A STUDY ON COMPONENT-BASED TECHNOLOGY FOR DEVELOPMENT OF COMPLEX BIOINFORMATICS SOFTWARE

ZURAINI ALI SHAH
SAFAAI DERIS
MUHAMAD RAZIB OTHMAN
ZALMIYAH ZAKARIA
PUTEH SAAD
ROHAYANTI HASSAN
MOHD HILMI MUDA
SHAHREEN KASIM
ROSFUZAH ROSLAN

FACULTY OF COMPUTER SCIENCE AND INFORMATION SYSTEMS
UNIVERSITI TEKNOLOGI MALAYSIA

2004

ABSTRACT

In the first chapter, entitled “Enhancement of Support Vector Machines for Remote Protein Homology Detection and Fold Recognition,” M. Hilmi Muda, Puteh Saad and Razib M. Othman present a comprehensive method based on two-layer multiclass classifiers. The first layer is used to detect up to superfamily and family in SCOP hierarchy, by using optimized binary SVM classification rules directly to ROC-Area. The second layer uses discriminative SVM algorithm with a state-of-the-art string kernel based on PSI-BLAST profiles that is used to leverage the unlabeled data. It will detect up to fold in SCOP hierarchy. They evaluated the results obtained using mean ROC and mean MRFP. Experimental results show that their approaches significantly improve the performance of protein remote protein homology detection for all three different datasets (SCOP 1.53, 1.67 and 1.73). They achieved 0.03% improvement in term of mean ROC in dataset SCOP 1.53, 1.17% in dataset SCOP 1.67 and 0.33% in dataset SCOP 1.73 when compared to the results produced by state-of-the-art methods.

In the second chapter “Hybrid Clustering Support Vector Machines by Incorporating Protein Residue Information for Protein Local Structure Prediction,” Rohayanti Hassan, Puteh Saad, and Razib M. Othman develop a predictive algorithm named R-HCSVM to predict protein local structure that works with following steps. Firstly, pre-process the input information for R-HCSVM. There are two types of input information needed namely protein residue score and protein secondary structure class. ResiduePatchScore information has been introduced as new method to pre-process protein residue score by combining protein conservation score that conserved rich functional information and protein propensity score that conserved rich secondary structural information. Hence, the protein residue score possess strength information that able to avoid bias scoring. Secondly, segment protein sequences into nine continuous length of protein subsequence. Next step which is highlighted another novel part in their study whereas a hybrid clustering SVM is introduced to reduce the training complexity. SOM and K-Means are integrated as a clustering algorithm to produce a granular input, while SVM is then used as a classifier. Based on the protein sequence datasets obtained from PISCES database, they found

that the R-HCSVM performs outstanding result in predicting protein local structure from a given protein subsequence compared to other methods.

In the third chapter "Incorporating Gene Ontology with Conditional-based Clustering to Analyze Gene Expression Data," Shahreen Kasim, Safaai Deris, and Razib M. Othman proposed a clustering algorithm named BTreeBicluster. The BTreeBicluster starts with the development of GO tree and enriching it with expression similarity from the Sacchromyces genes. From the enriched GO tree, the BTreeBicluster algorithm is applied during the clustering process. The BTreeBicluster takes subset of conditions of gene expression dataset using discretized data. Therefore, the annotation in the GO tree is already determined before the clustering process starts which gives major reflect to the output clusters. Their results of this study have shown that the BTreeBicluster produces better consistency of the annotation.

In the final chapter "Improving Protein-Protein Interaction Prediction by a False Positive Filtration Process," Rosfuzah Roslan and Razib M. Othman aimed to enhance the overlap between computational predictions and experimental results with the effort to partially remove the false positive pairs from the computational predicted PPI datasets. The usage of protein function prediction based on shared interacting domain patterns named PFP() for the purpose of aiding the Gene Ontology Annotation (GOA) is introduced in their study. They used GOA and PFP() as agents in the filtration process to reduce the false positive in computationally predicted PPI pairs. The functions predicted by PFP() which are in Gene Ontology (GO) IDs that were extracted from cross-species PPI data were used to assign novel functional annotations for the uncharacterized proteins and also as additional functions for those that are already characterized by GO. As known by them, GOA is an ongoing process and protein normally executes a variety of functions in different processes, so with the implementation of PFP(), they have increased the chances of finding matching function annotation for the first rule in the filtration process as much as 20%. Their results after the filtration process showed that huge sums of false positive pairs were removed from the predicted datasets. They used signal-to-noise ratio as a measure of improvement made by applying the proposed filtration process. While strength values were used to evaluate the applicability of the whole proposed computational framework to all the different computational PPI prediction methods.

