Journal of Bioinformatics and Computational Biology
Vol. 7, No. 5 (2009) 773 – 788

PROTEIN FOLD CLASSIFICATION WITH GENETIC ALGORITHMS AND FEATURE SELECTION

PENG CHEN[1], CHUNMEI LIU[2], LEGAND BURGE

Department of Systems and Computer Science, Howard University
2300 Sixth Street, NW
Washington, DC 20059, USA
[1]pchen78@scs.howard.edu, [2]chunmei@scs.howard.edu

MAHMOOD MOHAMMAD

Department of Mathematics, Howard University
2400 Sixth Street, NW
Washington, DC 20059, USA

WILLIAM SOUTHERLAND

Department of Biochemistry, Howard University
520 W Street, NW
Washington, DC 20059, USA

CLAY GLOSTER

Department of Electrical and Computer Engineering, Howard University
2300 Sixth Street, NW
Washington, DC 20059, USA

Protein fold classification is a key step to predicting protein tertiary structures. This paper proposes a novel approach based on genetic algorithms and feature selection to classifying protein folds. Our dataset is divided into a training dataset and a test dataset. Each individual for the genetic algorithms represents a selection function of the feature vectors of the training dataset. A support vector machine is applied to each individual to evaluate the fitness value (fold classification rate) of each individual. The aim of the genetic algorithms is to search for the best individual that produces the highest fold classification rate. The best individual is then applied to the feature vectors of the test dataset and a support vector machine is built to classify protein folds based on selected features. Our experimental results on Ding and Dubchak's benchmark dataset of 27-class folds show that our approach achieves an accuracy of 71.28%, which outperforms current state-of-the-art protein fold predictors.

Availability: http://mail.ustc.edu.cn/~bigeagle/ web/JBCB2009.htm

Keywords: Fold Classification; Genetic Algorithms; Support Vector Machine; Feature Selection.

1. Introduction

Protein tertiary structures play an important role in understanding the functions of the proteins. Although researchers have made huge progress in predicting protein tertiary structures, there still exist many proteins with unknown tertiary structures. Experimental efforts using X-ray and NMR technologies are generally used to predict protein tertiary

structures. However, they generally are very complicated, time consuming and expensive. Therefore, computational methods have been developed to predict protein tertiary structures. Although the details of the spatial structures of proteins are extremely complicated and irregular, their overall topological folding patterns are surprisingly simple and regular. Therefore, identifying the topological folding patterns of a protein is a very important step to predicting its tertiary structures. The folding patterns of proteins have different characteristics. Based on the characteristics, proteins are classified into a limited number of structural classes, among which, four structural classes are mostly typical: mainly α, mainly β, mixed α-β, and few secondary structures [1], or another four structural classes: all α, all β, α/β, and $\alpha+\beta$ [2, 3].

There are many learning based approaches on protein fold classification such as neural networks [4], support vector machines (SVM) [5, 6], bayesian networks [7], genetic algorithms and k-nearest neighbor (KNN) [8], KNN classifier [9], hierarchical learning architecture [10], hidden markov models [11, 12], etc. [13]. Dubchak [4, 14] proposed three descriptors with a three layer feed forward neural network to predict protein folds. The three descriptors, composition (C), transition (T), and distribution (D), describe the global composition of a given amino acid property in a protein, the frequencies that the amino acid property changes with the entire length of the protein, and the distribution pattern of the amino acid property along the sequence, respectively. Their experimental results showed that the three descriptors significantly outperform other approaches for protein fold classification. The best accuracy of their approach is up to 71.7% on a test dataset containing 83 folds.

Ding and Dubchak constructed a dataset with 27-class folds based on the SCOP PDB-40D database and divided it into a training set and a test set such that any two sequences in the training and test datasets have less than 35% sequence identity. They applied a well-known binary SVM classifier methodology on selected characteristic spaces from the dataset for protein fold classification [14]. The best accuracy of Ding's method on the dataset is 56%. Moreover, for the first time, the method presented the most predictive protein characteristic, amino-acid composition. Later, Shen and Chou [9] proposed a novel feature descriptor PseAA (pseudo-amino acid compositions) and obtained 62% classification rate using an optimized evidence-theoretic k-nearest neighbors (OET-KNN) method. Okun [15] explored a nonnegative matrix factorization (NMF) in combination with three nearest-neighbor classifiers for protein fold classification. As a result, when employed the best one out of the three classifiers and reduced the original dimensionality by around 30%, the method increased the accuracy by more than 4% than the classification rate with the original high-dimensional space.

Huang et al. [10] proposed a hierarchical learning architecture (HLA) method that classified proteins into four major classes: all alpha, all beta, alpha + beta, and alpha/beta. Then in the next level they used another set of classifiers (i.e. radial basis function network (RBFN)) to further classify the proteins into 27 folds [16]. The best accuracy of HLA method based on RBFN is up to 65.5% on Ding's benchmark dataset. In addition, Guo [8] proposed a genetic-algorithm evidence-theoretic KNNs method and finally got an accuracy of 64.7% on the same dataset.

