

Predicting subcellular location of proteins using integrated-algorithm method

Yu-Dong Cai · Lin Lu · Lei Chen · Jian-Feng He

Received: 29 April 2009 / Accepted: 11 July 2009 / Published online: 7 August 2009
© Springer Science+Business Media B.V. 2009

Abstract Protein's subcellular location, which indicates where a protein resides in a cell, is an important characteristic of protein. Correctly assigning proteins to their subcellular locations would be of great help to the prediction of proteins' function, genome annotation, and drug design. Yet, in spite of great technical advance in the past decades, it is still time-consuming and laborious to experimentally determine protein subcellular locations on a high throughput scale. Hence, four integrated-algorithm methods were developed to fulfill such high throughput prediction in this article. Two data sets taken from the literature (Chou and Elrod, Protein Eng 12:107–118, 1999) were used as training set and test set, which consisted of 2,391 and 2,598 proteins, respectively. Amino acid composition was applied to represent the protein sequences. The jackknife cross-validation was used to test the training set. The final best integrated-algorithm predictor was

constructed by integrating 10 algorithms in Weka (a software tool for tackling data mining tasks, <http://www.cs.waikato.ac.nz/ml/weka/>) based on an mRMR (Minimum Redundancy Maximum Relevance, <http://research.janelia.org/peng/proj/mRMR/>) method. It can achieve correct rate of 77.83 and 80.56% for the training set and test set, respectively, which is better than all of the 60 algorithms collected in Weka. This predicting software is available upon request.

Keywords mRMR (Minimum redundancy maximum relevance) · Subcellular localization · Amino acid composition · Integrated-algorithm method · Weka

Introduction

Subcellular location is a key functional characteristic of protein. A protein cannot execute its biological function if it is not located in the proper cellular compartment [2]. Correct subcellular localization is of great significance to the functional analysis of proteins. Yet, as an enormous amount of protein sequences are being produced by genome sequencing projects, it is time-consuming and costly to determine subcellular location entirely by biochemical experimental tests. Therefore, various data mining methods have been developed to realize rapid prediction of proteins' subcellular location in the recent decades, including Support Vector Machine [3], Hidden Markov Model [4], and Neural Network [5]. Although the validity of these previous studies has been proven, most of the algorithms previous used are restricted to only some kind of machine learning algorithms. In this article, we investigated machine learning algorithms as much as possible, instead of one or several algorithms. First, we introduced Weka [6] (<http://www.cs.waikato.ac.nz/~ml/weka/>), a software which collects a great variety of machine learning

Electronic supplementary material The online version of this article (doi:10.1007/s11030-009-9182-4) contains supplementary material, which is available to authorized users.

Yu-Dong Cai and Lin Lu are contribute equally to this work.

Y.-D. Cai (✉)
Institute of System Biology, Shanghai University,
99 ShangDa Road, 200244 Shanghai, China
e-mail: cyd@picb.ac.cn

Y.-D. Cai
Centre for Computational Systems Biology, Fudan University,
220 HanDan Road, 200433 Shanghai, China

L. Lu · J.-F. He
Department of Biomedical Engineering, Shanghai Jiao Tong
University, 200040 Shanghai, China

L. Chen
Shanghai Key Laboratory of Trustworthy Computing,
East China Normal University, 200062 Shanghai, China

algorithms for data mining tasks and has been proven to be useful for bioinformatics [7–9]. In the Weka version 3.4 there are 60 machine learning algorithms. In order to take full advantages of these algorithms, four schemes were developed to integrate algorithms into one predictor. They are Majority Voting with Non-weight (MVNW), Majority Voting with Weight (MVW), Majority Voting with Non-weight plus mRMR Selection (MVNWS), and Majority Voting with Weight plus mRMR Selection (MVWS). The mRMR (Minimum Redundancy Maximum Relevance) method used here is a kind of feature selection method, which was developed by Peng [10] in 2005 (<http://research.janelia.org/peng/proj/mRMR/>).