TABLE OF CONTENTS

Enhancement of Support Vector Machines for Remote Protein Homology Detection and Fold Recognition

M. Hilmi Muda
Laboratory of Computational Intelligence and Biology
Faculty of Computer Science and Information Systems
Universiti Teknologi Malaysia, 81310
UTM Skudai, MALAYSIA
+607-5599230
mrhilmi@gmail.com

Puteh Saad
Department of Software Engineering
Faculty of Computer Science and Information Systems
Universiti Teknologi Malaysia, 81310
UTM Skudai, MALAYSIA
+607-5532344
puteh@utm.my

Razib M. Othman
Laboratory of Computational Intelligence and Biology
Faculty of Computer Science and Information Systems
Universiti Teknologi Malaysia, 81310
UTM Skudai, MALAYSIA
+607-5599230
razib@utm.my

ABSTRACT

Remote protein homology detection and fold recognition refers to detection of structural homology in proteins where there are small or no similarity in the sequence. The issues arise on how to accurately classify remote protein homology and fold recognition in the context of Structural Classification of Proteins (SCOP) hierarchy database and incorporate biological knowledge at the same time. Homology-based methods have been developed to detect protein structural classes from protein primary sequence information which can be divided into three types: discriminative classifiers, generative models for protein families and pairwise sequence comparisons. We present a comprehensive method based on two-layer multiclass classifiers. The first layer is used to detect up to superfamily and family in SCOP hierarchy, by using optimized binary SVM classification rules directly to ROC-Area. The second layer uses discriminative SVM algorithm with a state-of-the-art string kernel based on PSI-BLAST profiles that is used to leverage the unlabeled data. It will detect up to fold in SCOP hierarchy. We evaluated the results obtained using mean ROC and mean MRFP. Experimental results show that our approaches significantly improve the performance of protein remote protein homology detection for all three different datasets (SCOP 1.53, 1.67 and 1.73). We achieved 0.03% improvement in term of mean ROC in dataset SCOP 1.53, 1.17% in dataset SCOP 1.67 and 0.33% in dataset SCOP 1.73 when compared to the results produced by state-of-the-art methods.

Keywords

Fold recognition; Multiclass classifiers; Remote protein homology detection; Support vector machines; Two-layer classifiers.

1. INTRODUCTION

Advances in molecular biology in past years like large-scale sequencing and the human genome project, have yielded an unprecedented amount of new protein sequences. The resulting sequences describe a protein in terms of the amino acids that constitute it and no structural or functional protein information is available at this stage. To a degree, this information can be inferred by finding a relationship (or homology) between new sequences and proteins for which structural properties are already known. Traditional laboratory methods of protein homology detection depend on lengthy and expensive procedures like x-ray crystallography and nuclear magnetic resonance (NMR). Since using these procedures is unpractical for the amount of data available, researchers are increasingly relying on computational techniques to automate the process. Accurately detecting homologs at low levels of sequence similarity (remote protein homology detection) still remains a challenging ordeal to biologists. Remote protein homology detection refers to detection of structural homology in proteins where there are small or no similarity in the sequence. To detect protein structural classes from protein primary sequence information, homology-based methods have been developed, which can be divided into three types: discriminative classifiers [2,10,15,16,25], generative models for protein families [13,21] and pairwise sequence comparisons [1]. Discriminative classifiers show superior performance when compared to other methods [16,23].

Support Vector Machines (SVM) and Neural Networks (NN) are two popular discriminative methods. Recent studies showed that SVM has faster training speed, more accurate and efficient compared to NN [4]. This classifier is uniquely different from generative models and pairwise sequence comparisons because it removes the amino acid sequence from the prediction step. The protein sequences are transformed into feature vectors and then are used to train an SVM to identify protein families. Feature vectors give the benefit of mapping the sequences into a multivariate representation and additionally do not depend on a single pairwise score.

The performance of remote protein homology detection has been further improved through the use of methods that explicitly model the differences between the various protein families (classes) and build discriminative models. In particular, a number of different methods have been developed that build these discriminative models based on SVM and have shown, provided there are sufficient data for training, to produce results that are in general superior to those produced by pairwise sequence comparisons or methods based on generative models [2,10,15,16,25], [15,7].

