SA_i, SA_j, PSSM_i, PSSM_j] \in \mathbb{R}^{50} \tag{1}$$

The components $H^*_{i,j}$ and $P^*_{i,j}$ measure the average of hydrophobicity and polarity between residues (i, j) and they are defined in Eq. 2, where H_i and H_j are the hydrophobicities of amino acids i and j respectively. P_i and P_j are the polarities of amino acids i and j respectively.

$$\begin{aligned} H^*_{i,j} &= \frac{H_i + H_j}{2} \in [0, 1] \\ P^*_{i,j} &= \frac{P_i + P_j}{2} \in [0, 1] \end{aligned} \tag{2}$$

We used the scale proposed by Black and Mould [6] for hydrophobicity and the scale proposed by Radzicka and Wolfenden [16] for polarity. We found that the hydrophobicity average of amino acids i and j ($H^*_{i,j}$) produces better predictions than separated hydrophobicities H_i and H_j into the profile, and it reduces the profile size. Same conclusions are applied to polarity.

We included into the profile the predicted secondary structure of amino acids i and j (SS_i and SS_j respectively). These predictions are returned by PSI-PRED [11] as commonly used by PSP methods in literature.

The SS values are encoded using three values for three secondary states: $\{0.5, 0, 0\}$ for alpha helix, $\{0, 0.5, 0\}$ for beta sheet and $\{0, 0, 0.5\}$ for random coil. Thus, the Euclidean distance between SS attributes is 0 or 1 depending whether amino acids have the same secondary structure or not, respectively. Therefore, SS_i and SS_j have 3 attributes each.

The components SA_i and SA_j of the profile are the predicted solvent accessibility of amino acids i and j respectively. For this purpose we used the predictor proposed by Rost and Sander [17] as in the work of Bacardit *et al.* [4]. We used a 5-state representation for SA, ranging from 0 to 4, where lower values mean a buried state and higher values represent exposed states.

Finally we included into the profile the evolutionary information ($PSSM_i$ and $PSSM_j$ for amino acids i and j) from PSI-BLAST [2] as widely used in PSP and bioinformatics literature. $PSSM_i$ and $PSSM_j$ have 20 attributes each.

All values in the profile are normalized between 0 and 1 in order to provide an equal contribution to distance calculation among profiles in further nearest neighbour search scheme, as we show in the next subsection.

2.3 Residue Contact Prediction

PKK-PCP begins with an initial set of proteins, it builds the profiles as we have explained in the previous subsection and divides the protein set into training and test folds according to a cross-validation scheme.

For each profile within the test fold, PKK-PCP assess the Euclidean distance between that profile and each profile in the training fold, in order to find the K training profiles with the lowest distances (most similar training profiles).

Then, the contact probability returned by our system for amino acids i and j of the test profile is calculated as the number of contacts in most similar training profiles divided by K.

In order to improve the performance, a pruning in the neighbour search is applied for all the methods in comparison. This pruning is based on the charge of amino acids i and j of the test sequence. Specifically, this means that given a test profile, if the net charge [12] of amino acid i or j is distinct to 0 (at least one amino acid is positive or negative charged), then the contact probability is 0 and the neighbour search is omitted.

This pruning is in line with the results obtained by Márquez-Chamorro *et al.* [13] in the Fig. 10. In that work the percentage of contacts between charged amino acids is close to 0 and they could be rejected. This pruning is introduced with the aim of providing a comparison using a high number of proteins in a reasonable time. Moreover, this pruning has a low impact to the effectiveness of methods in comparison, as we show in the experimentation section.

2.4 Predictor Implementation

The system PKK-PCP is implemented in Microsoft C++ 2012 using the release configuration for 64 bits and was built for multithreading architecture. PKK-PCP is based on the ANN library of David Mount [3] for nearest neighbor searching using k-d trees. We have adapted that library for parallel execution, cross validation and protein contact evaluation. That evaluation was designed to measure the protein inter-residue contact prediction, which is a classification problem with a binary class.