Materials and methods

Data set

The data set we studied are taken from Chou [1], which consists of a training set and a test set. There are 2,319 and 2,598 proteins in the training set and test set, respectively. The subcellular location of the total 4,917 proteins are classified into 12 protein subcellular location [1], namely, chloroplast, cytoplasm, cytoskeleton, endoplasmic reticulum, extra cell, Golgi apparatus, lysosome, mitochondria, nucleus, peroxisome, plasma membrane and vacuole, which cover almost all the organelles in an animal or plant cell. Amino acid composition is applied to represent the protein sequences. Tested by the jackknife cross-validation, 40 algorithms collected in Weka with correct rate on training set higher than 50% were selected to perform the integrated-algorithm schemes. Finally, the MVWS integrated-algorithm scheme was proven to be the best, which can achieve correct rate of 77.83 and 80.56% for the training set and test set, respectively. We

expect that the MVWS integrated-algorithm predictor could expedite the determination of protein subcellular locations and be useful for approaching other bioinformatics problems. The distribution of subcellular locations is listed in Table 1. In order to enable being processed by computer programs, all the protein sequences were transformed into numeric vectors according to their amino acid compositions. The numeric vector V of a protein is defined as follows:

$$V = [f(x_1), f(x_2), \dots, f(x_{20})];$$

$$f(x_i) = 100 \times \frac{x_i}{N} \quad (i = 1, 2, \dots, 20).$$

where x_i is the number of amino acid i that occurred in the protein and N is the length of the protein. $f(x)$ is actually a function to calculate the frequency of amino acid i within the protein. All the numeric vectors can be found in the supplement materials I.

Jackknife cross-validation

Jackknife cross-validation is recognized as the most objective way to evaluate the performance of predictors in statistical prediction [11]. Each numeric vectors in the data set will be singled out one by one and tested by prediction model trained with all the left samples.

Machine learning algorithms in Weka

Weka [6], developed by the University of Waikato in New Zealand, collects a variety of state-of-the-art machine learning algorithms. In the Weka version 3.4 used in this study, there are 60 machine learning algorithms. Tested by the jackknife cross-validation, 40 algorithms with correct rate of training set higher than 50% were selected to perform

Table 1 Distribution of proteins in 12 subcellular location

Subcellular location	Training set	Test set
Chloroplast	154	119
Cytoplasm	592	786
Cytoskeleton	37	19
Endoplasmic reticulum	3	108
Extra cell	230	101
Golgi apparatus	26	4
Lysosome	38	31
Mitochondria	86	165
Nucleus	288	431
Peroxisome	32	24
Plasma membrane	758	803
Vacuole	25	7
Total number	2,319	2,598

the following integrated-algorithm schemes. Their names are listed below:

BayesNet, ComplementNaiveBaye, NaiveBayes, NaiveBayesMultinomial, NaiveBayesSimple, NaiveBayesUpdatable, Logistic, MultilayerPerceptron, SimpleLogistic, SMO, IB1, IBK, Kstar, BFTree, J48, J48graft, NBTree, RandomForest, REPTree, SimpleCart, DecisionTable, Jrip, PART, AttributeSelectedClassifier, Bagging, ClassificationViaClustering, ClassificationViaRegression, Dagging, Decorate, END, EnsembleSelection, FilteredClassifier, LogitBoost, MultiClassClassifier, RacedIncrementalLogitBoost, RandomCommittee, RandomSubSpace, ClassBalancedND, DataNearBalancedND, ND

See supplement materials II for the details of the 40 algorithms.

Algorithm integration scheme

In this section, four integrated-algorithm schemes and mRMR method will be illustrated in details. They are Majority Voting with Non-weight (MVNW), Majority Voting with Weight (MVW), Majority Voting with Non-weight plus mRMR Selection (MVNWS), and Majority Voting with Weight plus mRMR Selection (MVWS). The mRMR method used here is a kind of feature selection method, which was developed by Peng [10] in 2005 (<http://research.janelia.org/peng/proj/mRMR/>).