Recently, Shamim et al. [17] developed a SVM based classifier for protein fold classification using the structural information of amino acid residues and amino acid residue pairs. The feature vector consists of predicted secondary structural state and predicted solvent accessibility state frequencies of amino acids and amino acid pairs. The classifier using secondary structural state frequencies of amino acids achieved an overall

accuracy of 65.2% for fold classification, which outperformed previous methods. The classifier using the combination of secondary structural state frequencies and solvent accessibility state frequencies of amino acids and amino acid pairs further improved the accuracy of fold classification to more than 70%, which is around 8% higher than the previous best method for fold classification. Furthermore, the study revealed that their three multi-class classification methods, namely *one versus all, one versus one* and *Crammer and Singer* method, yield similar predictions.

Damoulas and Girolami [5] proposed a probabilistic multi-class multi-kernel learning method to recognize protein folds. The method uses multiple characteristic spaces that are available, such as global characteristics like the amino-acid composition (C), predicted secondary structure (S), hydrophobicity (H), Van Der Waals volume (V), polarity (P), polarizability (Z), the PseAA [18], as well as two local alignment Smith–Waterman (SW) based characteristic spaces with different scoring matrices. It applies a single multi-class kernel machine on all of the characteristic spaces simultaneously and then combines their results. The method achieved the best accuracy of 68.1% on the benchmark dataset proposed by Ding and Dubchak [16].

In this paper, we study the problem of protein fold classification using the benchmark dataset of Ding and Dubchak [16]. We firstly obtain three improved descriptors and further apply them in protein fold classification. We develop a novel approach based on genetic algorithms and a support vector machine to determine the best feature selector. The SVM then applies the best feature selector to the feature vectors in the test dataset to classify the protein folds. Our method achieves an accuracy of 71.28%, which outperforms previous state-of-the-art methods.

2. Dataset and Methods

2.1. *Dataset preparation*

We used the benchmark dataset of Ding and Dubchak [16] except we removed proteins 2SCM_C and 2GPS from the training set and proteins 2YHX_1 and 2YHX_2 from the test set, because the four proteins do not have sequence records in the PDB databank [9, 19]. Finally, there are 311 proteins in the training dataset and 383 proteins in the test dataset. According to the SCOP database [2, 3], proteins in the training and test datasets could be further classified into the following 27-fold types [9, 14, 16]: (1) globin-like, (2) cytochrome c, (3) DNA-binding 3-helical bundle, (4) 4-helical up-and-down bundle, (5) 4-helical cytokines, (6) EF-hand, (7) immunoglobulin-like, (8) cupredoxins, (9) viral coat and capsid proteins, (10) conA-like lectin/glucanases, (11) SH3-like barrel, (12) OB-fold, (13) beta-trefoil, (14) trypsin-like serine proteases, (15) lipocalins, (16) (TIM)-barrel, (17) FAD (also NAD)-binding motif, (18) flavodoxin-like, (19) NAD(P)-binding Rossmann-fold, (20) P-loop, (21) thioredoxin-like, (22) ribonuclease H-like motif, (23) hydrolases, (24) periplasmic binding protein-like, (25) b-grasp, (26) ferredoxin-like, and (27) small inhibitors, toxins, lectins.

2.2. *Feature descriptors of amino acid sequences*

We extract characteristic spaces from the characters of residues or proteins. Some characteristic spaces from previous work are used in this work for protein fold classification, such as amino acid composition, predicted secondary structure,

hydrophobicity, polarity, van der waals volume, and polarizability [14, 16]. Additionally, we use four statistical contact potentials (SCP) [20-23], whose potential matrices can be seen in our supplementary material. Firstly, we get potential values of 20 residue types. Using the statistical contact potential matrix $P_{i,j}$ in [21], the potential value R_i of residue type i can be computed using Eq. (1). That is to say, each type of the 20 amino acids is assigned a potential value through the Eq. (1), such as -3.885 for residue A and -4.655 for residue R. The 20 amino acids are then allocated into three equally sized residue groups G1, G2, and G3 in terms of their potential values. Seven residues with larger values are classified into group G1, seven residues with smaller values are classified into group G3, and other residues are belonging to group G2. The detailed groups are shown in Fig. 1.

$$R_i = \frac{1}{20}\sum_{j=1}^{20} P_{i,j}, \quad i, j = 1, \cdots, 20 \tag{1}$$

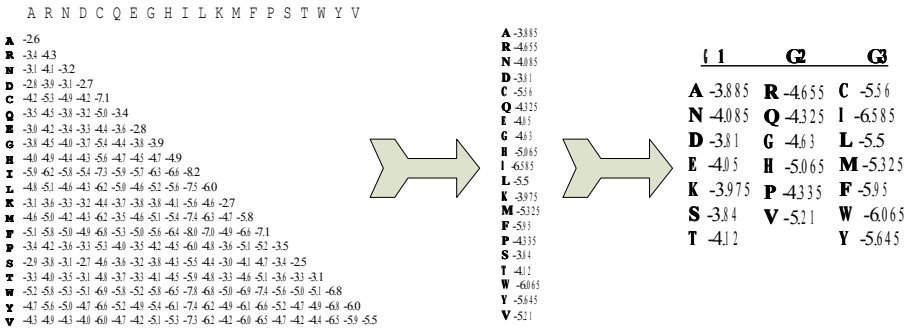

Fig. 1. Construction of residue groups G1, G2, and G3. The statistical contact potential matrix is from [21]. Residue group G1 consists of residues A, N, D, E, K, S, and T; residue group G2 consists of residues R, Q, G, H, P, and V; residue group G3 consists of residues C, I, L, M, F, W, and Y.

