International Journal of Neural Systems, Vol. 20, No. 1 (2010) 13–28

DOI: 10.1142/S0129065710002206

PROTEIN SUBCELLULAR MULTI-LOCALIZATION PREDICTION USING A MIN-MAX MODULAR SUPPORT VECTOR MACHINE

YANG YANG*
Department of Computer Science and Engineering
Shanghai Maritime University
1550 Haigang Ave., Shanghai, 201306, China
yangyang@shmtu.edu.cn

BAO-LIANG LU[†,‡,§,¶]
[†]*Department of Computer Science and Engineering*
Shanghai Jiao Tong University,
[‡]*Laboratory for Computational Biology*
Shanghai Center for Systems Biomedicine,
[§]*MOE — Microsoft Key Laboratory for*
Intelligent Computing and Intelligent Systems,
800 Dong Chuan Road, Shanghai, 200240, China
bllu@sjtu.edu.cn

Prediction of protein subcellular localization is an important issue in computational biology because it provides important clues for the characterization of protein functions. Currently, much research has been dedicated to developing automatic prediction tools. Most, however, focus on mono-locational proteins, i.e., they assume that proteins exist in only one location. It should be noted that many proteins bear multi-locational characteristics and carry out crucial functions in biological processes. This work aims to develop a general pattern classifier for predicting multiple subcellular locations of proteins. We use an ensemble classifier, called the min-max modular support vector machine (M^3-SVM), to solve protein subcellular multi-localization problems; and, propose a module decomposition method based on gene ontology (GO) semantic information for M^3-SVM. The amino acid composition with secondary structure and solvent accessibility information is adopted to represent features of protein sequences. We apply our method to two multi-locational protein data sets. The M^3-SVMs show higher accuracy and efficiency than traditional SVMs using the same feature vectors. And the GO decomposition also helps to improve prediction accuracy. Moreover, our method has a much higher rate of accuracy than existing subcellular localization predictors in predicting protein multi-localization.

Keywords: Protein subcellular multi-localization; min-max modular; support vector machine.

1. Background

In order to carry out their cellular functions, proteins need to be in the right compartments. Identification of subcellular location can provide information helpful for understanding protein function, regulation and protein-protein interaction. And, efficient computational tools can save costly and laborious wet-lab experiments. Therefore, prediction of protein subcellular localization has been an active research topic in bioinformatics in the last decade.[1–5]

*This work was mainly done while this author was a PhD student at the Department of Computer Science and Engineering, Shanghai Jiao Tong University.
¶Corresponding author.

To develop automatic tools for subcellular localization, machine learning methods, such as neural networks,[6] hidden Markov models (HMMs)[7] and support vector machines (SVMs),[4,8,9] have been widely used thanks to the abundance of proteins with known locations in the public databases. In recent years, some freely accessible web-servers have been developed. Especially, Chou and Shen have published a series of web-servers, namely Hum-Ploc,[10] Gpos-Ploc,[11] Virus-Ploc[12] and Euk-Ploc[13] for predicting subcellular localization of proteins from different species, Nuc-Ploc[14] for predicting subnuclear localization, and a web-server package, Cell-Ploc,[15] for predicting subcellular localization of proteins in six different organisms. These web-servers provide great convenience for biological scientists working in this area.[16]

The extracted features used in these classifiers fall into two types: sequence-based and annotation-based. Sequence-based methods use k-mer composition, or represent sequences as condensed feature vectors using pseudo-amino acid composition,[3] signal-processing and text processing techniques.[17,18] N-terminal signals are very effective in identifying mitochondrial, chloroplast, and secretory pathway proteins. But sometimes the leading sequences of the test proteins are missing, and for many locations, no obvious sorting signal can be detected. Hence, we mainly use global sequence features. As annotation improves and becomes more abundant, many studies use annotation-based methods, including motifs, function domains, or gene ontology (GO)[19] to improve prediction accuracy.

Most of these studies focus on mono-locational proteins, i.e., they assume that proteins exist in only one cellular compartment. This is not always the case. Many proteins are multi-locational. They translocate into different compartments, or secret out of the cell. In most of the previous prediction systems, such proteins were discarded or treated like mono-locational proteins. This kind of proteins are quite common and play an important role in biological processes.[20] For example, 39% of organellar proteins in mouse liver have multiple locations.[21] Cai and Chou[22] dealt with such proteins in budding yeast by unfolding multi-label data. A tri-localized protein would be unfolded into three distinct samples with different labels and then predicted. In essence, their method treats the multi-locational proteins like single-locational ones. Certain strategies and evaluation measures are needed to deal with the multi-locational cases. Chou and Shen discussed the multiplex protein problem in a comprehensive review.[23] And several recently developed classifiers can deal with the multi-locational proteins, such as Hum-mPLoc,[24,25] Euk-mPLoc[26] and Cell-PLoc.[15]

In our previous studies,[27,28] we collected the multi-locational proteins from Swiss-Prot.[29] However, the annotations on subcellular localization are incomplete for many entries, so the performance of the prediction system would be underestimated. Recently, Zhang *et al.*[30] published a database of proteins with multiple subcellular locations, collected from multiple sources including primary protein databases and experimentally determined subcellular localization databases. Given high quality of the database, the performance of predictors for multi-locational proteins is estimated more accurately.

The most widely-used method for solving multi-label tasks is to split the original problem into a set of binary classification tasks using a one-versus-rest decomposition strategy. Using this strategy, the binary problems have the same scale of the original problem, and become imbalanced regarding the numbers of samples from the two classes. A less common method regards each different subset of the total label set as a new label. This method considers correlations among labels, but results in too many labels, the majority of which are associated with very few examples.[31] An alternative method is to conduct single-label classification with label ranking.[32] In this case, the size of the label set of the test samples needs to be specified.

In this paper, we use a min-max modular support vector machine (M^3-SVM) to predict protein subcellular multi-localization. The classifier is an ensemble of support vector machines (SVMs),[33] and works in a 'divide and conquer' manner. Ensemble classifiers are widely used to improve the generalization performance over single classifiers.[34,35] To solve a large-scale and complex multi-label problem, our method consists of three main steps: (a) decompose the original problem into two-class problems; (b) further decompose the two-class problems which are difficult to be learned into a number of relatively smaller and balanced two-class subproblems. (c) combine all the submodules into a hierarchical, parallel, and modular pattern classifier.

The purpose of developing an ensemble classifier is multifold. One, as the number of protein sequences and various annotation data grow rapidly, the scale of classification problems expands accordingly. Due to limitations on computation and memory capacity, traditional methods may be incompetent when the size of data set and the dimensionality of data increase significantly. The state-of-the-art classifier, SVMs, have demonstrated to be powerful tools for subcellular localization,[4,8,9] membrane protein types prediction,[36] image segmentation,[37] gender classification,[38] etc. But they still suffer from the complexity of their training algorithms, while the ensemble classifier can implement parallel learning and is suited for large-scale problems.

Two, the subcellular localization prediction is generally an imbalanced classification problem. The numbers of proteins located in different compartments vary significantly, i.e., the class distribution is uneven. For example, proteins in the cytoplasm, membrane and nucleus are much more numerous than those in other locations.

A number of approaches have been proposed to address the class imbalance problem. Oversampling and undersampling are two typical methods. The oversampling approach[39] duplicates data from the minority class; the undersampling approach eliminates data from the majority class. Both methods aim to re-balance the classes. Oversampling increases the complexity of the classification problem, while undersampling results in information loss. Although SVMs make the decision boundary based only on support vectors rather than all data samples, they still can not work well in class imbalance problems because of the imbalanced support vector ratio and weakness of soft-margins.[40,41] Adjusting the misclassification costs of positive and negative classes was suggested by Veropoulos *et al.*,[42] but it does not help SVM accuracy as much as would be expected.[41]

Our M^3-SVM classifier is suited for imbalanced classification problems. It decomposes the original problem into relatively balanced subproblems in order to eliminate the skew of the decision boundary. It has the following three advantages over traditional SVMs: (a) a large-scale and complex multi-class problem can be decomposed to two-class subproblems as small as necessary; (b) the two-class subproblems are independent to each other, therefore, they can be solved in parallel without the trouble of communication; (c) the two module combination principles are simple, and easily implemented in software and hardware. How to decompose the data set of a class for an M^3-classifier has not yet been perfectly solved. Random decomposition is the most straightforward way, dividing the majority and minority classes randomly into nearly equal sizes. But it cannot ensure stable performance. In this paper, we propose a new decomposition method based on biological prior knowledge, i.e., GO annotation. We noticed that for many pattern classification problems, the training data are organized by some prior knowledge, which could be useful clues to data decomposition. Here, we calculate the semantic similarity of GO terms, use the similarity to cluster proteins and partition training data for the proposed ensemble classifier.