Majority voting with non-weight (MVNW)

The majority voting is one of representative integrating methods that can combine the outputs of multiple predictors. One merit of majority voting is that it does not require any previous knowledge or any additional complex computation [12]. Let $1, 2, \dots, 12$ denote the 12 protein subcellular locations. For each protein, there are 40 prediction outputs. (The prediction outputs for each protein can be found at supplement materials III.) Let $\text{predict}_j \in \{1, 2, \dots, 12\}$ ($1 \leq j \leq 40$) denote the predicted class by j th algorithm and χ_i ($1 = 1, 2, \dots, 12$) be a function from $\{1, 2, \dots, 12\}$ to $\{0, 1\}$:

$$\chi_i(a) = \begin{cases} 1 & a = i \\ 0 & a \neq i \end{cases}$$

For each protein, let $s_i = \sum_{j=1}^{40} \chi_i(\text{predict}_j)$ ($i = 1, 2, \dots, 12$) and $s = \max_{1 \leq i \leq 12} \{s_i\}$. Take $k \in \{1, 2, \dots, 12\}$, then $s_k = s$. That is, the output of the integrated-algorithm predictor for the test protein is the same as the class denoted by k . If $s = s_k = s_l$ and $k \neq l \in \{1, 2, \dots, 12\}$, then one of them will be taken arbitrarily.

Majority voting with weight (MVW)

In the MVNW, we suppose all algorithms are equal (i.e., the weight on every algorithm is identical). Yet, usually, the performance of some algorithms is better than the others. Hence, assigning different weights can help to balance the contributions made by different algorithms. In this study, the weight for an individual algorithm is the correct rate on training set tested by jackknife cross-validation.

Use the same notation as MVNW method and let w_j denote the weight of the j th algorithm. For every protein, let $s_i = \sum_{j=1}^{40} w_j \bullet \chi_i(\text{predict}_j)$ ($i = 1, 2, \dots, 12$), and $s = \max_{1 \leq i \leq 12} \{s_i\}$. In the same way as MVNW, let $k \in \{1, 2, \dots, 12\}$, then $s_k = s$. If $s = s_k = s_l$ and $k \neq l \in \{1, 2, \dots, 12\}$, then one of them will be taken arbitrarily.

Minimum redundancy maximum relevance

The Minimum Redundancy Maximum Relevance method (mRMR) [10] was developed by Peng for microarray data processing. All the input data would be transformed into numeric vectors. Intuitively, each element within the vector makes different contribution to classification task. When treating an element as a feature, the mRMR method can be used as a tool for feature pre-evaluation, that is, rank features according to their relevance with target variable and redundancy with other features. A “good” feature is characterized by maximal relevance with target variable and minimal redundancy with other features. No matter the relevance and the redundancy, they are both measured by mutual information (MI) defined as follows:

$$I(x, y) = \int \int p(x, y) \log \frac{p(x, y)}{p(x)p(y)} dx dy \quad (1)$$

where, x and y are two randomly given variables, $p(x, y)$ is their joint probabilistic density, and $p(x)$ and $p(y)$ are their marginal probabilistic density respectively.

The detailed description of mRMR is given as follows. Ω represents the feature pool, Ω_C represents the non-selected feature set with n features, while Ω_S represents the selected feature set with m features, and c represents the target variable.

Relevance, D , can be computed by Eq. 2

$$D = I(f_{\text{candidate}}, c). \quad (2)$$

Redundancy, R , can be computed by Eq. 3

$$R = \frac{1}{m} \sum_{f_i \in \Omega_S} I(f_{\text{candidate}}, f_i) \quad (3)$$

Integrate Eqs. 2 and 3 into Eq. 4 to maximize relevance and minimize redundancy:

$$\max_{f_j \in \Omega_c} \left[I(f_j, c) - \frac{1}{m} \sum_{f_i \in \Omega_s} I(f_j, f_i) \right] \quad (j = 1, 2, \dots, n) \quad (4)$$

For a feature pool with N ($N = n + m$) features, feature pre-evaluation will continue N rounds. After that, mRMR program will provide users with a feature set S as the result of pre-evaluation:

$$S = \{f_0, f_1, \dots, f_h \dots f_{N-1}\} \quad (5)$$

where the subscript h denotes at which round the feature is selected. The earlier the feature is selected into S , the better the feature is. For example, f_a is believed to be better than f_b , if $a < b$.