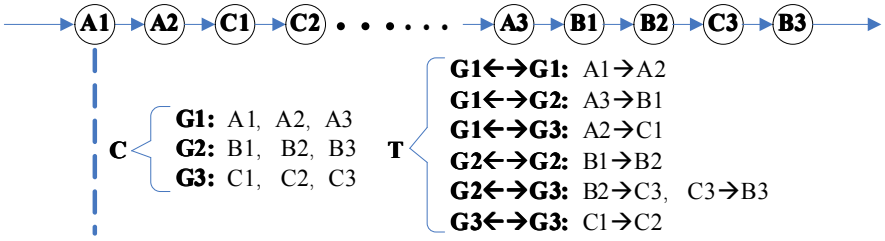

Fig. 2. Characteristic space extraction from a protein sequence. A1 denotes the first residue in residue group G1 in the sequence; B1 and C1 denotes the first residue in residue group G2 and G3 in the sequence, respectively; the same standard is used for other notations.

We extract composition (C) and transition (T) descriptors as discussed in [14]. However, different from the previous T descriptor, we add self-transition status to the same residue group, i.e. G1↔G1, G2↔G2, and G3↔G3. We calculate the number of elements of each residue group in the processes of composition and transition. For instance, the number of elements of residue group G1 in the C descriptor is 3 and the number of elements of G2↔G3 in the T descriptor is 2. As shown in Fig. 2, the C

descriptor has three feature groups, G1, G2, and G3, while the T descriptor has six feature groups, G1↔G1, G1↔G2, G1↔G3, G2↔G2, G2↔G3, and G3↔G3.

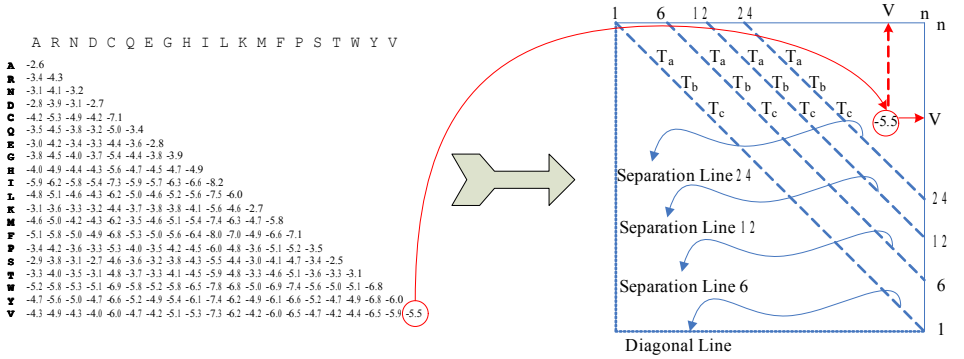

Fig. 3. Illustration of T descriptor for residue-residue pairs along a protein sequence. The left side is a statistical contact potential matrix [21], where score of each residue pair x1→x2 is equal to that of x2→x1. The right side describes how a statistical contact potential matrix transforms to residue features.

Another descriptor, distribution (D), represents the transformation from a statistical contact potential matrix to residue features. Each D descriptor is described as following. First, we define three score groups T_a, T_b, and T_c of residue-residue pairs based on the potential scores. For instance, score group T_a may consist of scores of residue-residue pair V-V and other pairs. Score group T_b, similar to the residue group G2 described above, contains residue-residue pairs with medium scores in the statistical contact potential matrix, while T_a and T_c contain residue pairs with large scores and small scores, respectively. As shown in Fig. 3, for a protein sequence with n residues, an $n \times n$ residue-residue matrix can be constructed from the statistical contact potential matrix. For example, residue-residue pair V↔V (red circle in the right side of Fig. 3) is the same as V↔V (red circle in the left side of Fig. 3) in the statistical contact potential matrix but its value is replaced with group T_b if V↔V belongs to T_b. Then we calculate the number of each group that falls into a region between two separation lines. There are four separation lines, separation lines 24, 12, 6, and the diagonal line. These four lines divide the upper-right region into four regions. The same as the concepts in the inter-residue contact problem [24], the numbers 6, 12, 24 denote the distances of residue pair in a protein sequence. The region between the separation line 6 and the diagonal line is called a local region. Similarly, the regions between the separation lines 6 and 12, the regions between the separation lines 12 and 24, the regions above the separation line 24 are called a short-range region, a medium-range region, and a long-range region, respectively. Therefore, D descriptor has 3 (groups)×4 (regions)=12 features.

There are 125 features in the characteristic spaces in Ding's method [15]. In addition, the characteristic spaces constructed based on the above description has (3(C)+6(T)+12(D))×4=84 features, where 4 is the number of selected statistical contact potentials. Finally, each protein in the training dataset and test dataset is represented with a vector of (84+125)=209 features. Such representation of protein features is then applied to the classification of protein fold types.

2.3. *Evaluation of the performance of the classifier*

We used the criteria of accuracy (*Acc*) to evaluate the performance of our classifier. *Acc* is defined as follows:

$$Acc=\frac{TP}{TP+FP} \tag{2}$$

where *TP* denotes the number of true positives, i.e., a protein is assigned to a fold type and it truly belongs to the fold type; *FP* denotes the number of false positives, i.e., protein is assigned to a fold type but in fact it doesn't belong to the fold type.