Many studies have indicated that sequence-based prediction approaches, such as protein subcellular location prediction,[15,16,26] protein structural class prediction,[43,44] protein quaternary attribute prediction,[45,46] protein folding rate prediction,[47,48] identification of membrane proteins and their types,[49] identification of enzymes and their functional classes,[50] identification of GPCR and their types,[51,52] identification of proteases and their types,[53,54] protein cleavage site prediction,[55–57] signal peptide prediction,[58,59] and protein 3D structure prediction based on sequence alignment,[60] can timely provide very useful information and insights for both basic research and drug design and hence are widely welcome by science community; particularly, if a user-friendly web-server could be provided for each of these methods as stated in a recent review.[23] The present study is attempted to develop a new method for predicting subcellular localization of multiplex proteins using a M^3-SVM in hopes that it may become a useful complementary tool to the existing methods in the relevant areas.

The proposed method was evaluated by two data sets, namely DBMLoc[30] and yeast proteome.[20] In DBMLoc, all proteins have more than one location and are grouped into 13 classes according to their subcellular locations. The yeast dataset involves 22 subcellular compartments and about one third of the proteins are multi-locational. For both data sets, the M^3-SVMs showed better performance

than traditional SVMs using the same feature vectors. For DBMLoc, we compared GO decomposition and random decomposition, and found that GO decomposition helped to improve the prediction accuracy. We also compared the proposed method with three popular subcellular localization predictors, SubLoc,[8] LOCtree[61] and pTARGET.[62] Our method has a much higher total accuracy and location average accuracy on the multi-locational data sets. We believe that it provides a general solution to the imbalanced subcellular multi-localization of protein sequences.

2. Methods

In this paper, we propose a modular classifier, min-max modular support vector machine (M^3-SVM),[63,64] which is an ensemble of SVMs. Each SVM classifier is trained on a subset of the original data set. Given a test sample, each trained SVM outputs a classification result. Then all of the outputs are integrated for a final solution to the original problem according to two module combination rules, namely the minimization and the maximization principles.

2.1. *Classification of multi-label problems*

The traditional method for solving a multi-label task is to split the original problem into a set of binary classification tasks using the one-versus-rest decomposition strategy. For a K-class multi-label problem, let T denote its training set:

$$T = \{(x_m, t_m)\}_{m=1}^{L}, \quad t_m = \{t_m^k\}, \quad k = 1, \ldots, \tau_m, \tag{1}$$

where $x_m \in \mathrm{R}^n$ is the mth sample in the data set, t_m is the label set of x_m, t_m^k is the kth label of x_m, τ_m denotes the total number of labels of x_m, and L is the total number of samples.

By decomposing a K-class multi-label problem T into K mono-label two-class problems T_i for $i = 1, \ldots, K$, we have the training set of T_i as follows:

$$T_i = \{(x_m^{i+}, +1)\}_{m=1}^{L_i^+} \cup \{(x_m^{i-}, -1)\}_{m=1}^{L_i^-}, \tag{2}$$

where L_i^+ is the number of positive samples of the two-class problem T_i, and L_i^- is the number of negative samples. For T_i, positive samples are the samples whose label sets contain label i, and negative samples are the remaining ones. Thus x_m will appear τ_m times as positive training data, and $(K - \tau_m)$ times as negative training data.

Each binary classifier decides whether or not a novel sample belongs to a particular class. Obviously, each of the binary problems has the same size as the original problem. The data distribution would become more imbalanced because of the one-versus-rest strategy. For a complex and imbalanced binary classification problem, we further divide it into a number of relatively small and balanced two-class subproblems, each of which is solved by a SVM, and then all the subproblems are combined using MIN and MAX principles. The function of MIN is to find the minimum value of all its inputs:

$$y = \min_{1 \le i \le n} x_i, \tag{3}$$

where y is the output, and x_i is the input. Likewise, the function of MAX is to find the maximum value of all its inputs:

$$y = \max_{1 \le i \le n} x_i. \tag{4}$$

For a two-class problem T_i, its positive and negative training sets, T_i^+ and T_i^-, are further decomposed into N_i^+ and N_i^- subsets, where $1 \le N_i^+ \le L_i^+$ and $1 \le N_i^- \le L_i^-$.

$$T_i^{+j} = \{(x_m^+, +1)\}_{m=1}^{L_i^{+j}}, \quad j = 1, \ldots, N_i^+, \tag{5}$$

$$T_i^{-j} = \{(x_m^-, -1)\}_{m=1}^{L_i^{-j}}, \quad j = 1, \ldots, N_i^-, \tag{6}$$

where L_i^{+j} and L_i^{-j} are the numbers of samples in T_i^{+j} and T_i^{-j}, respectively.

Each two-class problem T_i is solved by an M^3 network shown in Fig. 1.

According to the MIN and MAX principles,[63] N_i^+ MIN units and one MAX unit are required to combine all the $(N_i^+ \times N_i^-)$ modules. Each of the MIN units combines N_i^- modules. The final output is determined by the outputs of all modules.

This ensemble classifier has some advantages over other methods in dealing with imbalanced problems. Compared with the under-sampling method, it makes full use of the training data without information loss. Compared with the over-sampling method, it does not add to learning cost in each module, and speeds up the training process. The MIN and MAX rules provide an effective ensemble principle to obtain a solution to the original problem from the sub-problems, and they are easy to implement.

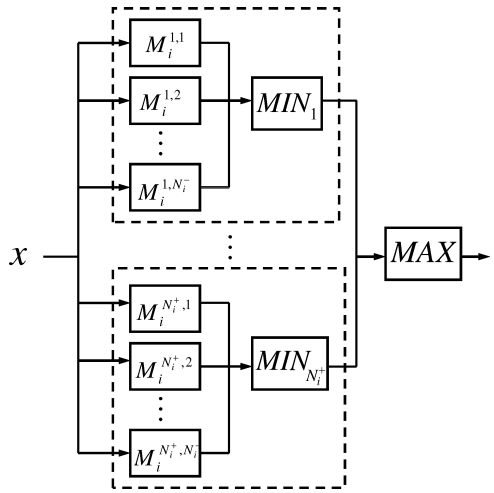


Fig. 1. Structure of M^3 network for a two-class problem T_i, which is divided into $(N_i^+ \times N_i^-)$ two-class subproblems.

2.2. *Task decomposition*

According to the one-versus-rest strategy, a K-class classification problem is decomposed into K two-class problems. Some of the two-class problems may have an extremely imbalanced distribution of the positive and negative classes. Moreover, some of the two-class problems may be too large for fast learning. The most important advantage of the M^3 model is that it can further divide the large and imbalanced two-class problems into relatively smaller and more balanced subproblems.

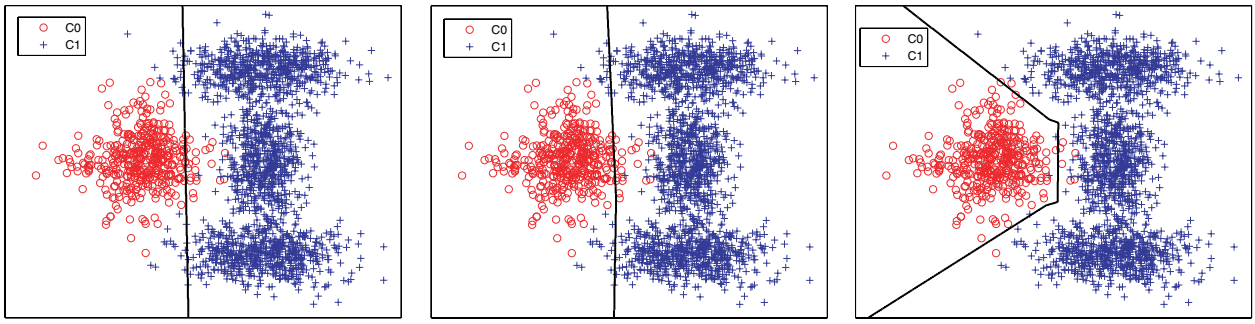
Task decomposition is a key issue for modular algorithms. Appropriate decomposition strategy can simplify the decision boundary, improve generalization performance and save learning time.