Motivated by positive results from Rangwala and Karypis [24] and Ie et al. [9], we further study the problem of building SVM based multi-class classification models for remote protein homology detection in the context of the Structural Classification of Proteins (SCOP) [21] protein classification scheme. We present a comprehensive method based on two layers multiclass classifiers. The first layer can detect up to superfamily and family in SCOP hierarchy by using optimized binary SVM classification rules directly to ROC-Area. The second layer of multiclass classifier uses discriminative SVM algorithm with a state-of-the-art string kernel based on PSI-BLAST profiles to leverage unlabeled data. This will detect up to fold in SCOP hierarchy.

Details are explained in the methods section. We evaluated our result using mean ROC and mean RFP. Experimental results show that our approaches significantly improve the performance of protein remote protein homology detection.

2. METHODS

In this section, we will briefly explain our proposed method named SVM-2L to build two layers multiclass classifiers. Based on idea of Lorena and Carvalho [19] we tuned SVM's parameters in our first layer multiclass classifier to influence their performance. They are the value of the regularization constant, *C* and kernel type, with its respective parameter. With the combination of the second layer multiclass classifier which uses the SVM with improved kernel based on PSI-BLAST profiles to leverage unlabeled data, it is expected to improve performance of remote protein homology detection and fold recognition by adding elements without overfitting. The overall steps of the SVM-2L is as shown in Figure 1.

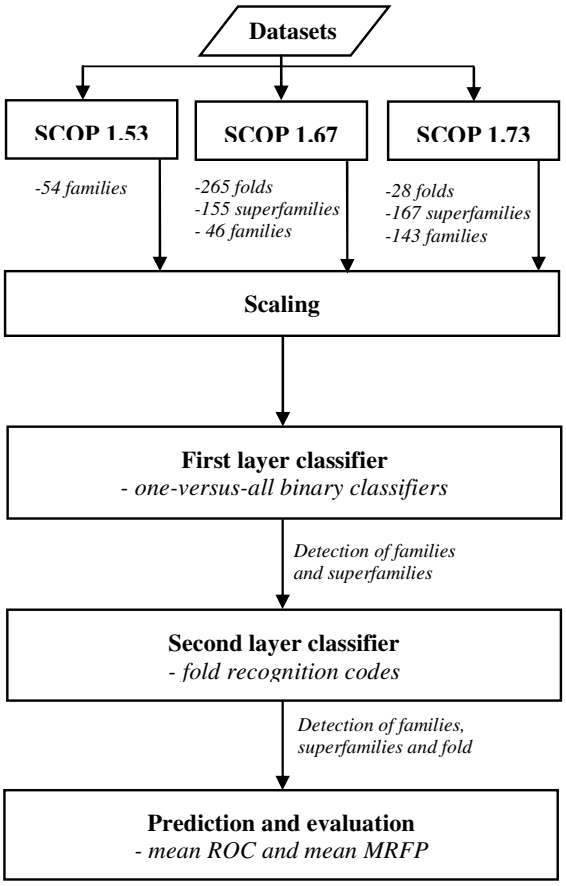


Figure 1. Overall steps to build the classifier.

2.1 Experimental Datasets

We evaluated the performance of our method using three datasets. The first dataset, SCOP version 1.53, we emulate the benchmark procedure presented by Liao and Noble [18]. The data consist of 4352 sequences extracted from the Astral [4] database grouped into families and superfamilies. For each family, the protein domains within the family are considered positive test examples, and protein domains within the superfamily but outside the family are considered positive training examples. This yields 54 families with at least 10 positive training examples and five positive test examples. Negative examples for the family are chosen from outside of the positive sequences fold, and were randomly split into training and test sets in the same ratio as the positive example.

Second dataset are derived from SCOP version 1.67 created by Rangwala and Karypis [25]. Datasets fd25 were designed to evaluate the performance of fold recognition and were derived by taking only the domains with less than 25% pairwise sequence identity, respectively. This set of domains was further reduced by keeping only the domains belonging to folds that contained at least three superfamilies and at least three of these superfamilies contained more than three domains. For fd25, the resulting dataset contained 1294 domains organized in 265 folds, 155 superfamilies and 46 families.