2.4. *Approach*

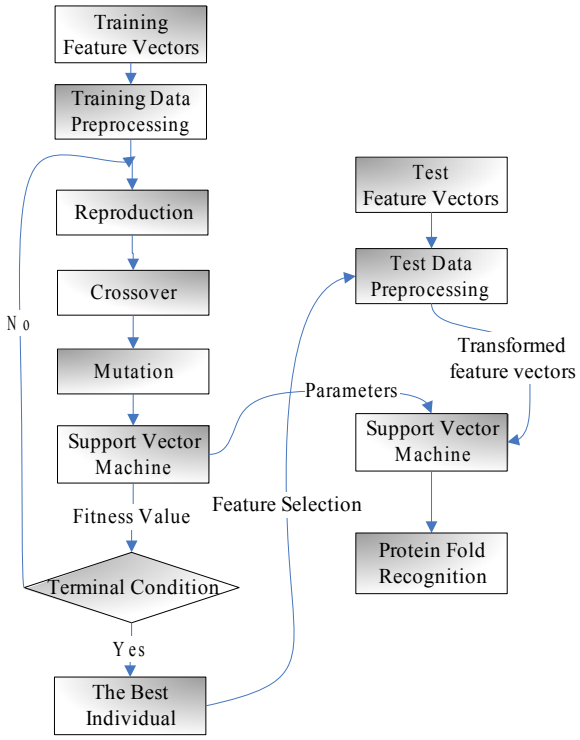

Fig. 4. Flowchart of our approach to classify protein folds.

As discussed in the previous section, each protein in the dataset is represented as a vector of features. We first encode each feature vector of the training dataset into a string. Each individual (or chromosome) of our genetic algorithms is a vector of the same size as each feature vector and each element of an individual is called a *transformer*, which will be explained later. 40 individuals compose the initial population of our genetic algorithms. Individuals of top 10% fitness values are selected to the next generation directly, while the others will go through the crossover or mutation procedure based on the preset crossover and mutation probabilities. A support vector machine is used to evaluate the fitness value of each individual. After a number of generations, the genetic algorithms terminate and we obtain the best individual which is considered to be the best feature selector for protein fold classification. Test data preprocessing then applies the best

feature selector to the feature vectors in the test dataset to obtain transformed feature vectors. Furthermore, another SVM applies the transformed feature vectors to classify protein folds. The procedure of our approach is shown in Fig. 4.

2.4.1 *Chromosome encoding*

Genetic algorithms (GA) [26] are adaptive heuristic search algorithms, which have been commonly used for optimization problems in searching for local optimal solutions within a solution space. Specifically, genetic algorithms have been applied in solving many problems in bioinformatics, such as the classification of inter-residue contacts [24, 27], protein structure alignment [28], and protein folding simulation [29]. The technique simulates Darwinian evolution by maintaining a population of solutions based on a fitness function, and searching for an individual with the maximum or minimum fitness value in the population. Individuals in the population are encoded with strings and are evaluated by a fitness function for their capability of surviving to the next generation during the evolutionary process.

In our protein fold classification study, we let V be a feature space set V= (v_1, v_2,…,v_m), where v_i is a feature variable and m is the dimension of feature vectors in the dataset. Each protein in the dataset is represented as a feature vector of V. We want to train a SVM based classifier that can correctly classify the feature vectors into K classes $C_1, C_2, \ldots, C_K$. Our goal is to search for an optimal feature selector T that maximizes the classification rate based on the corresponding selected features. To obtain the optimal feature selector T, GAs are applied to search through the space of feature transformers with a fitness function. Firstly, a vector of the feature space V is represented as a chromosome string S_t. A chromosome is composed of three kinds of *transformers* represented by characters *a*, *b*, and *c*, and the size of a chromosome is the same as a feature vector.

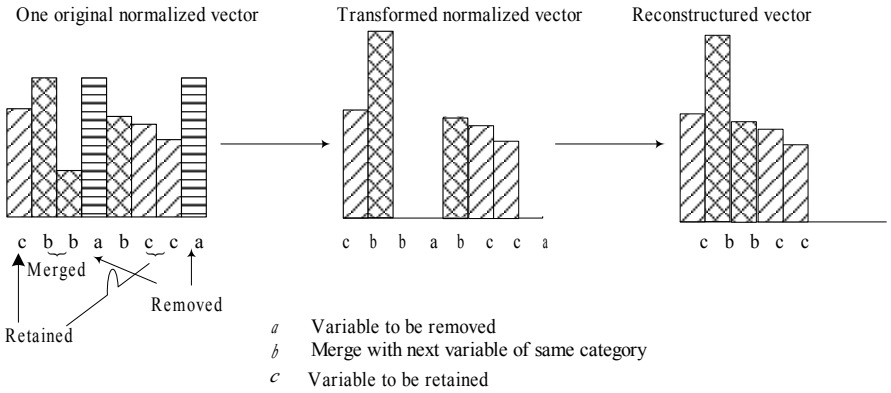

Fig. 5. Selection process of a sample feature vector of dimension 8. A bar in the histogram in the vector denotes a feature and the height of the bar corresponds to the magnitude of the feature. The transformed vector should also be normalized.