Random partition is the simplest and most straightforward way. Given a specific module size, when we choose samples randomly from the training set to build a module, the samples may have no distribution relationship with each other. In such cases, although the subproblem has a reduced data size, it still may be hard to solve, and have complex decision boundary apt to overfit. Since the overall classification capability lies on the performance of all the modules, the poor boundaries learned by some modules degrade the prediction accuracy for the whole system. Therefore, the random partition can not obtain stable performance.

Several decomposition strategies have been developed for the M^3 model, such as hyperplane decomposition[64] and equal clustering.[65] Hyperplane decomposition uses a group of parallel hyperplanes to partition data into subsets. This method is fast and suitable for sparse data. Equal clustering (EC) works similarly to K-means clustering. The only difference is that EC pays more attention to load balancing for the seek of parallel learning, thus the clusters are kept in nearly equal size. All these methods aim to utilize the geometric distribution characteristics of data points in the high-dimension space.

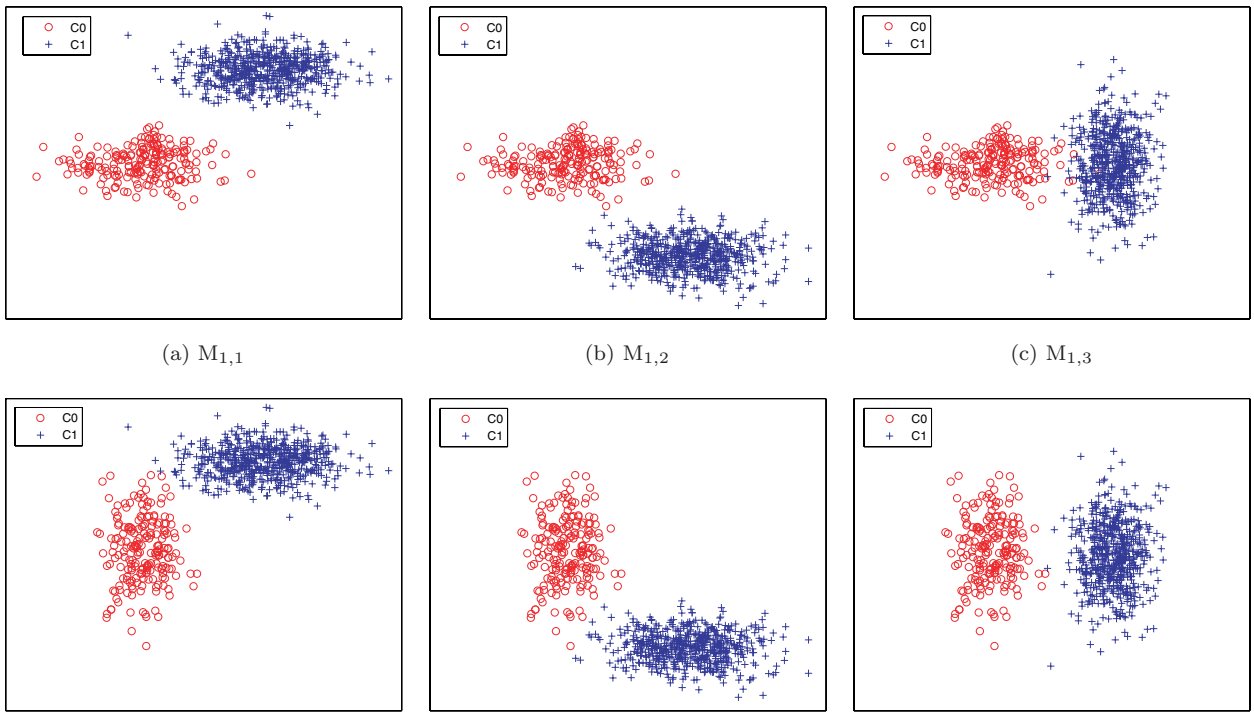
As a simple illustration on how module decomposition influences the decision boundary, we generated a two-dimensional data set including two classes, C_0 (red circle) and C_1 (blue plus sign), shown in Fig. 2. Class C_0 is a mixture of two 2D gaussian distributions, each of which has 200 samples. Class C_1 is a mixture of three 2D gaussian distributions, each of which has 600 samples. Figure 2 shows the decision boundaries learned by a traditional SVM and M^3-SVMs with two different module decomposition methods, where C_0 is divided into 2 modules, and C_1 is divided into 3 modules. They all use linear kernels with the same parameters. In Fig. 2(c), the data is decomposed by separating different Gaussian distributions. (Figure 3 shows the six subproblems, each of which consists of a module from the positive set and a module from the negative set.) In this example, we find that SVM and M^3-SVM(Random) have very similar decision boundaries. That is because in random decomposition, each subproblem is built by random sampling and has nearly the same distribution as the original data. After the combination of these subproblems, the decision boundary remains in similar shape. But there is a slight difference between (a) and (b): in the area that the two classes overlap, M^3-SVM(Random) pushes the boundary toward the majority class, and thus benefits the classification of the minority class. M^3-SVM with non-random decomposition has a much different decision boundary, which fits closer to the original distribution of the data set.

The combination process (MIN and MAX) of M^3-SVM (Fig. 4) shows how the final decision boundary is produced. Since we used a linear kernel, the decision boundary of each module is a straight line. Figure 4(a) shows a MIN unit, which is an intersection of the three boundaries for the positive class. Figure 4(b) shows another MIN unit, and



(a) SVM (b) M^3-SVM (Random) (c) M^3-SVM (non-Random)

Fig. 2. Decision boundaries of three classifiers on an example data set.



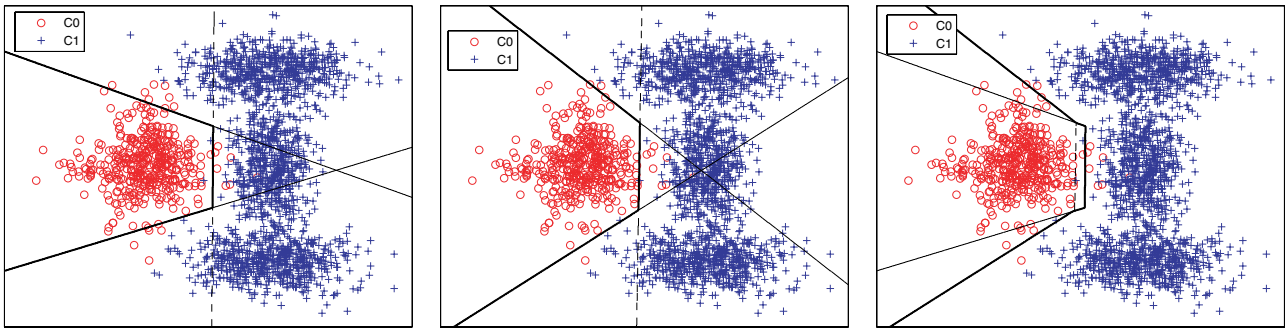
(a) $M_{1,1}$ (b) $M_{1,2}$ (c) $M_{1,3}$

(d) $M_{2,1}$ (e) $M_{2,2}$ (f) $M_{2,3}$

Fig. 3. Subproblems of M^3-SVM. $M_{i,j}$ denotes the subproblem consisting of the ith module of Class C_0 and the jth module of Class C_1.

Fig. 4(c) shows the final decision boundary which is a union of the two MIN units.