In this study, mRMR method was used to pre-evaluate the 40 algorithms, that is, we treat each algorithm as a feature. The feature vectors studied by mRMR method consist of the prediction results of the 40 algorithms. They are all 40-dimension vectors (see supplement materials III for details).

Majority voting with non-weight plus mRMR selection (MVNWS)

With the help of the mRMR method, the 40 algorithms could be ranked according to their relevance with classification variable and redundancy with the other algorithms. Intuitively, we should choose the algorithms with high rank to construct the integrated predictor. However, how many algorithms should be chosen is not known. Therefore, we add algorithms to the integrated predictor according to their orders in $S = \{f_0, f_1, \dots, f_h \dots f_{N-1}\}$ (see Eq. 5). The integrated predictor would be tested by jackknife cross-validation each time when adding a new algorithm. Finally, that integrated predictor with the highest correct rate would be the final predictor.

After the above processing, the algorithms selected to construct the integrated predictor are determined. The means to integrate the selected algorithms into predictor is the same as MVNW.

Majority voting with weight plus mRMR selection (MVWS)

The way to determine “good” algorithms to construct the integrated predictor is the same as MVNWS. The means to integrate the selected algorithms is the same as MVWS.

Results

Prediction results of the 40 algorithms

The Weka software tool used in this study was Weka version 3.4 (<http://www.cs.waikato.ac.nz/ml/weka/>). All the 60 algorithms in Weka 3.4 were tested by jackknife cross-validation on the training set with default parameters. Forty algorithms with correct rate higher than 50% were selected to perform the following integrated-algorithm schemes. The details of their outputs on the training set (2,319 proteins) and the test set (2,598 proteins) can be found in supplement materials III and IV, respectively. Their correct rates are listed in Table 2. From Table 2, we can see that the IB1 and IBK are the best algorithms among the 40 individual algorithms. They achieve correct rate 73.95 and 77.98% on training set and test set, respectively.

Correct rates of MVNW and MVW

The correct rates of MVNW and MVW are presented in Table 3. From Tables 2 and 3, we can see that, when using majority voting, no matter with or without weight, the performance of integrated predictor was worse than using the best individual algorithm (IB1 or IBK).

The deficiency of solely using majority voting lay in the fact that majority voting was easily biased by algorithms with great similarity. Because there was not enough pre-knowledge about the 40 algorithms, it was unavoidable that some algorithms of them with great similarity would be included into the voting system. Thus, those algorithms with great similarity might have excessive influence on the voting result. For example, the first six algorithms (BayesNet, ComplementNaiveBayes, NaiveBayes, NaiveBayesMultinomial, NaiveBayesSimple, NaiveBayesUpdateable) all had great relation with Bayes. Their prediction results were often identical. Therefore, the final majority voting might somewhat incline toward the result of Bayes.

Result of mRMR selection

Owing to the consideration of the bias brought in by algorithms with great similarity, we introduced mRMR method [10] to rank algorithms. We expected that similar algorithms could be excluded from the voting system and the bias could be reduced as much as possible while maintaining the performance of the integrated predictor.

The mRMR program used in this contribution was downloaded from website <http://research.janelia.org/peng/proj/mRMR/>. The input into mRMR program was the prediction