We also tested our method on the latest version dataset from SCOP version 1.73. We follow the filtering step by Rangwala and Karypis [25] to select the dataset, which results 1597 domains organized in 28 folds and 167 superfamilies. We derived the dataset by taking only the domains with less than 95% and 40% pairwise sequence identity according to Astral database. This set of domain was further reduced by keeping only the domains belonging to fold that contained at least 3 superfamilies, and one of these superfamilies contained multiple families.

Dataset SCOP 1.53 contains superfamilies and families only, while datasets SCOP 1.67 and dataset SCOP 1.73 contains up to folds.

2.2 Scaling

Scaling the datasets before applying SVM is essential. The main advantage is to avoid attribute in greater numeric ranges dominate those in smaller numeric ranges. Other than that, it is also used to avoid numerical difficulties during the calculations. Because kernel values usually depends on the inner products of feature vectors, e.g. the linear kernel and the polynomial kernel in which large attribute values might result in numerical problems. We linearly scale each attribute to the range [-1, 1] [19]. Testing and training datasets must obviously be scaled using the same method.

Suppose to scale a certain attribute of training dataset from $[y_{\min}, y_{\max}]$ to $[y'_{\min}, y'_{\max}]$, where y is the raw attribute value of training or testing datasets. The scaled value is obtained from Zheng et al. [34] as follows

$$y' = y'_{\min} + \frac{y'_{\max} - y'_{\min}}{y_{\max} - y_{\min}}(y - y_{\min}) . \quad (1)$$

2.3 First Layer Classifiers

The various one-versus-all binary classifiers were constructed using SVM. One of the implementations is SVMstruct [14] that train conventional linear classification SVM optimizing error rate in time that is linear in the size of the training data through an alternative, but equivalent formulation of the training problem. It implements the alternative structural formulation of the SVM optimization problem for conventional binary classification with error rate and ordinal regression. Moreover, SVMstruct used small memory (15500 Kilobytes) resource when training large set of data, which make it more efficient [21]. We used the formulation of the SVM optimization problem by Joachims [20] that provides the basis of our algorithm, both for classification and for ordinal regression SVM.

2.3.1 Classification

For a given training dataset $(x_1,y_1),...,(x_n,y_n)$ with n length, $x_i \in \Re^N$ where $\Re^N$ a radical power of large of features, N is the large number of features, y is stated as $y_i \in \{-1,+1\}$ training a binary classification SVM means solving the optimization problem [13]. For simplicity of the theoretical results (Eq. 2), we focus on classification rules $h_w(x)=\sin(w^T x+b)$ with $b=0$, where w is the empty stack of constraints, T is the iterations and b is regression loss. A non-zero b can easily be modeled by adding an additional feature of constant value to each x.

$$\min_{w,\xi_i \geq 0} \frac{1}{2} w^T w + \frac{C}{n} \sum_{i=1}^{n} \xi_i \,, \tag{2}$$

where $\forall i \in \{1,...,n\}: y_i (w^T x_i) \geq 1-\xi_i$.

We adopted the formulation of [30], [33] where sum of linear slack variables, $\Sigma \xi_i$ is divided by n length to better capture how trade-off between training error and margin, C, scales with the training set size. The Eq. 3 in the following considers a different optimization problem, which was proposed for training SVM to predict structured outputs as been done by Tsochantaridis et al. [30].

$$\min_{w,\xi \geq 0} \frac{1}{2} w^T w + C\xi \,, \tag{3}$$

where $\forall c \in \{0,1\}^n : \frac{1}{n} w^T \sum_{i=1}^{n} c_i y_i x_i \geq \frac{1}{n} \sum_{i=1}^{n} c_i - \xi$.

While Eq. 3 has 2^n constraints, one of each possible vector $c=(c_1,...,c_n)$ $(x_1,y_1),...,(x_n,y_n)$, it only has one slack variable ξ that is shared across all the constraints. Each constraint in this equation corresponds to the sum of a subset of constraints from Eq. 1, and the c_i select the subset. $\frac{1}{n} \sum_{i=1}^{n} c_i$ can be seen as the maximum fraction of training errors possible over each subset and ξ is an upper bound on the fraction of training errors made by h_w .