In the past, we either divided the data randomly or based the division on the distance between sample points in the feature space, like hyperplane decomposition and equal clustering,[28,65] but ignored prior knowledge, which can contribute useful information to clustering within a big class. Here, we want to fully utilize the GO information and achieve a better partition, such that the proteins sharing some common attributes could be grouped together. In the GO graph, GO terms are structured hierarchically and have semantic relations ('is-a' and 'part-of') with each other. A child node is more specialized than its parental nodes, and more than one parental node may exist. Here, we use a similarity measure of GO terms based on their semantic relations to cluster proteins in a class that needs to be decomposed. Given the similarity between two GO terms, the similarity between two sets of GO terms can be



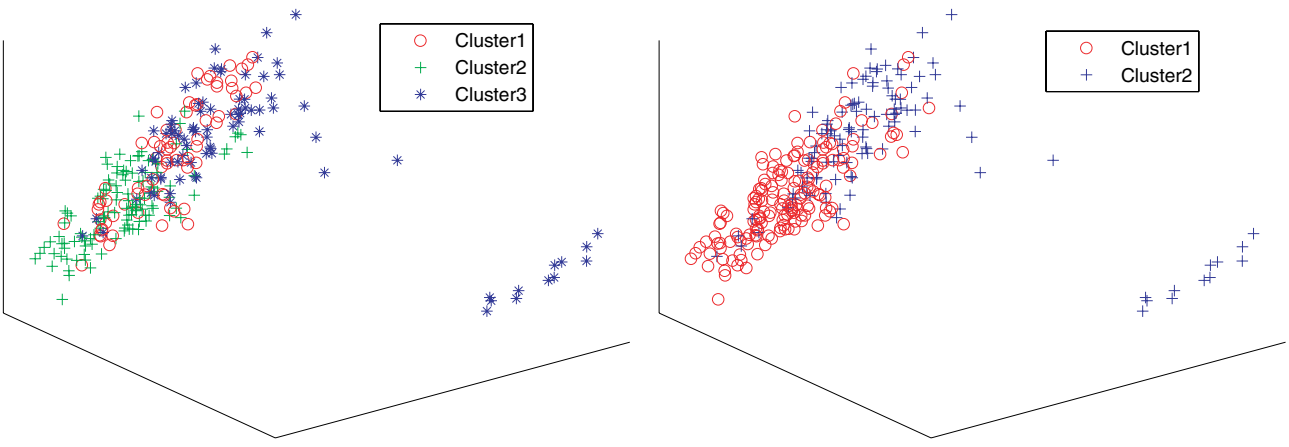
(a) MIN_1 (b) MIN_2 (c) MAX

Fig. 4. Combination of decision boundaries using MIN and MAX principles. The bold line denotes the final boundary.

calculated. Suppose each protein corresponds to a set of GO terms, the similarity between two proteins can be obtained accordingly.

The data set used in our experiments were annotated with GO terms, including cellular component, biological process and molecular function. In this work, we adopt the method proposed by Wang *et al.*[66] to measure the semantic similarity of GO terms. We build the GO similarity matrix for our training data set, and use the clustering tool, CLUTO[67] to partition the data based on the similarity matrix. To specify the number of clusters when running the program "scluster", we calculate the number of clusters according to the predefined module size: suppose that Class C_i has m samples and C_i is divided into k modules when n is the predefined module size, then m/k must be the closest value to n among all possible module sizes of C_i. Random decomposition makes each module of equal size, while in GO decomposition, the actual size of each module is determined by the clustering method.

Since it is hard to get an impression of how GO decomposition works at such a high-dimension feature space, we cluster 283 membrane proteins from the high quality non-redundant set of DBMLoc, and use the principal component analysis (PCA) to reduce the feature vectors into 3 dimensions. Figure 5 shows the data points of different clusters in different colors. In Fig. 5(a), the module size is 100 and three clusters are generated, while in Fig. 5(b), the module size is 200 and two clusters are generated. Although there is some overlap of data points between clusters which may be caused by the dimension reduction, we still find that the GO decomposition matches with data distribution well.



(a) Module size = 100 (b) Module size = 200

Fig. 5. Decomposition of membrane proteins based on GO information.

2.3. *Feature extraction*

The feature vectors have 100 dimensions. The former 60 dimensions are the amino acid composition of the full sequence on three secondary structure elements, i.e., strand (E), helix (H) and coil (C). The value of each dimension is calculated by

$$f_i^k = \frac{N_i^k}{L}, \tag{7}$$

where $k = \{H, E, C\}$, N_i^k is the frequency of amino acid i in secondary structure element k, and L is the length of the sequence. The latter 40 dimensions are the amino acid composition on two solvent accessibility statuses, namely buried (B) and exposed (E), and is calculated similarly as Eq. (7), with $k = \{B, E\}$.

The secondary structure elements were predicted by PSIPRED,[68] and solvent accessibility statuses were predicted by ACCpro.[69] Both of them are highly accurate prediction methods. All the feature vectors were scaled in the range of $[0, 1]$ using SVM-Scale in the LibSVM package.[70]

Amino acid frequency on different secondary structure elements and solvent accessibility statuses were first used in protein fold classification by Shamim *et al.*[71] This method was also demonstrated to be effective in our experiments.

3. Results and Discussion

In order to test the performance of our methods, we applied them to two data sets. The first one was extracted from a high-quality multi-locational protein database, DBMLoc, published by Zhang *et al.*[30] which was collected from multiple databases, such as Swiss-Prot/TrEMBL,[29] experimentally determined subcellular localization databases, and literature references. The second one is a well-studied yeast proteome data set published by Huh *et al.*[20] The subcellular locations were determined by green fluorescent protein (GFP) experiments, including over 4,000 budding yeast proteins located in 22 subcellular locations.

For each data set, we compared the accuracy of our method with traditional SVMs. Since the DBMLoc data is fully annotated with gene ontology information, we assess the potential of using GO semantic information as a similarity measure to decompose training data for the ensemble classifier. We also compared three other protein subecellular predictors on the same test data set. For the yeast proteome data, since Chou and Cai[22,32] have done extensive research for predicting the multiple subcellular locations on it, we took their method as a baseline to examine the performance of M^3-SVMs.

3.1. *Experiment settings and evaluation criteria*

We chose LibSVM version 2.8[70] as the base classifier for the ensemble classifier. And the traditional SVM also uses LibSVM version 2.8 as its implementation. We experimented with polynomial, sigmoid and RBF kernels and observed that RBF kernel has the best classification accuracy. All experiments were conducted on a Pentium 4 double CPU (2.8GHz) PC with 2GB RAM.

Multiple measures were used to assess the performance of our proposed method, including precision (P), recall (R), F_1, total accuracy (TA), location accuracy (LA) and average F_1 ($aveF_1$). The former three measures, P, R and F_1, are used to measure the prediction quality of each location, and the last three measures, TA, LA and $aveF_1$, are used to measure the overall prediction quality across all locations. These six measures are defined below.

For each location l, the precision, recall and F_1 can be defined by true positive (TP), false positive (FP), false negative(FN) as follows

$$P_l = \frac{TP_l}{TP_l + FP_l} \tag{8}$$

$$R_l = \frac{TP_l}{TP_l + FN_l} \tag{9}$$

$$F_{1,l} = \frac{2 \times P_l \times R_l}{P_l + R_l}, \tag{10}$$

where R_l is the ratio of samples belonging to class C_l and classified correctly compared to the number of samples of C_l. P_l is the ratio of samples belonging to C_l classified correctly compared to the number of samples classified into C_l. The $F_{1,\,l}$ measure corresponds to the harmonic mean of R_l and P_l.

In Ref. 9, Park *et al.* defined total accuracy and location accuracy. We redefine these two measures as following equations because of the multi-label characteristics,

$$TA = \frac{\sum_{l=1}^{K} TP_l}{N} \tag{11}$$

$$LA = \frac{\sum_{l=1}^{K} R_l}{K}, \tag{12}$$

where N is the total number of labels instead of proteins in the data set, K is the number of subcellular locations, and R_l is the recall of location l. The average F_1 is defined by

$$aveF_1 = \frac{\sum_{l=1}^{K} F_{1,l}}{K}. \quad (13)$$

Unlike TA and LA, which only focus on the ratio of true positives, $aveF_1$ also reflects false positive rate.

3.2. *Experimental results on DBMLoc database*

DBMLoc database collects multi-locational proteins from animal, plant, virus and bacteria. All the cellular compartments were assigned into twelve categories: extracellular, cell wall, membrane, cytoplasm, mitochondrion, nucleus, ribosome, plastid, endoplasmic reticulum (ER), Golgi apparatus, vacuole and virion. Some subcellular location annotations that can not be classified into the twelve categories are assigned to 'others'. As a result, the number of classes used in our prediction system is 13.