The schema for chromosome encoding is as follows: ① Character *a* in a chromosome indicates that the value in the corresponding position in all feature vectors in V will be removed; ② Two consecutive *b*'s indicate that the values in the corresponding positions

will merged together; ③ Character *c* indicates that the values in the corresponding positions will remain in the feature vectors. For instance, for a feature vector of 8 dimensions, its corresponding chromosome is a string of 8 characters from the ternary alphabet {*a*, *b*, *c*}. Fig. 5 illustrates the selection process for a feature vector (x_1,x_2,x_3,x_4,x_5,x_6,x_7,x_8). In Fig. 5, the corresponding chromosome is '*cbbabcca*'. After being applied the transformers, the elements of the sample feature vector that remain or are merged are concatenated and normalized to form a new vector of five dimensions. The new normalized vector will be used for protein fold classification.

2.4.2 *Fitness function*

SVMs have been proved to have excellent performance in many real-world applications such as text categorization, hand-written character classification, image classification, inter-residue contacts prediction, and biological sequence analysis. SVMs can handle large feature spaces and condense the information from training datasets using support vectors [30]. They demonstrate high prediction accuracy while avoid over-fitting. In this paper, we use a SVM to calculate the fitness values of individuals in genetic algorithms. In the training process, a feature vector will be assigned a value from 1 to 27 which correspond to 27 protein fold types. A support vector machine is applied to evaluate the individuals. The SVM computes the classification rate of each individual with a 2-fold cross-validation strategy. We first randomly divide the training dataset into two subsets of equal sizes. We then train the SVM using one subset and then apply the SVM to the other subset to classify proteins fold types. We then exchange the two subsets and conduct the same procedure again. The fitness value of an individual is assigned to be the average of the two classification rates obtained from the above procedure. After the genetic algorithms terminate, the individual with the highest fitness value will be considered as the best feature selector. The original feature vectors will then be transformed according to the best feature selector and then be normalized for subsequent steps.

3. Results and Discussions

In our genetic algorithms, we use parameters that have the reputation of ensuring the steadiness and rapidity of the underlying training algorithms. In particular, the possibility of crossover is set to be 0.7, while the possibility of mutation is 0.005. The population is set to contain 40 individuals and the selection procedure uses the roulette wheel algorithm.

In the process of genetic optimization, transformed vectors can be obtained by being normalized after our removing and merging some features from the original feature vectors, which dramatically reduce the computational complexity while still maintain the input information. Each individual corresponds to a discard ratio, which is the ratio of the number of removed or merged feature variables to that of total feature variables. For instance, a discard ratio of 42.58% for an individual is illustrated in Fig. 6. Fig. 6 (a) denotes the original normalized feature vector of 209 dimensions while Fig. 6 (b) illustrates its normalized transformed vector after being applied the transformers of the individual. There are 68 features removed and 21 features merged together.

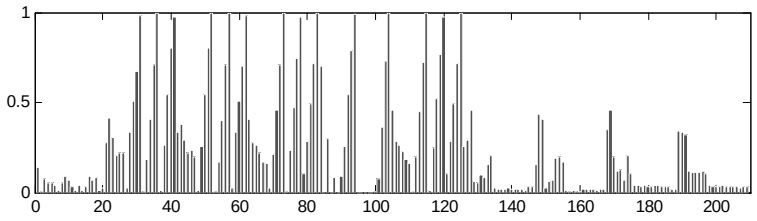

Fig. 6. (a) An original normalized feature vector. (b) The normalized transformed feature vector.

Table 1. Accuracy comparison with other methods

Charactertics Space	GA /SVM	VBKC	Ding	Shen & Chou
Amino acid composition (C)	52.74	51.2±0.5	44.9	
Predicted secondary structure (S)	38.38	38.1±0.3	35.6	
Hydrophobicity (H)	34.2	32.5±0.4	36.5	
Polarity (P)	36.29	32.2±0.3	32.9	
Van Der Waals volume (V)	42.56	32.8±0.3	35	
Polarizability (Z)	37.08	33.2±0.4	32.9	
CSHPVZ	62.52	58.6±1.1	53.9	
CSHPVZ$\lambda^1\lambda^4\lambda^{14}\lambda^{30}$	63.19	63.5	—	62.1
CSHPVZ$\lambda^1\lambda^4\lambda^{14}\lambda^{30}SW_1SW_2$	—	68.1±1.2	—	—
CSHPVZ + SCP*	71.28	—	—	—
Final Accuracy	71.28	68.1±1.2	56	62.1

—Not used in the corresponding method.
*SCP stands for the characteristic spaces extracted from the four statistical contact potentials.

Table 1 shows the accuracy comparison of our approach with some other methods. Using the six common attributes such as amino acid composition, predicted secondary structure, hydrophobicity, polarity, van der waals volume, and polarizability, our classifier has a better performance compared with VBKC and Ding's methods [5, 16]. Our characteristic space consists of the above six characteristic spaces together with another four new characteristic spaces as listed in Table 1. VBKC and Ding methods achieve their best performance using the characteristic space of amino acid composition

alone than using other characteristic spaces. VBKC method used two other characteristic spaces PseAA plus SW with BLOSUM62 and PAM50. Our classifier achieves the best performance accuracy of 71.28% using feature selection in the same characteristic space. It also outperforms Shen and Chou's ensemble classifier [9], which achieves an accuracy of 62.1% using CSHPVZ characteristic spaces and PseAA characteristic spaces.