2.3.2 Ordinal Regression

In an example (x_i,y_i), the label y_i indicates a rank instead of a nominal class in ordinal regression. We let $y_i \in \{1,...,Z\}$ with Z length, so that the values $1,...,Z$ are related on an ordinal scale, without loss of generality. The goal is to learn a function $h(x)$ so that for many pair of examples x_i, y_i and x_j, y_j it holds that

$$h(x_i) > h(x_j) \Leftrightarrow y_i > y_j \,. \tag{4}$$

Given a training dataset $(x_1,y_1),...,(x_n,y_n)$ with $x_i \in \Re^N$ and $P=\{(i,j): y_i > y_j\}$, formulate the ordinal regression SVM (Eq. 5). Denote with P the set of pairs (i,j) for which example i has a higher rank than example j, i.e. $P=\{(i,j): y_i > y_j\}$, and let $m=|P|$.

$$\min_{w,\xi_{ij} \geq 0} \frac{1}{2} w^T w + \frac{C}{m} \sum_{(i,j) \in P} \xi_{ij} \,, \tag{5}$$

where $\forall (i,j) \in P: (w^T x_i) \geq (w^T x_j) + 1 - \xi_{ij}$.

These formulations find a large margin linear function $h(x)$, which minimizes the number of pairs of training examples that are swapped with respect to their desired order. As in other classification, Eq. 5 is a convex quadratic program. Ordinal regression problems have applications in learning retrieval functions for search engines [7, 27, 29]. Furthermore, if the labels y takes only two values, Eq. 5 optimizes the ROC-Area of the classification rule.

2.4 Second Layer Classifiers

We used profile-based string kernel SVM that are trained to perform binary classifications on the fold and superfamily levels of SCOP as a base for our multi-class protein classifiers. The profile kernel defined as a function that is used to measure the similarity of two protein sequence profiles based on their representation in a high-dimensional vector space indexed by all k-mers (k-length subsequences of amino acids).

Binary one-vs-the-rest SVM classifiers that are trained to recognize individual structural classes yield prediction scores that are incomparable, so that standard "one-vs-all" classification performs sub optimally when the number of classes is very large, as in this case. We used fold recognition codes that learn relative weights between one-vs-the-rest classifiers and further, encode information about the protein structural hierarchy for multi-class prediction, as to deal with this challenging problem. In large scale benchmark results based on the SCOP database, our method significantly improves on the prediction accuracy of both a baseline use of PSI-BLAST and the standard one-vs-all method.

The use of profile-based string kernels is an example of semi-supervised learning, since unlabeled data in the form of a large sequence database is used in the discrimination problem. Moreover, profile kernel values can be efficiently computed in time that scales linearly with input sequence length. Equipped with such a kernel mapping, one can use SVM to perform binary protein classification on the fold level and superfamily level.

2.4.1 Fold Recognition Code

Suppose that we have trained q fold detectors. Then, for a protein sequence x, we form a prediction discriminant vector $\vec{f}(x)=(f_1(x),...,f_q(x))$. The simple one-versus-all prediction rule for multi-class fold prediction is $\hat{y}=\arg\max_j f_j(x)$. The problem with this prediction rule is that the discriminant values produced by the different SVM classifiers are not necessarily comparable. We used an approach by learning the optimal weighting for a set of classifiers, scaling their discriminant values and making them more readily comparable. To fit the training datasets, we adapt the coding system by learning a weighting of the code elements (or classifiers). The final multi-class prediction rule is $\hat{y}=\arg\max_j (W * \vec{f}(x)).K_j$, where * denotes the component-wise multiplication between vectors and W is a weight vector.

2.5 Evaluation Measures

To assess the performance of a remote protein homology detection method, we consider two metrics: the Receiver Operating Characteristics (ROC) and median Rate of False Positives (RFP). ROC is a sophisticated technique that is used to evaluate the results of a prediction, for visualizing, organizing and selecting classifiers based on their performance. The performances in our method are measured on how precise the detection and classification of the sequence to its correct group.

The ROC curve is obtained by plotting the True Positive Rate (TPR) against the False Positive Rate (FPR), for the entire range of possible cutoff values, *c*. On this plot, the line through the origin with slope 1 would correspond to the performance of a similarity detection based on a random similarity score. A method which detects SCOP similarity better than randomly must show a ROC curve situated above this diagonal.

MRFP is a RFP median value of each protein sequences grouped in several families. Mean MRFP is MRFP average value for entire set of protein sequences families. The MRFP is bounded by 0 and 1 and is used to measure the error rate of the prediction under the score threshold where half of the true positives can be detected. These measures are used for evaluation cited in [12, 18].