2.5 Effectiveness and Efficiency Evaluation

The effectiveness and efficiency of PKK-PCP are assessed using several measures. Regarding the effectiveness, we computed accuracy (Acc) and coverage (Cov), defined as shown in Eqs. 3 and 4 respectively. These measures are widely used by PSP methods in literature [1,4]. The reason is that these measures are focused in prediction of positive cases of contacts between amino acids, and these cases are quite less frequent than negative ones, as it is shown in Table 1.

$$Acc = \frac{TP}{TP + FP} \tag{3}$$

$$Cov = \frac{TP}{NumContacts} \tag{4}$$

TP are true positives and FP are false positives. $NumContacts$ are the number of real contacts. The accuracy is the ratio of predicted contacts that are presented in the native structure. The coverage is the ratio of native contacts that are predicted to be contacts.

In this work, we used a cut-off value of 8 angstroms in order to define a contact between two amino acids, which is commonly used in literature [9,19].

Regarding the efficiency, we computed the elapsed time during the prediction process and the space consumed in computer memory.

3 Experimentation

This section presents the experimentation followed and the results obtained by our method. The aim of this experimentation consists on providing an analysis of the effectiveness and efficiency of our system for different sizes of protein sets. The results have been compared with Weka IBk algorithm [18] with $K = 1$ using both linear and k-d tree search algorithms.

3.1 Datasets

Protein datasets used in our experimentation were selected from the dataset of Bacardit *et al.* [4]. This dataset is derived from PDB-REPRDB [15] and consists of 3,262 protein chains with sequence identity lower than 30 %, a resolution smaller than 2Å and a crystallographic R factor lower than 20 %.

We have randomly extracted from this dataset several subsets: DS25, DS50, DS75, DS100, DS200, DS300. We used them for our experimentation, where the number included in their names indicates the number of proteins in each subset. The aim of using incremental size of subsets is to test the efficiency of our approach in terms of time and memory consumption. Table 1 shows the main characteristics of each dataset used in the experimentation.

Table 1. Number of proteins, minimum, maximum and average length of protein sequences, number of contacts and non-contacts between residues and their ratio for each dataset employed in the experimentation.

Dataset	#	min	max	avg	contacts	non-contacts	ratio(c:nc)
DS25	25	54	405	123.08	4,871	282,692	1:58
DS50	50	54	405	119.20	11,114	646,518	1:58
DS75	75	53	405	120.77	17,852	1,167,011	1:65
DS100	100	53	405	124.02	23,750	1,497,252	1:63
DS200	200	53	602	169.78	48,809	3,075,366	1:63
DS300	300	50	822	195.53	84,203	7,122,243	1:84

Table 2. Comparative study of efficiency (time in hours and memory in megabytes) of PKK-PCP and Weka IBk algorithm using the charge-based pruning. Weka IBk uses linear search (Weka column) and k-d tree (Weka-KDT column) as search algorithms. The effectiveness (accuracy and coverage) is also shown.

Dataset	Measure	Weka	Weka-KDT	PKK-PCP
DS25	Acc	0.797	0.797	0.797
	Cov	0.689	0.689	0.689
	Time	38.12	9.31	1.07
	Mem	6259	7140	505
DS50	Acc	0.788	0.788	0.788
	Cov	0.682	0.682	0.682
	Time	312.07	29.25	3.04
	Mem	8522	10,832	935
DS75	Acc	0.798	0.798	0.798
	Cov	0.691	0.691	0.691
	Time	744.52	63.72	6.26
	Mem	9,425	11,974	1,340
DS100	Acc	0.795	0.795	0.795
	Cov	0.690	0.690	0.690
	Time	–	92.11	10.47
	Mem	10,054	12,720	1,859
DS200	Acc	0.791	0.791	0.791
	Cov	0.685	0.685	0.685
	Time	–	183.42	44.72
	Mem	13,289	16,542	4,597
DS300	Acc	0.789	0.789	0.789
	Cov	0.690	0.690	0.690
	Time	–	295.04	114.11
	Mem	17,254	20,358	7,362

3.2 Configuration

All the experiments were run on a 64-bit workstation, a Dell Precision T7400, with 2x Intel Xeon X5482 3.2 GHz (2x4 cores), 32 GB DDR2 RAM, SATA2 7200rpm HD and Windows 7 Ultimate.