To remove the homologous sequences from the benchmark dataset, a cutoff threshold of 25% was imposed in Refs. 10–14 to exclude those proteins from the benchmark datasets that have equal to or greater than 25% sequence identity to any other in a same subset. In this study, we also used a non-redundant data with sequence similarity below 25%. The training and test data sets are mutually exclusive. Test data is a high quality set including 631 proteins. Training data has a total of 2344 proteins, extracted from the complete non-redundant set (25%) by removing the overlapping data with test set. The statistics of the data sets are shown in Table 1. From this table, we see that the data distributions are very imbalanced on different cellular compartments. Proteins of membrane, cytoplasm, and nucleus make an overwhelming portion, which adds to classification difficulty.

All of the proteins in DBMLoc database have no less than two locations. The training proteins have at most five locations, while test proteins have at most four locations. Table 2 lists the detailed number of proteins in the training and test data for each compartment, and for bi-, tri-, quad- and penta-locational cases, in that order. For example, of the 471 extracellular proteins, 395 are bi-localized, 63 are tri-localized, 11 are quad-localized and 2 are penta-localized. The protein number at the end of each column is the sum of the column divided by the number of labels of the column. For example, the number of quad-localized proteins in the training set is $92/4 = 23$. The average number of labels for each protein is 2.2.

Although this data set is fully annotated, we did not use gene ontology as features to build our predictor based on the consideration that many test proteins are novel and do not have GO information. But

Table 1. Training and test data distributions of DBMLoc.

Lable	Location	Training	Test
1	Others	134	36
2	Extracellular	471	43
3	Ribosome	58	15
4	Virion	31	2
5	Membrane	1240	283
6	Cytoplasm	1172	417
7	Mitochondrion	445	123
8	Nucleus	844	344
9	Plastid	132	8
10	Vacuole	16	4
11	Cell wall	21	5
12	ER	322	53
13	Golgi	162	45
Total label no.		5048	1378
Total protein no.		2344	631

Table 2. Label distribution of DBMLoc.

Label	Training label no.				Test label no.		
	2	3	4	5	2	3	4
1	0	126	6	2	0	32	4
2	395	63	11	2	32	10	1
3	25	33	0	0	8	4	3
4	29	2	0	0	2	0	0
5	1041	179	17	3	217	58	8
6	988	160	21	3	341	64	12
7	375	66	3	1	99	18	6
8	728	107	8	1	290	45	9
9	88	44	0	0	8	0	0
10	11	4	1	0	3	1	0
11	18	3	0	0	2	2	1
12	234	75	12	1	32	16	5
13	94	53	13	2	22	20	3
Label no.	4026	915	92	15	1065	270	52
Protein no.	2013	305	23	3	528	90	13

the prior knowledge about training data could be fully utilized. Therefore, we used GO semantic similarity to guide the module decomposition of training data. We experimented with three methods using the same feature vectors as mentioned in Sec. 2.3. One is a min-max modular support vector machine with gene ontology decomposition (M^3-SVM(GO)). The second is a min-max modular support vector machine with random decomposition (M^3-SVM(R)). The last one is a traditional SVM. The module sizes of 100, 400 and 800 were tested for both M^3-SVM(GO) and M^3-SVM(R). And the results of M^3-SVM(R) were averaged after five repeated experiments.

Table 3 shows the overall performance (TA, LA and $aveF_1$) of the three methods, where traditional SVMs, M^3-SVM(R) and M^3-SVM(GO) and three different module sizes for M^3-SVMs are compared. Column 2 shows the predefined module sizes. Column 3 shows the numbers of subproblems. From this table, several observations can be made. First, both M^3-SVMs with random decomposition and with GO decomposition have higher TA and LA than traditional SVMs. The M^3-SVMs improve not only average location accuracy but also total accuracy, which indicates that they do not sacrifice majority classes for the classification of minority classes. Second, M^3-SVMs have higher $aveF_1$ than traditional SVMs except M^3-SVM(R) with module size 100. And M^3-SVM(GO) with module size 800 achieved the highest $aveF_1$. As the module size decreases, LA increases, but $aveF_1$ decreases, which suggests a higher false positive rate of M^3-SVMs when the module size becomes small. So there is a trade-off between location accuracy and false positive rate. Finally, M^3-SVM(R) has lower TA and $aveF_1$ than M^3-SVM(GO) but higher LA, suggesting that M^3-SVM(R) gives more preference to the minority classes. And M^3-SVM(R) performs worse than M^3-SVM(GO) when the module size is very small (100). Since random decomposition randomly divides each training class into equal size modules, while GO decomposition is based on the relationship between proteins, the GO decomposition has a more stable performance.

Table 3. Overall accuracy of three methods on DBMLoc.

Method	Module size	Module no.	TA (%)	LA (%)	$aveF_1$ (%)
SVM	2344	13	64.7	41.4	42.0
M^3-SVM (Random)	100	848	65.1	48.0	40.7
	400	83	66.9	47.7	42.7
	800	38	66.2	45.2	43.4
M^3-SVM (GO)	100	848	66.2	**48.5**	42.3
	400	83	**67.1**	45.2	42.6
	800	38	66.3	43.4	**43.6**

Tables 4 and 5 list detailed *Recall* and F_1 of traditional SVMs and M^3-SVMs(GO) on each location. Obviously, the modulization helps improve *Recall* a lot especially for the minority classes, while the two biggest classes, membrane and Cytoplasm are recognized best by M^3-SVM(GO) with module size 800 and the traditional SVM, respectively.

As the F_1 values listed in Table 5 show, M^3-SVMs(GO) with module size 800 has the best performance. It outperforms on 5 locations, extracellular, ribosome, virion, membrane, and mitochondrion, among the four methods, and has higher F_1 than traditional SVMs on 8 locations.

In this experiment, M^3-SVMs(GO) with module size 800 performs the best considering all the measures. A too small module size would result in numerous modules which increase computation cost and deteriorate the performance. However, we could specify different module sizes for different binary

Table 4. Recall comparison on DBMLoc. a, b, c and d denote four methods. a: SVM, b: M^3-SVM (GO) with module size 100, c: M^3-SVM (GO) with module size 400, d: M^3-SVM (GO) with module size 800. 1–13 correspond to the 13 subcellular locations listed in Table 1.

Label	Recall (%)			
	a	b	c	d
1	0.0	**5.6**	2.8	2.8
2	48.8	**62.8**	53.5	55.8
3	20.0	20.0	20.0	**26.7**
4	100.0	100.0	100.0	100.0
5	63.3	61.8	63.3	**66.4**
6	**84.7**	82.5	83.5	83.9
7	52.8	57.7	**59.3**	57.7
8	69.2	73.3	**75.0**	71.2
9	25.0	**50.0**	25.0	12.5
10	0.0	0.0	0.0	0.0
11	20.0	**60.0**	40.0	40.0
12	41.5	43.4	**45.3**	32.1
13	13.3	13.3	**20.0**	15.6

Table 5. F_1 comparison on DBMLoc. a, b, c and d are the same as in Table 4.

Label	F_1 (%)			
	a	b	c	d
1	0.0	**7.0**	4.5	4.8
2	44.7	42.9	44.2	**46.2**
3	31.6	24.0	27.3	**38.1**
4	**100.0**	80.0	80.0	**100.0**
5	67.5	66.7	67.4	**69.8**
6	81.1	**81.2**	80.7	80.6
7	52.2	50.4	51.0	**53.0**
8	72.7	76.0	**76.6**	74.0
9	16.7	**17.8**	12.5	8.0
10	0.0	0.0	0.0	0.0
11	22.2	**50.0**	44.4	40.0
12	38.9	34.3	**40.0**	30.9
13	18.5	20.0	**25.4**	20.9

classification problems according to the ratio of training samples of the positive and negative classes.

In addition, we notice that all the classifiers failed to recognize vacuole proteins from other proteins. One reason is that it has the least training samples (16 proteins) and only 4 test samples, hence the ratio of positive and negative data is too small to classify the positive data correctly. The other reason would be that the current feature vectors do not contain features that are informative enough to discriminate vacuole proteins from others.

3.3. *Comparison with other methods on DBMLoc test set*

To our knowledge, there is no predictor specialized for multi-locational proteins. It is still interesting to see how existing predictors classify the multi-locational proteins. We compared our method with SubLoc,[8] LOCtree[61] and pTARGET[62] on the DBMLoc test set. SubLoc pioneered the SVM-based prediction of protein localization using animo acid composition. LOCtree is a hierarchical classifier also based on SVMs. pTARGET calculates scores on Pfam domains and amino acid composition.