3. Results and Discussion

As discussed in the introduction section, our research in this paper is motivated by the idea and work from Rangwala and Karypis [26] and Ie et al. [10], by which they solve the classification problem in the context of remote homology detection and fold recognition. Based on their work, we presented a two-layer multiclass classifiers approach called SVM-2L. We compare our method with other eight different methods: SVM Struct [30], SVM-Fold [22], SVM-Pairwise [19], SVM-Fisher [11], SVM-HMMSTR [34], SVM-Ngram-LSA [6], SVM-Pattern-LSA [6] and SVM-Motif-LSA [31] that already has been used to detect remote protein homology. The performance of various schemes in term of mean ROC and mean RFP is shown in Table 1(a) and Table 1(b) respectively for remote protein homology detection using standard benchmark dataset, SCOP 1.53. We split the results to the group of family and superfamily. The result of SVM-Pairwise, SVM-Fisher and SVM-HMMSTR are retrieved from [34]. We use publicly available SVM-Motif-LSA to search sequence databases for matches to motifs. Based on our results on mean ROC in Table 1, it shows that our proposed method significantly outperforms existing state-of-the-art methods. Comparison of results by group of family and group of superfamily also clearly shows that our proposed methods are really efficient. This scenario is influenced by the use of large margin SVM classifier and its discriminative approach that we implemented in our framework. We find out that some of these results agree with previous assessments. For example, the relative performance of SVM-Fisher agrees with the results given by Jaakkola et al. [32]. Although in that work the difference was more pronounced and relative performance of SVM-Pairwise results given in [8].

Table 1: Mean ROC (a) and mean MRFP (b) for different methods for family and superfamilies using SCOP 1.53 dataset.

(a)

Method	Family	Superfamily	Overall
SVM-2L	0.9998	0.9976	0.9345
SVM Struct	0.8987	0.9521	0.8543
SVM-Fold	0.9458	0.9424	0.9342
SVM-Pairwise	0.4380		0.4380
SVM-Fisher	0.4370		0.4370
SVM-HMMSTR	0.6400		0.6400
SVM-Ngram-LSA	0.8929	0.8992	0.9132
SVM-Motif-LSA	0.9995	0.9897	0.9335
SVM-Pattern-LSA	0.9964	0.9925	0.9264

(b)

Method	Family	Superfamily	Overall
SVM-2L	0.0012	0.0019	0.0015
SVM Struct	0.0060	0.0002	0.0031
SVM-Fold	0.0018	0.0008	0.0013
SVM-Fisher	0.0963	0.0096	0.0486
SVM-Pairwise	0.1173		0.1173
SVM-HMMSTR	0.0380		0.0380
SVM-Ngram-LSA	0.1017		0.1017
SVM-Motif-LSA	0.9953		0.9953
SVM-Pattern-LSA	0.0703		0.0703

Table 2: Mean ROC (a) and mean MRFP (b) for different methods for family and superfamilies using SCOP 1.67 dataset.

(a)

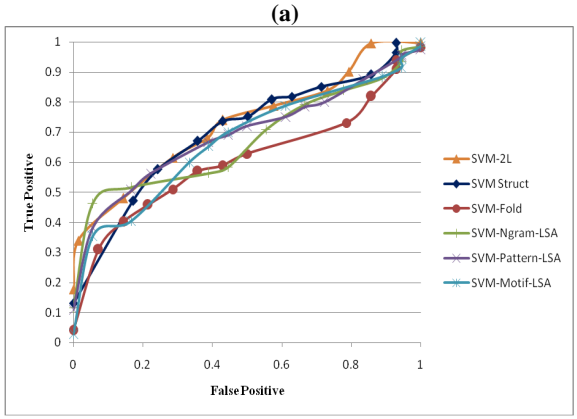
Method	Family	Superfamily	Fold	Overall
SVM-2L	0.9987	0.9991	0.9876	0.9951
SVM Struct	0.9458	0.9867	0.9753	0.9692
SVM-Fold	0.9532	0.9986	0.9986	0.9834
SVM-Ngram-LSA	0.9038	0.9645	0.9856	0.9513
SVM-Motif-LSA	0.8973	0.9979	0.9884	0.9612
SVM-Pattern-LSA	0.9234	0.9753	0.9981	0.9656

(b)

Method	Family	Superfamily	Fold	Overall
SVM-2L	0.00056	0.00087	0.00065	0.00208
SVM Struct	0.00063	0.00065	0.00074	0.00202
SVM-Fold	0.00087	0.00045	0.00053	0.00185
SVM-Ngram-LSA	0.00722	0.00056	0.00062	0.00840
SVM-Motif-LSA	0.00066	0.00076	0.00034	0.00176
SVM-Pattern-LSA	0.00099	0.00063	0.00024	0.00186