Table 2 Performance of the 40 algorithms

No	Algorithm	Training set (jackknife test) (%)	Test set (%)
1	BayesNet	63.52	65.78
2	ComplementNaiveBayes	61.79	69.25
3	NaiveBayes	63.13	61.93
4	NaiveBayesMultinomial	65.80	68.32
5	NaiveBayesSimple	63.17	63.13
6	NaiveBayesUpdateable	63.13	61.93
7	Logistic	69.17	70.94
8	MultilayerPerceptron	69.43	71.44
9	SimpleLogistic	69.38	71.02
10	SMO	66.41	69.78
11	IB1	73.95	77.98
12	IBk	73.95	77.98
13	Kstar	67.05	72.75
14	BFTree	59.25	65.13
15	J48	57.31	60.28
16	J48graft	59.72	61.97
17	NBTree	63.48	66.01
18	RandomForest	70.20	73.75
19	REPTree	59.08	61.35
20	SimpleCart	61.75	64.59
21	DecisionTable	54.25	56.04
22	Jrip	60.80	60.47
23	PART	60.93	62.39
24	AttributeSelectedClassifier	58.69	61.39
25	Bagging	68.18	70.40
26	ClassificationViaClustering	50.41	54.08
27	ClassificationViaRegression	67.70	71.32
28	Dagging	63.69	67.51
29	Decorate	67.10	69.13
30	END	71.37	74.02
31	EnsembleSelection	65.07	67.13
32	FilteredClassifier	56.10	62.78
33	LogitBoost	64.98	66.17
34	MultiClassClassifier	67.44	71.17
35	RacedIncrementalLogitBoost	55.15	56.43
36	RandomCommittee	69.00	72.98
37	RandomSubSpace	65.50	69.63
38	ClassBalancedND	58.00	58.35
39	DataNearBalancedND	58.21	60.05
40	ND	57.09	61.01

results of the 40 algorithms (see supplemental materials III for details). As all of the input vectors were integral vectors, we specified parameter $t = 0$ in mRMR program to tackle the integral classification problem.

Results of MVNWS and MVWS

From Table 4, we could attain the ranks of algorithms. We added algorithms one by one to the integrated predictor

Table 3 Performance of MVNW and MVW

Data set	MVNW (%)	MVW (%)
Training set	71.71	71.92
Test set	75.64	76.10

according to their ranks. The integrated predictor would be tested by jackknife cross-validation on training set each time when a new algorithm was added. Therefore, we could determine the number of algorithms we needed. Figure 1 indicates that the integrated predictor constructed by the MVNWS method can achieve the highest correct rate of 77.05% on the training set and the integrated predictor constructed by the MVWS method can achieve the highest correct rate of 77.83%. Figure 1 also pointed out that, given the same number of algorithms, the integrated predictor constructed by the MVWS method was always better than that constructed by the MVNWS method.

Discussion

Comparison between predictors

Table 5 shows the performance of five predictors. The correct rate on training set is computed by jackknife cross-validation, while the correct rate on test set is computed by the prediction model trained by the training set. Table 5 indicated two points. First, the MVWS and MVNWS methods were better than the others, meaning that the bias brought in by the similarity between algorithms had been somewhat reduced. According to [6], the initial 40 algorithms belong to six cat-

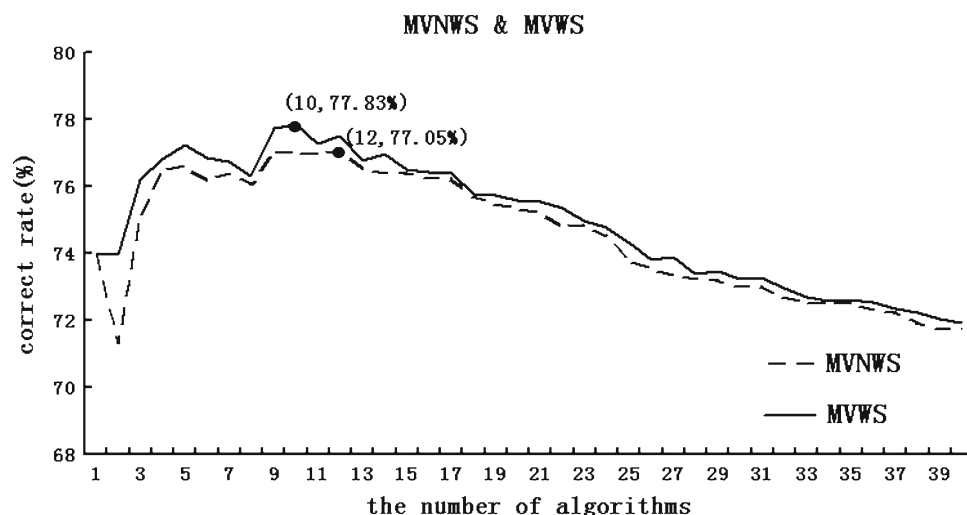
egories, that is, Bayes, functions, lazy, trees, rules, and meta (see <http://www.cs.waikato.ac.nz/ml/weka/>). Bayes algorithms apply Bayes' theorem to learn the underlying probability distribution of the data set. Functions algorithms can be written down as mathematical equation in a reasonably natural way. Lazy algorithms store the training instances and do no real work until classification time. Tree algorithms implement many decision tree methods. Rule algorithms classify data set by generating rules. Meta algorithms take biased classifiers and turn them into more powerful learners.