Because these methods can not predict all of the subcellular locations of our data set, we calculate TA and LA only on the overlapping classes. For SubLoc and Loctree, four locations (extracellular, cytoplasm, mitochondria and nucleus) were compared, and for pTARGET, seven locations were compared (extracellular, membrane, cytoplasm, mitochondria, nucleus, ER and Golgi).

The total accuracy and location accuracy are listed in Table 6. The classifiers that can only deal with mono-locational proteins have a much lower accuracy. We also found that the results of these methods have a small portion of overlap, i.e., they make different decisions on most of the multi-label data. SubLoc and LOCtree have relatively high TA and LA because they only predict 4 locations. And, LOCtree's hierarchical architecture helps improve location accuracy.

3.4. *Experimental results on the yeast proteome*

Chou and Cai[22,32] first formalized the problem of subcellular multi-localization based on the yeast proteome data set.[20] They excluded redundant proteins with high sequence similarity and also those whose locations were ambiguous, and extracted a condensed set.[22] We tested our method on the same non-redundant set (sequence identity below 40%), which has a total of 3555 proteins and 4709 labels. Tables 7 and 8 show the data and label distributions of this data set. We can see that about 70% proteins are mono-locational, and only one has five labels.

We carried out a 10-fold cross-validation using both M^3-SVMs and traditional SVMs. Among the independent dataset tests, sub-sampling (e.g., 5 or 10-fold cross-validation) test and jackknife test, which are often used for examining the accuracy of a statistical prediction method,[72] the jackknife test was deemed the most objective that can always yield a unique result for a given benchmark dataset, as elucidated in Ref. 15 and demonstrated by Eq. (50) of Ref. 16. Therefore, the jackknife test has been

Table 6. Comparison with other predictors. The labels correspond to the locations as shown in Table 1, i.e., 2: Extracellular, 6: Cytoplasm, 7: Mitochondria, 8: Nucleus, 5: Membrane, 12: ER, 13: Golgi.

Label	Method	TA (%)	LA (%)
2, 6, 7, 8	SubLoc	41.0	38.6
	LOCtree	44.2	51.2
	M^3-SVM(GO)	**74.4**	**67.2**
2, 5, 6, 7, 8, 12, 13	pTARGET	34.6	34.7
	M^3-SVM(GO)	**69.0**	**54.7**

Table 7. Data distribution of the yeast data.

Label	Location	Protein No.
1	Actin	29
2	Bud	23
3	Bud neck	60
4	Cell periphery	106
5	Cytoplasm	1576
6	Early Golgi	51
7	Endosome	43
8	ER	272
9	ER to Golgi	6
10	Golgi	40
11	Late Golgi	37
12	Lipid particle	19
13	Microtubule	20
14	Mitochondrion	494
15	Nuclear periphery	59
16	Nucleolus	157
17	Nucleus	1333
18	Peroxisome	20
19	Punctuate composite	123
20	Spindle pole	58
21	Vacuolar membrane	54
22	Vacuole	129
	Total label no.	4709
	Total protein no.	3555

Table 8. Label distribution of the yeast data.

Location no.	1	2	3	4	5
Protein no.	2476	1013	58	7	1

increasingly used and widely recognized by investigators to test the power of various prediction methods. However, to avoid intensive computational cost, here we adopt the 10-fold cross-validation which is also widely used in the studies using SVMs as the prediction engine. For M^3-SVMs, the big classes were decomposed randomly because the GO annotations are incomplete in this data set. We experimented with 4 different module sizes, 50, 100, 500 and 1000.

Amino acid composition on different secondary structure and solvent accessibility status were used to be the basic features. Chou and Cai developed the GO-FunD-PseAA method, hybridizing three kinds of feature space, gene ontology, function domain and pseudo-amino acid composition. This method first classifies proteins based only on GO terms. The proteins without GO information are classified by function domains, and those without function domains are dealt with in the third feature space. In order to get competitive feature vectors for the comparison on classification methods, we also added GO terms into our feature vectors in a binary form. Each value denotes whether a certain GO term is present or absent in the annotation of a test protein. The combined feature vectors have $(100 + m)$ dimensions, where m is the total number of GO terms appeared in the data set. Compared with their feature extraction, our method is easier to implement and could be more suited for those proteins that are not well-annotated. Some proteins in this data set have no GO annotation, so their last m dimensions are all zero.

To measure performance, we list TA, LA and $aveF_1$ (Table 9) of three different classifiers using the same feature vectors mentioned above. Chou and Cai only reported the total accuracy (68%). They used a nearest neighbor classifier with a generalized distance definition. We implemented their classifier using our feature vectors, and an approximative total accuracy of 67% was obtained. Therefore, the LA and $aveF_1$ are very close estimations. Later in Ref. 32, they reported a higher total accuracy of 70%, but it was on a redundant set that contains 3875 proteins. We only compare accuracy on the same non-redundant set.

The same trend as that in the former experiment can be observed here. M^3-SVMs have both higher TA and LA than traditional SVMs. And M^3-SVM with module size 500 has the highest $aveF_1$, which demonstrates the effectiveness of modularization. As the module size decreases, although the TA and LA rises, the average F_1 drops. That is because the number of false positives increases when the module size is small. Detailed recall and F_1 for each location can be

Table 9. Accuracy comparison on yeast data. NN denotes the nearest neighbor classifier with a generalized distance defined in Refs. 22 and 32.

Method	TA (%)	LA (%)	$aveF_1$ (%)
M^3-SVM (50)	**74.3**	**59.4**	54.0
M^3-SVM (100)	74.4	58.7	55.5
M^3-SVM (500)	73.4	53.6	**56.7**
M^3-SVM (1000)	71.3	52.1	56.4
SVM	69.3	49.1	55.6
NN	67.0	48.3	49.5

Table 10. Accuracy comparison of three methods on yeast data. a, b and c denote three methods. a: M^3-SVM(100), b: M^3-SVM(500), c: Traditional SVM.

Label	Recall (%)			F_1(%)		
	a	b	c	a	b	c
1	**58.6**	**58.6**	48.3	60.7	**64.2**	57.1
2	**30.4**	13.0	4.4	**25.9**	17.7	7.1
3	**81.4**	69.5	62.7	**70.1**	66.1	66.1
4	**55.8**	47.1	32.7	45.9	**50.8**	43.6
5	75.7	**77.3**	75.7	74.7	76.0	**76.4**
6	**51.0**	37.3	27.5	40.9	**41.8**	35.0
7	**67.4**	**67.4**	65.1	65.9	67.4	**72.7**
8	**73.8**	73.4	62.7	66.1	69.3	**69.6**
9	66.7	66.7	66.7	80.0	80.0	80.0
10	**60.0**	**60.0**	55.0	51.1	**56.5**	54.3
11	13.9	**25.0**	19.4	14.3	**36.7**	30.4
12	47.4	47.4	**47.7**	45.0	48.7	**52.9**
13	**40.0**	30.0	20.0	**47.1**	38.7	27.6
14	85.3	**85.5**	81.5	76.5	79.1	**81.6**
15	**55.9**	52.5	49.2	**61.1**	59.1	58.6
16	**77.6**	74.4	74.4	71.4	76.1	**78.1**
17	81.7	**83.4**	78.8	80.0	81.0	**81.9**
18	50.0	55.0	**60.0**	58.8	64.7	**68.6**
19	**22.8**	8.9	8.1	**17.5**	12.2	13.1
20	**67.2**	63.8	62.1	65.6	**69.2**	**69.2**
21	**50.0**	48.2	46.3	49.1	**54.7**	53.8
22	**58.9**	41.1	33.3	44.6	**46.9**	45.5

found in Table 10, which compares the three methods, M^3-SVM with module size 100, M^3-SVM with module size 500 and traditional SVMs. Undoubtedly, M^3-SVM(100) has the highest recall values on most of the locations, while regarding F_1, M^3-SVM(500) performs the best. Compared with traditional SVMs, M^3-SVM(500) has higher F_1 at 10 locations, 6 of which obtain more than 5% increase (actin, bud, cell periphery, early Golgi, late Golgi and microtubule), while SVMs have over 5% higher F_1 at only one location, endosome.