We achieve a significant result of our proposed method on dataset SCOP 1.67, which is specially created for this research to detect fold. Our result as shown in Table 2 shows higher mean ROC compared with other state-of-the-art methods. Figure 2 (a) and Figure 2 (b) illustrate the ROC and RFP curve. Using our proposed method, we are able to improve about 1.17% from the current result. This happened as the effect of tuned the SVM's parameters, which is the value of the regularization constant, *C* in our first layer multiclass classifier to prevent overfitting. We only compare our proposed method with five methods, which are SVM Struct, SVM-Fold, SVM-Ngram-LSA, SVM-Motif-LSA and SVM-Pattern-LSA. This is because the source code for SVM-Pairwise, SVM-Fisher and SVM-HMMSTR are no longer available and we manage to get only the result that those methods produced.

(a)



(b)

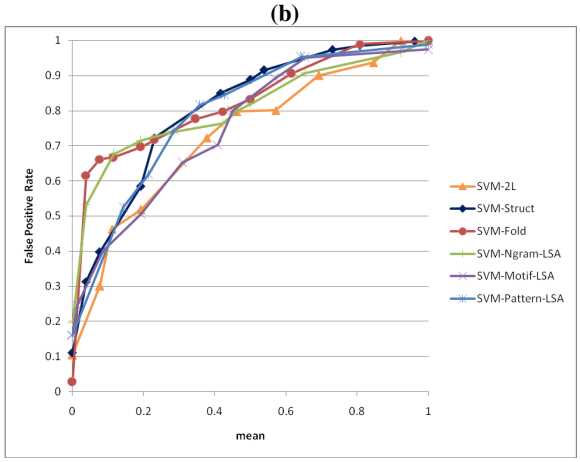


Figure 2. Curve of Mean ROC (a) and mean MRFP (b) for dataset SCOP1.67.

For dataset SCOP 1.73, we achieve improvement of 0.14% which is depicted in Table 3(a) and Table 3(b). The mean ROC of our methods improves from state-of-the-art methods as depicted in Figure 3(a) and Figure 3(b). Although, there is only a slight improvement, however our proposed method demonstrates a stable performance. This is the impact of using the fold detection codes which encodes information about the protein structural hierarchy for multi-class detection and the repetition of cross validation process in the first layer method. Meanwhile, in mean RFP result, our proposed method contributes 0.0072% better when compared to results produced by SVM Struct. When it is tested on dataset SCOP 1.73, it produces a lower error rate, as shown in good result in median rate of false positive in Table 3(b).

Table 3: Mean ROC (a) and mean MRFP (b) for different methods for family and superfamilies using SCOP 1.73 dataset.

(a)

Method	Family	Superfamily	Fold	Overall
SVM-2L	0.9118	0.8329	0.8295	0.9019
SVM Struct	0.8897	0.8495	0.8390	0.8871
SVM-Fold	0.8952	0.8952	0.9363	0.8295
SVM-Ngram-LSA	0.8746	0.8871	0.8615	0.8481
SVM-Motif-LSA	0.8592	0.8826	0.8273	0.8733
SVM-Pattern-LSA	0.8794	0.8979	0.8798	0.8986

(b)

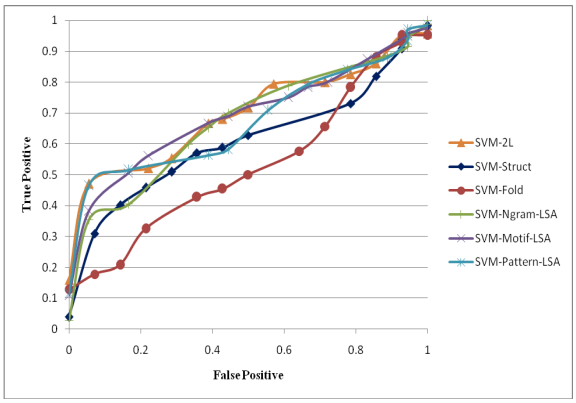
Method	Family	Superfamily	Fold	Overall
SVM-2L	0.0386	0.0563	0.1443	0.0238
SVM Struct	0.0342	0.1724	0.1366	0.0304
SVM-Fold	0.1136	0.0967	0.0945	0.0303
SVM-Ngram-LSA	0.1390	0.1764	0.1157	0.0386
SVM-Motif-LSA	0.1411	0.1515	0.2075	0.0495
SVM-Pattern-LSA	0.1157	0.1600	0.4814	0.0437