As we mentioned before, our algorithm returns a contact probability for each test instance. We determine a contact if this probability is higher than 0.5. We also set the number of neighbors to $K = 1$. A 10-fold cross-validation with 5 runs per fold was applied.

To evaluate the predictions, we have obtained a coverage and accuracy value for each test protein, instead for each data point. Since Weka does not obtain

Table 3. Comparative study of efficiency (time in hours and memory in megabytes) of PKK-PCP and Weka IBk algorithm without using the charge-based pruning. Weka IBk uses linear search (Weka column) and k-d tree (Weka-KDT column) as search algorithms. The effectiveness (accuracy and coverage) is also shown.

Dataset	Measure	Weka	Weka-KDT	PKK-PCP
DS25	Acc	0.803	0.803	0.803
	Cov	0.712	0.712	0.712
	Time	54.35	13.27	1.45
	Mem	6259	7140	505
DS50	Acc	0.811	0.811	0.811
	Cov	0.709	0.709	0.709
	Time	447.21	41.33	4.27
	Mem	8522	10832	935
DS75	Acc	0.817	0.817	0.817
	Cov	0.719	0.719	0.719
	Time	1,055.64	90.68	8.81
	Mem	9,425	11,974	1,340
DS100	Acc	0.812	0.812	0.812
	Cov	0.716	0.716	0.716
	Time	–	130.85	14.45
	Mem	10,054	12,720	1,859

Table 4. Number of pruning applied for each dataset in a 10-fold cross validation and number of false negatives incurred when the pruning are applied.

Dataset	Instances	Pruning	FN
DS25	287,563	123,105(42.8 %)	7,521(6.1 %)
DS50	657,632	286,201(43.5 %)	17,085(5.9 %)
DS75	1,184,863	508,424(42.9 %)	31,624(6.2 %)
DS100	1,521,002	647,946(42.6 %)	43,477(6.7 %)
DS200	3,124,175	1,377,761(44.1 %)	84,456(6.1 %)
DS300	7,206,446	3,124,066(43.3 %)	200,252(6.4 %)

these measures for each protein, we have implemented an external evaluator in order to evaluate the Weka predictions for each test protein using exactly the same training instances as in our approach.

Furthermore, the Weka IBk algorithm (with both linear and k-d tree search) was modified to include the charge-based pruning mentioned in Sect. 2.3, in order to perform a fair comparison between PKK-PCP and Weka IBk algorithm (linear and k-d tree) in terms of efficiency.

3.3 Results

The time and memory consumed were analysed comparing PKK-PCP to Weka IBk method, using both linear and k-d tree search algorithms implemented in Weka. The effectiveness (accuracy and coverage) is also shown, but these values are the same for all the methods because they share the same training nearest neighbours and the comparison is focused in their efficiency.

As we can see in Table 2, the execution time of our method widely improves the Weka IBk k-d tree results. In the worst case (for dataset DS25), PKK-PCP used only the 11.5 % of the time employed by Weka with k-d tree (with a difference of 8.24 h). The optimized implementation of nearest neighbors approach inside PKK-PCP and its parallel execution contributes to this efficiency improvement.

Table 3 is included in order to appreciate the impact of removing the charge-based pruning to the effectiveness and efficiency of methods in comparison. Table 3 shows the accuracy, coverage, time in minutes and memory in megabytes of the three methods in comparison with no pruning in the neighbour search. Only datasets DS25, DS50, DS75 and DS100 are presented in Table 3. Note that the differences with and without pruning, in terms of times and memory consumption, are high (around 43 % faster with the charge-based pruning). However, the differences of accuracy and coverage are slight (around 0.15 of accuracy and 0.21 of coverage better without pruning). This low difference in effectiveness due to pruning is explained by the values shown in Table 4. According to values shown in Table 4, it is remarkable the low number of false negatives incurred despite the high number of pruning performed (up to 6.7 % false negatives with respect to the total of pruning, in the worst case DS100). The high number of instances to tackle, when the number of proteins on datasets increases, leads to