3.5. *Response time comparison*

Response time should also be considered as an important factor when measuring the performance of a classifier. Table 11 exhibits a comparison of response time between traditional SVMs and M^3-SVMs of different module sizes. We report two categories of run times. 'Time1' is the response time (including training and test time) of the classifier running all subproblems in series. 'Time2' is the training time for a single subproblem which costs the longest time. In parallel learning, 'Time2' is more important. For traditional SVMs, a subproblem means a two-class problem.

Table 11. A comparison of response times.

Dataset	Method	Time1 (sec.)	Time2 (sec.)
DBMLoc	SVM	23.4	3.3
	M^3-SVM (100)	22.2	<0.1
	M^3-SVM (400)	**19.2**	0.4
	M^3-SVM (800)	19.7	1.3
Yeast	SVM	33.3	5.7
	M^3-SVM(50)	36.3	<0.1
	M^3-SVM(100)	26.3	<0.1
	M^3-SVM(500)	**23.5**	0.6
	M^3-SVM(1000)	25.2	2.3

In both experiments, M^3-SVMs obtained a shorter response time than traditional SVMs even in sequential running. Regarding 'Time1', M^3-SVMs with the module sizes 400 and 500 are the most efficient for DBMLoc and yeast data, respectively, because they achieve a balance between the number and size of modules. For large-scale data, we can train modules in parallel, and choose a module size as small as possible.

4. Conclusion

This paper introduces an ensemble classifier for protein subcellular multi-localization. The classifier works in a modular manner, and has several advantages in solving large-scale, class imbalance, multi-label problems. Parallel and distributed training can be easily implemented because of its modularity, and it has a balanced performance on all classes because various task decomposition strategies can be used.

Taking the GO information into account, we partition large classes into smaller modules according to GO semantic similarity matrices. The ensemble predictor discriminates a wide range of subcellular compartments, and also predicts multiple locations for a protein, thus supplying useful hints for life scientists to determine locations within a possible compartment set.

The experiments were conducted on two different data sets. The experimental results demonstrate that M^3-SVM is very competent in solving complex

problems with class imbalance and multi-label characteristics. And, it is recognized that incorporating biological knowledge into task decomposition is a reliable way to improve the generalization ability and efficiency of M^3-SVM. As more and more large-scale applications appear in the realm of computational biology, parallel and distributed methods are of great need. We believe that M^3-SVM, which uses simple combination principles and can be implemented easily in practice, will be very useful for solving complex classification problems with large data sets.

Acknowledgments

This work was partially supported by the National Natural Science Foundation of China (Grants No. 60773090 and No. 90820018), the National Basic Research Program of China (Grant No. 2009CB320901), and the National High-Tech Research Program of China (Grant No. 2008AA02Z315).

References

1. J. Cedano, P. Aloy, J. A. Perez-Pons and E. Querol, Relation between amino acid composition and cellular location of proteins, *Journal of Molecular Biology* **266**(3) (1997) 594–600.
2. K. C. Chou and D. W. Elrod, Protein subcellular location prediction, *Protein Engineering* **12** (1999) 107–118.
3. K. C. Chou, Prediction of protein cellular attributes using pseudo-amino acid composition, *Proteins: Structure, Function, and Genetics* **43** (2001) 246–255.
4. K. C. Chou and Y. D. Cai, Using functional domain composition and support vector machines for prediction of protein subcellular location, *J Biol Chem* **277** (2002) 45765–45769.
5. G. P. Zhou and K. Doctor, Subcellular location prediction of apoptosis proteins, *PROTEINS: Structure, Function, and Genetics* **50** (2002) 44–48.
6. A. Reinhardt and T. Hubbard, Using neural networks for prediction of the subcellular location of proteins, *Nucleic Acids Research* **26**(9) (1998) 2230–2236.
7. Y. Fujiwara, M. Asogawa and K. Nakai, Prediction of mitochondrial targeting signals using hidden markov model, *Genome Informatics* **8** (1997) 53–60.
8. S. Hua and Z. Sun, Support vector machine approach for protein subcellular localization prediction, *Bioinformatics* **17**(8) (2001) 721–728.
9. K. J. Park and M. Kanehisa, Prediction of protein subcellular locations by support vector machines using compositions of amino acids and amino acid pairs, *Bioinformatics* **19**(13) (2003) 1656–1663.
10. K. C. Chou and H. B. Shen, Hum-PLoc: A novel ensemble classifier for predicting human protein subcellular localization, *Biochemical and Biophysical Research Communications* **347** (2006) 150–157.
11. H. B. Shen and K. C. Chou, Gpos-PLoc: An ensemble classifier for predicting subcellular localization of Gram-positive bacterial proteins, *Protein Engineering, Design and Selection* **20** (2007) 39–46.
12. H. B. Shen and K. C. Chou, Virus-PLoc: A fusion classifier for predicting the subcellular localization of viral proteins within host and virus-infected cells, *Biopolymers* **85** (2007) 233–240.
13. H. B. Shen, J. Yang and K. C. Chou, Euk-PLoc: An ensemble classifier for large-scale eukaryotic protein subcellular location prediction, *Amino Acids* **33** (2007) 57–67.
14. H. B. Shen and K. C. Chou, Nuc-PLoc: A new web-server for predicting protein subnuclear localization by fusing PseAA composition and PsePSSM, *Protein Engineering, Design and Selection* **20** (2007) 561–567.
15. K. C. Chou and H. B. Shen, Cell-PLoc: A package of web-servers for predicting subcellular localization of proteins in various organisms, *Nature Protocols* **3** (2008) 153–162.
16. K. C. Chou and H. B. Shen, Review: Recent progresses in protein subcellular location prediction, *Analytical Biochemistry* **370** (2007) 1–16.
17. Y. Yang and B. L. Lu, Extracting features from protein sequences using chinese segmentation techniques for subcellular localization, *Proc. the 2005 IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology* (2005) 288–295.
18. W. Y. Yang, B. L. Lu and Y. Yang, A comparative study on feature extraction from protein sequences for subcellular localization prediction, *Proc. the 2006 IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology* (2006) 201–208.
19. The Gene Ontology Consortium, Gene ontology: tool for the unification of biology, *Nature Genet.* **25** (2000) 25–29.
20. W. K. Huh, J. V. Falvo, L. C. Gerke, A. S. Carroll, R. W. Howson, J. S. Weissman and E. K. O'Shea, Global analysis of protein localization in budding yeast, *Nature* **425**(6959) (2003) 686–691.
21. L. J. Foster, C. L. de Hoog, Y. Zhang, Y. Zhang, X. Xie, V. K. Mootha and M. Mann, A mammalian organelle map by protein correlation profiling, *Cell* **125**(1) (2006) 187–199.
22. Y. D. Cai and K. C. Chou, Predicting 22 protein localizations in budding yeast, *Biochem Biophys Res Commun* **323**(2) (2004) 425–428.
23. K. C. Chou and H. B. Shen, Review: Recent advances in developing web-servers for predicting protein attributes, *Natural Science* **2** (2009) 63–92.