The distribution of the 40 algorithms can be found in Fig. 2. Owing to the quantitative unbalance between different categories, if all of the 40 algorithms were integrated into the voting system, then the voting result might be biased by the categories with large number. Fortunately, after the ranking procedure with the help of mRMR method, some algorithms with great similarity were excluded (also see Fig. 2). The number of algorithms within each category became close, and the quantitative unbalance was greatly weakened. Furthermore, some category with bad performance was excluded, however. For example, all the algorithms belonging to Bayes category were excluded, because even the best algorithm within Bayes category (NaiveBayesUpdateable) could not exceed correct rate of 66% on training set (see Table 2).

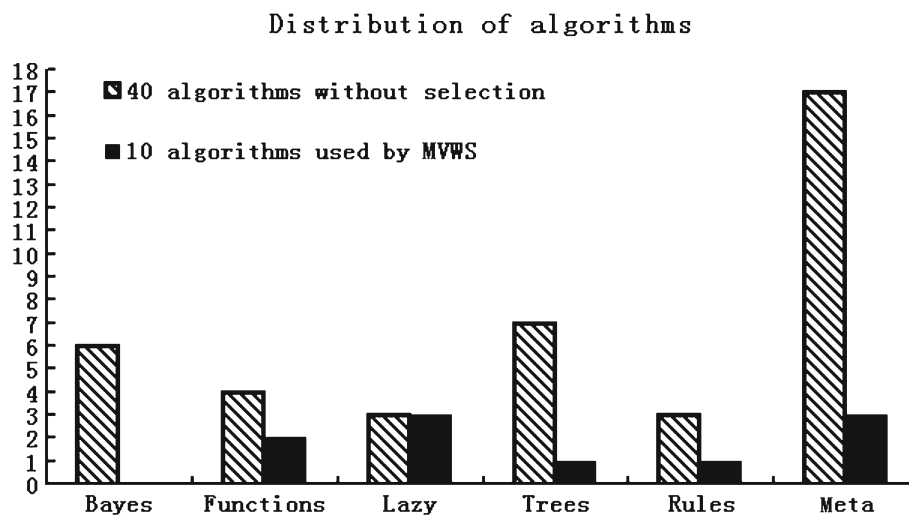
Tables 2 and 5, and Fig. 2 proved that, with regard to prediction results of algorithms as features, the mRMR method was also capable of efficiently ranking the algorithms, though it was usually applied as a tool for feature selection. Maximizing relevance was corresponding to keeping the algorithms with good performance and excluding algorithms with bad performance, while minimizing redundancy was corresponding to excluding algorithms with great similarity.

Table 4 Rank of the 40 algorithms

Rank	Algorithms	Rank	Algorithms	Rank	Algorithms
1	IB1	15	ClassBalancedND	29	REPTree
2	SimpleLogistic	16	Logistic	30	MultiClassClassifier
3	Decorate	17	J48	31	J48graft
4	Kstar	18	Bagging	32	Dagging
5	RandomForest	19	ND	33	BFTree
6	MultilayerPerceptron	20	DataNearBalancedND	34	EnsembleSelection
7	RandomCommittee	21	NaiveBayesMultinomial	35	NaiveBayesSimple
8	END	22	FilteredClassifier	36	ComplementNaiveBayes
9	IBk	23	RandomSubSpace	37	RacedIncrementalLogitBoost
10	PART	24	ClassificationViaClustering	38	BayesNet
11	ClassificationViaRegression	25	SMO	39	DecisionTable
12	AttributeSelectedClassifier	26	LogitBoost	40	NBTree
13	NaiveBayes	27	SimpleCart		
14	Jrip	28	NaiveBayesUpdateable		