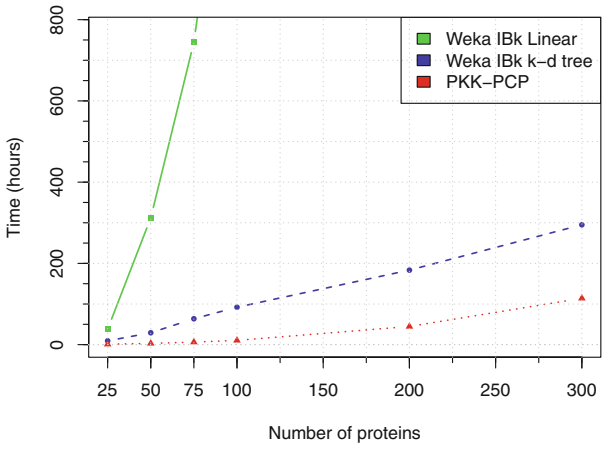

Fig. 1. Evolution of execution time (in hours) of Weka IBk with linear search, Weka IBk with k-d tree and PKK-PCP for each dataset of proteins.

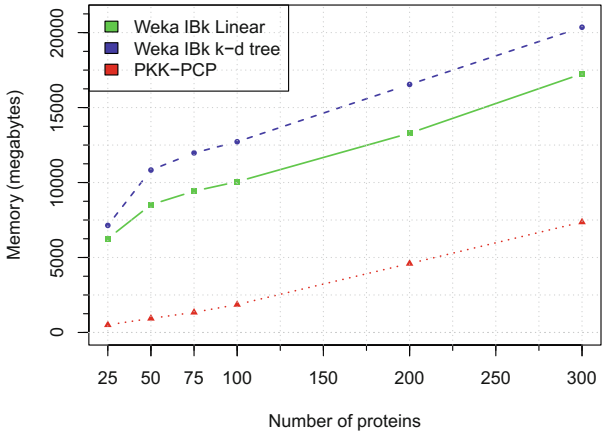

Fig. 2. Evolution of computer memory (in megabytes) used by Weka IBk with linear search, Weka IBk with k-d tree and PKK-PCP for each dataset of proteins.

using some type of pruning like we have used. In our case, the selected pruning gains an important benefit in terms of performance with a low decrement of effectiveness.

Figure 1 shows a chart which represents the execution times of PKK-PCP and Weka IBk (with linear and k-d trees search algorithms) for the different datasets. While k-d trees provides a complexity order of $NlogN$ to nearest neighbor approach, Weka IBk with linear search belongs to a quadratic order N^2. However, as we can see in Fig. 1, the execution time of our method remains in lower values than Weka k-d tree method for all studied datasets.

Figure 2 presents the memory used by PKK-PCP and Weka-IBk for different datasets. Our method obtains lower rates of memory consuming due to the optimization of the data structures we have introduced in our implementation. PKK-PCP achieved a memory consumption improvement of $10,494 \pm 1,990$ megabytes (mean and standard deviation values) with respect to Weka IBk (k-d tree) for all datasets.

4 Conclusions and Future Work

The presented work provides an efficient implementation of K-NN approach and a labelling rule for pruning nearest neighbour search specific for the protein contact prediction problem. A faster and a low memory consumption method is presented to handle a high number of proteins in a reasonable time. The use of our charge-based pruning has allowed, on the one hand, to improve considerably the efficiency, and, on the other hand, to loose the minimum degree of effectiveness.

The efficiency of our approach, in terms of execution time and memory used, was shown in comparative terms, and we have found that our system achieved

times and memory consumption much better than Weka IBk with k-d tree. This seems to be a wide improvement of efficiency in nearest neighbor approach applied to protein inter-residue contact prediction.

The low memory consumption achieved by PKK-PCP allows to include more number of attributes than classical nearest neighbors approaches for a predetermined time horizon. For that reason, as future work, we will consider the addition of two amino acid windows to represent the environments of the target residues in the profiles, and also include more physico-chemical properties of amino acids like residue volume, accessible surface area or molecular weights.

Acknowledgments. This research was supported by the Spanish MEC under project TIN2011-28956-C02-01.