Similar to almost all previous methods, our method using the amino acid composition achieves better performance than using other individual characteristic spaces, such as polarity and polarizability. VBKC method achieves its best performance when the local characteristic (SW) with BLOSUM62 is applied in the prediction.

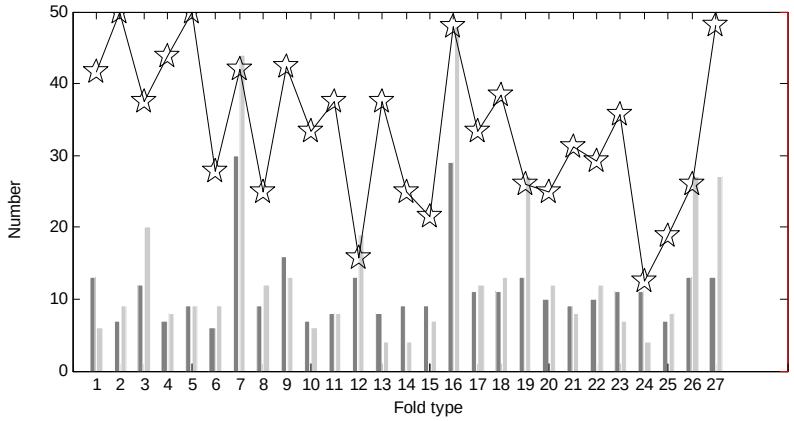

Fig. 7. The number of proteins of each fold type vs. prediction accuracy (pentagram). The left bars denote the number proteins of each protein fold type in the training dataset, while the right bars denotes the number of proteins of each protein type in the test dataset.

The detailed classification performance with respect to each protein fold is shown in Fig. 7. We also demonstrate the classification performance with respect to the number of each protein fold in the training set and the test set, respectively. Fold types 2 and 5 with small scales with 100% classification accuracy means that such scales can be used to train our classifier and achieve the best accuracy. Fold types 2 and 5 only contain 7 and 6 proteins in the training set, respectively, and each of them contains 9 proteins in the test set. However, fold type 24 containing more proteins in the training set but making the worst classification performance suggests that our classifier cannot be applied to classify these proteins. The possible reason for it could be that the number of proteins in the fold type is still small for obtaining a satisfying classification. Fold type 12, similar to fold type 24, also makes a bad classification performance at an accuracy of about 31.58%. Protein fold 7 and 16 contain 30 and 29 proteins in the training dataset, respectively. Both of the two types belong to the beta structural class [3]. Our classifier achieves better performances on the two fold types due to their containing more proteins compared to other fold types. Therefore, our classifier achieves better performance on a training fold type with more proteins. In fact, our classifier achieves 73.4% accuracy if it is trained on the test dataset (having 383 proteins) and tested on the training dataset (only 311 proteins). Such a little progress in accuracy may be due to that the test dataset contains more proteins than the training dataset so that if the two datasets are switched, the

classifier can be trained more sufficiently. However, too large size of a training dataset may cause over-fitting. Therefore, having a training dataset of a proper size is the requirement for a good classifier.

Furthermore, folds 16 and 27 achieve accuracies higher than 90%.

Table 2. Accuracy comparison of the four structural classes

	N_{Fold}	$N_{Training}$	N_{Test}	$N_{correct\ classified}$	Accuracy
α	6	54	61	50	81.97%
β	10	138	165	124	75.15%
α/β	8	86	95	56	58.95%
α+β	3	33	62	43	69.35%
Total	27	311	383	273	71.28%

*N_{Fold} denotes the number of fold types, while $N_{Training}$, N_{Test}, and $N_{correct\ predicted}$ stand for the number of proteins in the training dataset, the number of proteins in the test dataset, and the number of correctly classified proteins in the test dataset, respectively.

All the proteins in training and test datasets are classified into four structural classes, alpha, beta, alpha/beta, and alpha+beta [2, 3]. In detail, fold types from 1 to 6 belong to alpha structural class, while fold types from 7 to 16, 17 to 24, and 25 to 27 belong to beta, alpha/beta, alpha+beta classes, respectively. Table 2 shows the accuracy comparison of the four structural classes. Alpha class has the best accuracy of about 81.97%, while the alpha/beta class has the worst accuracy of about 58.95%. With a small scale of the training dataset, however, alpha+beta class gives a neutral performance. Moreover, similar to the discussion above, our classifier trained on the test dataset and tested on the training dataset with respect to alpha+beta class achieves a better performance with an increased accuracy of around 2.5%.

To perform K-fold cross-validation experiments, we combine the training and test datasets. The combined dataset consists of 694 proteins that are also divided into 27-classes fold types. Fold type 13 contains the least number of proteins with only 12 proteins. As we know, if there are much more positive instances than negative instances in a dataset, there is a chance that a given fold may not contain any negative instances. Therefore, we let *K* range from 2 to 12. Fig. 8 shows experiment results of K-fold cross-validation tests. In K-fold cross-validation, the original sample is partitioned into *K* subsamples. Of the *K* subsamples, a single subsample is used as the validation data for testing the model, and the remaining *K*-1 subsamples are used as training data. The cross-validation process is then repeated *K* times (the number of folds), with each of the *K* subsamples being used exactly once as the validation data. The *K* results from the folds then can be averaged (or otherwise combined) to produce a single estimation. We can see from Fig. 8 that our classifier achieves the best performance when *K* is 9. Actually, our tests using cross-validation intervals from 9-12 folds are available and yield better performance than on 2-8 folds. Moreover, using a proper size of the training dataset is significant to train a classifier as discussed above. For instance, among the 11 cross-validation tests, the test based on fold type 9 shows the best prediction accuracy, about 72.27%, while the test based on fold type 2 shows the worst performance.