Fig. 1 The testing correct rates of MVNWS and MVWS**Table 5** Performance of five predictors

Predictor	Correct rate	
	Training set (%)	Test set (%)
The best individual algorithm in Weka	73.95	77.98
MVNW	71.71	75.94
MVW	71.92	76.10
MVNWS	77.05	79.41
MVWS	77.83	80.56

Fig. 2 Distribution of the 40 algorithms without selection and the 10 algorithms used by MVWS

Second, majority voting with weight is better than majority voting without weight, no matter whether using mRMR method or not. Obviously, voting with equal rights could not reflect the differentiation of performance among different algorithms. In order to attain an excellent integrated predictor, big weight should be assigned to algorithms with good performance, while small weight should be assigned to algorithms with bad performance. Assigning different weights could be considered as an efficient way to compensate differentiation of performance among algorithms.

Amino acid composition

From the good results obtained by our method, amino acids composition was really related to protein subcellular localization, which was consistent with the biology analysis from the reference [13]. It was known to us that different types of proteins usually required distinct amino acids compositions in response to specific physiological functions. Owing to the heterogeneous distribution of proteins, amino acids composition is heterogeneous too. For example, in nucleus,

chromosomes contained large amounts of histones which were rich in histidine. Therefore, histidine enrichment might be regarded as an effective feature for nucleus localization.

Conclusion

Correct subcellular localization is of great significance to the functional analysis of proteins. In order to realize high throughput reliable prediction of subcellular location, four integrated-algorithm methods were developed to combine algorithms introduced from Weka into integrated predictor in this study. Among them, the MVWS scheme based on mRMR method was proven to be the best, which could exclude algorithms with great similarity while keeping those algorithms with good performance. We expect that the MVWS method would be also useful for other bioinformatics problems. The predicting software is available upon request.

References

1. Chou KC, Elrod DW (1999) Protein subcellular location prediction. *Protein Eng* 12:107–118
2. Eisenhaber F, Bork P (1998) Wanted: subcellular localization of proteins based on sequence. *Trends Cell Biol* 8:169–170
3. Hua S, Sun Z (2001) Support vector machine approach for protein subcellular localization prediction. *Bioinformatics* 17:721–728
4. Yuan Z (1999) Prediction of protein subcellular locations using Markov chain models. *FEBS Lett* 451:23–26
5. Reinhardt A, Hubbard T (1998) Using neural networks for prediction of the subcellular location of proteins. *Nucleic Acids Res* 26:2230–2236
6. Frank E, Witten IH (2005) Data mining: practical machine learning tools and techniques. Morgan Kaufmann, San Francisco
7. Gewehr JE, Szugat M, Zimmer R (2007) BioWeka—extending the Weka framework for bioinformatics. *Bioinformatics* 23:651–653
8. Gonzalez-Diaz H, Agüero-Chapin G, Varona J, Molina R, Delogu G, Santana L, Uriarte E, Podda G (2007) 2D-RNA-coupling numbers: a new computational chemistry approach to link secondary structure topology with biological function. *J Comput Chem* 28:1049–1056
9. Munteanu CR, Gonzalez-Diaz H, Magalhaes AL (2008) Enzymes/non-enzymes classification model complexity based on composition, sequence, 3D and topological indices. *J Theor Biol* 254:476–482
10. Peng HC, Long FH, Ding C (2005) Feature selection based on mutual information: criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Trans Pattern Analysis Mach Intell* 27:1226–1238
11. Cai YD, Chou KC (2006) Predicting membrane protein type by functional domain composition and pseudo-amino acid composition. *J Theor Biol* 238:395–400
12. Won HH, Kim MJ, Kim S, Kim JW (2008) EnsemPro: an ensemble approach to predicting transcription start sites in human genomic DNA sequences. *Genomics* 91:259–266
13. Cedano J, Aloy P, Perez-Pons JA, Querol E (1997) Relation between amino acid composition and cellular location of proteins. *J Mol Biol* 266:594–600