24. H. B. Shen and K. C. Chou, A top-down approach to enhance the power of predicting human protein subcellular localization: Hum-mPLoc 2.0, *Analytical Biochemistry* **394** (2009) 269–274.
25. H. B. Shen and K. C. Chou, Hum-mPLoc: An ensemble classifier for large-scale human protein subcellular location prediction by incorporating samples with multiple sites, *Biochem Biophys Res Commun* **355** (2007) 1006–1011.
26. K. C. Chou and H. B. Shen, Euk-mPLoc: A fusion classifier for large-scale eukaryotic protein subcellular location prediction by incorporating multiple sites, *Journal of Proteome Research* **6** (2007) 1728–1734.
27. Y. Yang and B. L. Lu, Prediction of protein subcellular multi-locations with a min-max modular support vector machine, *Lecture Notes in Computer Science* **3973** (2006) 667–673.
28. K. Chen, B. L. Lu and J. T. Kwok, Efficient classification of multi-label and imbalanced data using min-max modular classifiers, *Intl. Joint Conf. on Neural Networks* (2006) 1770–1775.
29. B. Boeckmann, A. Bairoch, R. Apweiler, M. C. Blatter, A. Estreicher, E. Gasteiger, M. J. Martin, K. Michoud, C. O'Donovan, I. Phan *et al.*, The SWISS-PROT protein knowledgebase and its supplement TrEMBL in 2003, *Nucleic Acids Research* **31**(1) (2003) 365–370.
30. S. Zhang, X. Xia, J. Shen, Y. Zhou and Z. Sun, DBMLoc: A Database of proteins with multiple subcellular localizations, *BMC Bioinformatics* **9**(1) (2008) 127–131.
31. G. Tsoumakas and I. Vlahavas, Random k-labelsets: An ensemble method for multilabel classification, *Lecture Notes in Computer Science* **4701** (2007) 406–417.
32. K. C. Chou and Y. D. Cai, Predicting protein localization in budding Yeast, *Bioinformatics* **21**(7) (2005) 944–950.
33. V. Vapnik, *Statistical Learning Theory* (Wiley-Interscience, NY, USA, 1998).
34. M. A. H. Akhand, Md. Monirul Islam and K. Murase, A comparative study on data sampling techniques for constructing neural network ensembles, *Intl. Journal of Neural Systems* **19**(2) (2009) 67–89.
35. C. Silva and B. Ribeiro, Towards Expanding Relevance Vector Machine to Large Scale Datasets, *International Journal of Neural Systems* **18**(1) (2008) 45–58.
36. Y. D. Cai, G. P. Zhou and K. C. Chou, Support vector machines for predicting membrane protein types by using functional domain composition, *Biophysical Journal* **84** (2003) 3257–3263.
37. B. Cyganek, Colour Image Segmentation with Support Vector Machines: Applications to Road Signs Detection, *International Journal of Neural Systems* **18**(4) (2008) 339–345.
38. H. C. Lian and B. L. Lu, Multi-view gender classification using multi-resolution local binary patterns and support vector machines, *International Journal of Neural Systems* **17**(6) (2007) 479–487.
39. N. V. Chawla, K. W. Bowyer, L. O. Hall and W. P. Kegelmeyer, SMOTE, Synthetic minority oversampling technique, *Journal of Artificial Intelligence Research* **16** (2002) 321–357.
40. G. Wu and E. Y. Chang. Class-boundary alignment for imbalanced dataset learning, *Proc. the ICML'03 Workshop on Learning from Imbalanced Data Sets* (2003).
41. R. Akbani, S. Kwek and N. Japkowicz, Applying support vector machines to imbalanced datasets, *Lecture Notes in Computer Science* **3201** (2004) 39–50.
42. K. Veropoulos, C. Campbell and N. Cristianini. Controlling the sensitivity of support vector machines, *Proc. the Intl. Joint Conf. on Artificial Intelligence* (1999) 55–60.
43. K. C. Chou, A novel approach to predicting protein structural classes in a (20-1)-D amino acid composition space, *Proteins: Structure, Function and Genetics* **21** (1995) 319–344.
44. K. C. Chou and Y. D. Cai, Predicting protein structural class by functional domain composition, *Biochemical and Biophysical Research Communications* **321** (2004) 1007–1009.
45. H. B. Shen and K. C. Chou, QuatIdent: A web server for identifying protein quaternary structural attribute by fusing functional domain and sequential evolution information, *Journal of Proteome Research* **8** (2009) 1577–1584.
46. X. Xiao, P. Wang and K. C. Chou, Predicting protein quaternary structural attribute by hybridizing functional domain composition and pseudo amino acid composition, *Journal of Applied Crystallography* **42** (2009) 169–173.
47. H. B. Shen, J. N. Song and K. C. Chou, Prediction of protein folding rates from primary sequence by fusing multiple sequential features, *Journal of Biomedical Science and Engineering* (*JBiSE*) **2** (2009) 136–143.
48. K. C. Chou and H. B. Shen. FoldRate: A web-server for predicting protein folding rates from primary sequence, *The Open Bioinformatics Journal* **3** (2009) 31–50.
49. K. C. Chou and H. B. Shen, MemType-2L: A Web server for predicting membrane proteins and their types by incorporating evolution information through Pse-PSSM, *Biochem Biophys Res Comm.* **360** (2007) 339–345.
50. H. B. Shen and K. C. Chou, EzyPred: A top-down approach for predicting enzyme functional classes

and subclasses, *Biochem Biophys Res Comm.* **364** (2007) 53–59.
51. K. C. Chou, Prediction of G-protein-coupled receptor classes, *Journal of Proteome Research* **4** (2005) 1413–1418.
52. X. Xiao, P. Wang and K. C. Chou, GPCR-CA: A cellular automaton image approach for predicting G-protein-coupled receptor functional classes, *Journal of Computational Chemistry* **30** (2009) 1414–1423.
53. K. C. Chou and H. B. Shen, ProtIdent: A web server for identifying proteases and their types by fusing functional domain and sequential evolution information, *Biochem Biophys Res Comm.* **376** (2008) 321–325.
54. H. B. Shen and K. C. Chou, Identification of proteases and their types, *Analytical Biochemistry* **385** (2009) 153–160.
55. K. C. Chou, A vectorized sequence-coupling model for predicting HIV protease cleavage sites in proteins, *J Biol Chem* **268** (1993) 16938–16948.
56. K. C. Chou, Review: Prediction of HIV protease cleavage sites in proteins, *Analytical Biochemistry* **233** (1996) 1–14.
57. H. B. Shen and K. C. Chou, HIVcleave: A web-server for predicting HIV protease cleavage sites in proteins, *Analytical Biochemistry* **375** (2008) 388–390.
58. K. C. Chou and H. B. Shen, Signal-CF: A subsite-coupled and window-fusing approach for predicting signal peptides, *Biochem Biophys Res Comm.* **357** (2007) 633–640.
59. H. B. Shen and K. C. Chou, Signal-3L: A 3-layer approach for predicting signal peptide, *Biochem Biophys Res Comm.* **363** (2007) 297–303.
60. K. C. Chou, Review: Structural bioinformatics and its impact to biomedical science, *Current Medicinal Chemistry* **11** (2004) 2105–2134.
61. R. Nair and B. Rost, Mimicking cellular sorting improves prediction of subcellular localization, *Journal of Molecular Biology* **348**(1) (2005) 85–100.
62. C. Guda, pTARGET, a web server for predicting protein subcellular localization, *Nucleic Acids Research* **34** (2006) W210–W213.
63. B. L. Lu and M. Ito, Task decomposition and module combination based on class relations: A modular neural network for pattern classification, *IEEE Trans. on Neural Networks* **10**(5) (1999) 1244–1256.
64. B. L. Lu, K. A. Wang, M. Utiyama and H. Isahara, A part-versus-part method for massively parallel training of support vector machines, *Proc. IEEE Intl. Joint Conf. on Neural Networks* (2004) 735–740.
65. Y. M. Wen, B. L. Lu and H. Zhao, Equal clustering makes min-max modular support vector machine more efficient, *Proc. the 12th Intl. Conf. on Neural Information Processing* (2006) 77–82.
66. J. Z. Wang, Z. Du, R. Payattakool, P. S. Yu and C. F. Chen, A new method to measure the semantic similarity of GO terms, *Bioinformatics* **23**(10) (2007) 1274–1281.
67. G. Karypis, CLUTO a Clustering Toolkit, Technical Report, Dept. of Computer Science, University of Minnesota (Storming Media, 2002).
68. L. J. McGuffin, K. Bryson and D. T. Jones, The PSIPRED protein structure prediction server, *Bioinformatics* **16**(4) (2000) 404–405.
69. J. Cheng, A. Z. Randall, M. J. Sweredoski and P. Baldi, SCRATCH, a protein structure and structural feature prediction server, *Nucleic Acids Research* **33**(Web Server Issue) (2005) W72–W76.
70. C. C. Chang and C. J. Lin, LIBSVM, a library for support vector machines, *Software available at http://www. csie. ntu. edu.tw/cjlin/libsvm* (2001).
71. M. T. A. Shamim, M. Anwaruddin and H. A. Nagarajaram, Support Vector Machine-based classification of protein folds using the structural properties of amino acid residues and amino acid residue pairs, *Bioinformatics* **23**(24) (2007) 3320–3327.
72. K. C. Chou and C. T. Zhang, Review: Prediction of protein structural classes, *Critical Reviews in Biochemistry and Molecular Biology* **30** (1995) 275–349.