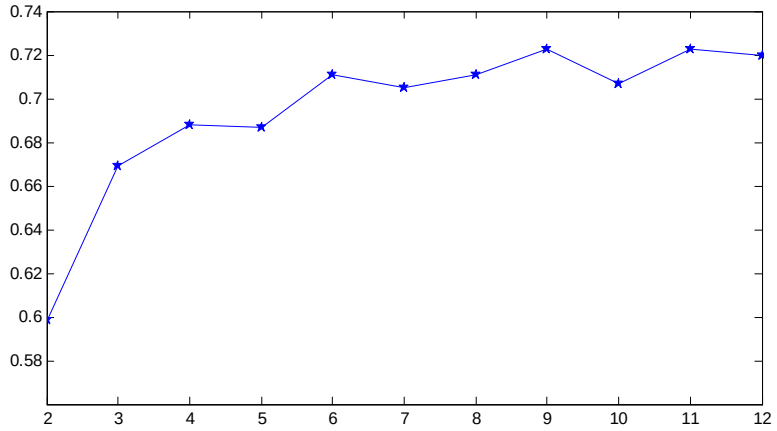

Fig. 8. The accuracy of K-fold cross-validation tests using different cross-validation intervals.

4. Conclusions

Actually, feature selection in our method is useful for achieving significant features to classify protein folds. However, the preprocess of feature selection would take extra time to achieve key features although the significant features may speed up the subsequent classification of protein folds. In addition, similar to other previous works, it is too difficult to predict all of the more than 1000 protein folds in SCOP database. More importantly, some folds even contain only one protein chain, such as 1kr7a, 2iy5a1, and 1ivsa1 and so on. Therefore, we adopt the dataset of 27 folds which is used as a benchmark for evaluating protein fold classification. Actually, it is not indicated in our work that the prediction accuracy would be decreasing when more fold families are added to the selected dataset due to no related works to be compared with. Therefore, it is always a very difficult task to overcome and will be our future work we should do.

In this paper, we have investigated protein fold classification using genetic algorithms and new feature descriptors on the benchmark dataset proposed by Ding and Dubchak. Our experimental results show that not all of the features play significant roles in protein fold classification. Adopting some features will worsen rather than improve the accuracy of protein fold classification. Therefore, a feature selection scheme is necessary to select significant features. We provide a hybrid algorithm based on genetic algorithms and SVM for feature selection. Our studies demonstrate that combining amino acid composition and statistical contact potentials work best for protein fold classification. As a result, the newly developed GA/SVM based approach presented in this paper achieves an accuracy of 71.28% on the dataset of 27-class fold types, which outperforms previous state-of-the-art predictors. Therefore, our approach can be used to extract significant features from selected feature spaces and can be further used to reconstruct protein three-dimensional structure. Supplementary materials can be found at: http://mail.ustc.edu.cn/~bigeagle/ web/JBCB2009.htm.

Acknowledgment

This work was supported in part by grant 2 G12 RR003048 from the RCMI Program, Division of Research Infrastructure, National Center for Research Resources, NIH and

the Mordecai Wyatt Johnson program, Howard University.