References

1. Abu-doleh, A., Al-jarrah, O., Alkhateeb, A.: Protein contact map prediction using multi-stage hybrid intelligence inference systems. J. Biomed. Inf. **45**, 173–183 (2012)
2. Altschul, S., Madden, T., Schffer, A., Zhang, J., Zhang, Z., Miller, W., Lipman, D.: Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. Nucleic Acids Res. **25**(17), 3389–3402 (1997)
3. Arya, S., Mount, D., Netanyahu, N., Silverman, R., Wu, A.: An optimal algorithm for approximate nearest neighbor searching fixed dimensions. J. ACM **45**(6), 891–923 (1998)
4. Bacardit, J., Widera, P., Márquez-Chamorro, A., Divina, F., Aguilar-Ruiz, J., Krasnogor, N.: Contact map prediction using a large-scale ensemble of rule sets and the fusion of multiple predicted structural features. Bioinformatics **28**(19), 2441–2448 (2012)
5. Bentley, J.: Multidimensional binary search trees used for associative searching. Commun. ACM **18**(9), 509–517 (1975)
6. Black, S., Mould, D.: Development of hydrophobicity parameters to analyze proteins which bear post or cotranslational modifications. J. Anal. Biochem. **193**, 72–82 (1991)
7. Colubri, A., Jha, A., Shen, M., Sali, A., Berry, R., Sosnick, T., Freed, K.: Minimalist representations and the importance of nearest neighbor effects in protein folding simulations. J. Mol. Biol. **363**, 835–857 (2006)
8. Davies, J., Glasgow, J., Kuo, T.: Visio-spatial case-based reasoning: a case estudy in prediction of protein structure. Comput. Intel. **22**, 194–207 (2006)
9. Fariselli, P., Casadio, R.: A neural network based predictor of residue contacts in proteins. Protein Eng. **12**, 15–21 (1999)
10. Glasgow, J., Kuo, T., Davies, J.: Protein structure from contact maps: a case-based reasoning approach. Inf. Sys. Front **8**, 29–36 (2006)
11. Jones, D.: Protein secondary structure prediction based on position-specific scoring matrices. J. Mol. Biol. **292**, 195–202 (1999)
12. Klein, P., Kanehisa, M., DeLisi, C.: Prediction of protein function from sequence properties: discriminant analysis of a data base. J Biochim. Biophys. **787**, 221–226 (1984)

13. Márquez-Chamorro, A., Asencio-Cortés, G., Divina, F., Aguilar-Ruiz, J.: Evolutionary decision rules for predicting protein contact maps. In: Pattern Analysis and Applications, September 2012, pp. 1–13 (2012)
14. Márquez-Chamorro, A.E., Divina, F., Aguilar-Ruiz, J.S., Bacardit, J., Asencio-Cortés, G., Santiesteban-Toca, C.E.: A NSGA-II algorithm for the residue-residue contact prediction. In: Giacobini, M., Vanneschi, L., Bush, W.S. (eds.) EvoBIO 2012. LNCS, vol. 7246, pp. 234–244. Springer, Heidelberg (2012)
15. Noguchi, T., Matsuda, H., Akiyama, Y.: PDB-REPRDB: a database of representative protein chains from the protein data bank (PDB). Nucl. Acids Res. **29**(1), 219–220 (2001)
16. Radzicka, A., Wolfenden, R.: Comparing the polarities of the amino acids: side-chain distribution coefficients between the vapor phase, cyclohexane, 1-octanol, and neutral aqueous solution. J. Biochem. **27**, 1664–1670 (1988)
17. Rost, B., Sander, C.: Conservation and prediction of solvent accessibility in protein families. Proteins **20**(3), 216–26 (1994)
18. Witten, I., Frank, E., Hall, M.: Data Mining: Practical Machine Learning Tools and Techniques. Morgan Kauffman, San Francisco (2011)
19. Zhang, G., Huang, D., Quan, Z.: Combining a binary input encoding scheme with RBFNN for globulin protein inter-residue contact map prediction. Pattern Recognit. Lett. **26**, 1543–1553 (2005)