References

1. Orengo, C.A., Michie, A.D., Jones, D.T., Swindells, M.B., and Thornton, J.M., "CATH: A Hierarchic Classification of Protein Domain Structures", *Structure*, Vol. 5, pp. 1093-1108 (1997).
2. Andreeva, A., Howorth, D., Chandonia, J.-M., Brenner, S.-E., Hubbard, T.-J.-P., Chothia, C., and Murzin, A.-G., "Data growth and its impact on the SCOP database: new developments", *Nucl. Acid Res.*, 36, pp. D419-D425 (2008).
3. Murzin, A.G., Brenner, S.E., Hubbard, T., and Chothia, C.: 'SCOP: a structural classification of proteins database for the investigation of sequences and structures', *J. Mol. Biol.*, 247, pp. 536-540 (1995).
4. Dubchak, I., Muchnik, I., Mayor, C., Dralyuk, I., and Kim, S.-H.: 'Recognition of a protein fold in the context of the SCOP classification', *Proteins: Structure, Function, and Genetics*, 1999, 35, (4), pp. 401-407.
5. Damoulas, T., and Girolami, M.A.: 'Probabilistic multi-class multi-kernel learning: on protein fold recognition and remote homology detection', *Bioinformatics*, 2008 24, (10), pp. 1264-1270.
6. Markowetz, F., Edler, L., and Vingron, M.: 'Support Vector Machines for Protein Fold Class Prediction ', *Biometrical Journal*, 2003, 45 (3), pp. 377 - 389.
7. Raval, A., Ghahramani, Z., and Wild, D.L.: 'A Bayesian network model for protein fold and remote homologue recognition ', *Bioinformatics* 2002, 18 (6), pp. 788-801.
8. Guo, X., and Gao, X.: 'A novel hierarchical ensemble classifier for protein fold recognition', *Protein Engineering Design and Selection*, 2008 0, (gzn045), pp. v1-6.
9. Shen, H.-B., and Chou, K.-C.: 'Ensemble classifier for protein fold pattern recognition', *Bioinformatics*, 2006, 22, pp. 1717-1722.
10. Huang, C.-D., Lin, C.-T., and Pal, N.R.: 'Hierarchical Learning Architecture With Automatic Feature Selection for Multiclass Protein Fold Classification', *IEEE Transactions on Nanobioscience*, 2003 2, (4), pp. 221-232.
11. Jeanette Hargbo, A.E.: 'Hidden Markov models that use predicted secondary structures for fold recognition', *Proteins: Structure, Function, and Genetics*, 1999, 36, (2), pp. 68-76.
12. Karplus, K., Barrett, C., and Hughey, R.: 'Hidden Markov models for detecting remote protein homologies', *Bioinformatics*, 1998 14, pp. 846-856.
13. Chi, P.-H., Shyu, C.-R., and Xu, D.: 'A fast SCOP fold classification system using content-based E-Predict algorithm', *BMC Bioinformatics*, 2006, 7, pp. 362.
14. Dubchak, I., Muchnik, I., Holbrook, S.R., and Kim, S.H.: 'Prediction of protein folding class using global description of amino acid sequence', *PNAS*, 1995, 92 (19), pp. 8700-8704.
15. Okun, O., and Priisalu, H.: 'Fast Nonnegative Matrix Factorization and Its Application for Protein Fold Recognition', *EURASIP Journal on Applied Signal Processing*, 2006, pp. 1-8.
16. Ding, C.H.Q., and Dubchak, I.: 'Multi-class protein fold recognition using support vector machines and neural networks', *Bioinformatics*, 2001, 17 (4), pp. 349-358.
17. Shamim, M.T.A., Anwaruddin, M., and Nagarajaram, H.A.: 'Support Vector Machine-based classification of protein folds using the structural properties of amino acid residues and amino acid residue pairs', *Bioinformatics*, 2007, 23, (24), pp. 3320-3327.
18. Chou, K.: 'Using amphiphilic pseudo-amino acid composition to predict enzyme subfamily classes', *Bioinformatics*, 2005, 21, pp. 10-19.
19. Berman, H.M., Westbrook, J., Feng, Z., Gilliland, G., Bhat, T.N., Weissig, H., Shindyalov, I.N., and Bourne, P.E.: ' The Protein Data Bank', *Nucleic Acids Research*, 2000, (28), pp. 235-242.
20. Vendruscolo, M., and E., D.: 'Pairwise contact potentials are unsuitable for protein folding', *J. Chem. Phys.*, 1998, 109, pp. 11101-11108.

21. Tanaka, S., and Scheraga, H.A.: 'Medium- and long-range interaction parameters between amino acids for predicting three-dimensional structures of proteins', *Macromolecules*, 1976, 9, pp. 945-950.
22. Thomas, P.D., and Dill, K.A.: 'An iterative method for extracting energy-like quantities from protein structures', *Proc. Natl. Acad. Sci.*, 1996, 93, pp. 11628-11633.
23. Simons, K.T., Ruczinski, I., Kooperberg, C., Fox, B.A., Bystroff, C., and Baker, D.: ' Improved recognition of native-like protein structures using a combination of sequence-dependent and sequence-independent features of proteins', *Proteins*, 1999, 34, pp. 82-95.
24. Chen, P., Wang, B., Wong, H.-S., and Huang, D.-S.: 'Prediction of Long-range Contacts from Sequence Profile', *International Joint Conference on Neural Networks*, 2007, pp. 938-943.
25. Fischer, D., Rychlewski, L., Dunbrack, R.L., Ortiz, A.R., and Elofsson, A.: 'CAFASP3: the third critical assessmentof fully automated structure prediction methods', *Proteins: Structure,Function, and Genetics*, 2003, 53, pp. 503-516
26. Goldberg, D.E.: 'Genetic Algorithms in Search Optimization and Machine Learning' Addison-Wesley, 1989.
27. MacCallum, R.M.: 'Striped sheets and protein contact prediction', *Bioinformatics*, 2004, 20, (Suppl. 1), pp. 224-231.
28. Szustakowski, J.D., and Weng, Z.: 'Protein structure alignment using a genetic algorithm ', *Proteins: Structure, Function, and Bioinformatics*, 38 (4), pp. 428 - 440 (2000).
29. Cui, Y., Chen, R.S., and Wong, W.H.: 'Protein folding simulation with genetic algorithm and supersecondary structure constraints ', *Proteins: Structure, Function, and Bioinformatics*, 31 (3), pp. 247 - 257 (1998).
30. Hua, S., and Sun, Z.: 'A novel method of protein secondary structure prediction with high segment overlap measure: support vector machine approach', *J. Mol. Biol.*, 308, pp. 397-407 (2001).

*Electronic version of an article published as [**Journal of Bioinformatics and Computational Biology**, Volume 7, Issue 5, 2009, Pages: 773-788] [Article DOI: **10.1142/S0219720009004321**] © [copyright World Scientific Publishing Company] [Journal URL: http://www.worldscinet.com/jbcb/]*