From stability of the curve of mean ROC and mean RFP in Figure 2 and Figure 3, we can conclude that our proposed method produced a stable result for all datasets. Even though for some point the curves show a low result, however it produces a positive effect to the result. Other than that, our method is consistent for all datasets. In summary, overall result from our method shows more than 0.9 in the term of mean ROC for all three different experimental datasets. We achieved 0.03% improvements in dataset SCOP 1.53, 1.17% in dataset SCOP 1.67 and 0.33% in dataset SCOP 1.73 when compared to the result produced by state-of-the-art methods.

4. Conclusion

This paper demonstrate that the performance of remote protein homology detection and fold recognition has been further improved through the use of methods that explicitly model the differences between the various protein families (classes) and build discriminative models. We also presented a comprehensive method for detection of remote protein and fold recognition based on two layers multiclass classifiers. Our first layer is only capable to detect family and superfamily in SCOP hierarchy by using optimizes binary SVM classification rules directly to ROC-Area. The second layer of multiclass classifier that is capable to detect up to fold in SCOP hierarchy uses discriminative SVM algorithm

(a)



(b)

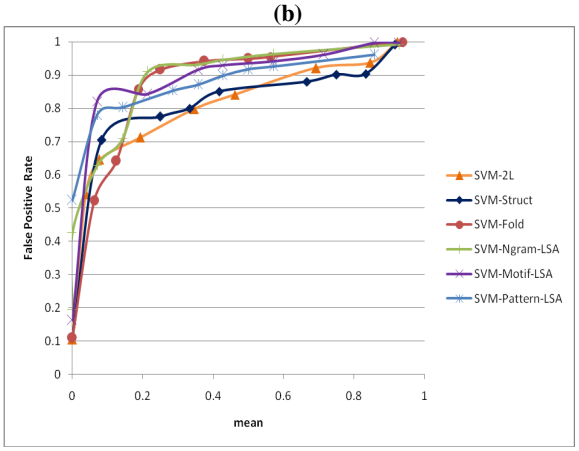


Figure 3. Curve of Mean ROC (a) and mean MRFP (b) for dataset SCOP1.73.

with a state-of-the-art string kernel based on PSI-BLAST profiles to leverage unlabeled data. A number of different methods have been developed that build these discriminative models based on SVM and have shown, provided there are sufficient data for training, to produce results that are in general superior to those produced by pairwise sequence comparisons or methods based on generative models. The result produced by our method also shows good improvements in all three different datasets. In the future, we intend to enhance our method by using the realignment approach that will correct misalignments between a sequence and the rest of profile. Other than that, implementation of other kernel functions in SVM classifiers is hypothesized to improve the performance of remote protein homology detection and fold recognition, since different kernel function corresponds to different input.

5. ACKNOWLEDGMENTS

This project is funded by Malaysian Ministry of Higher Education (MOHE) under Fundamental Research Grant Scheme (project no 78092).

6. REFERENCES

[1] Altschul, S. F., Gish, W., Miller, W., Myers, E. W., and Lipman, D. J. 1990. A Basic Local Alignment Search Tool. Journal of Molecular Biology 215 (3), 403-410.

[2] Andreeva, A., Howorth., D., Chandonia, J., Brenner, S., Hubbard, T., Chothia C., and Murzin, A. (2008). Data growth and its impact on the SCOP database: new developments. Nucleic Acids Research. 36 (1), 419-425.

[3] Ben-Hur, A and Brutlag D. 2003. Remote homology detection: a motif based approach. In Proceedings of the International Conference on Intelligent Systems for Molecular Biology (Brisbane, Australia, June 29-July 3, 2003).

[4] Brenner, S., Koehl, P. and Levitt, M. 2000. The ASTRAL compendium for sequence and structure analysis. Nucleic Acids Research. 28 (1), 254-256.

[5] Ding, C.H.Q., and Dubchak, I. 2001. Multi-class protein fold recognition using support vector machines and neural networks. Bioinformatics. 17 (4), 349-358.

[6] Dong, Q., Wang, X., Lin, L. 2006. Application of latent semantic analysis to protein remote homology detection. Bioinformatics. 22 (3), 285-290.

[7] Haoliang, Q., Sheng, L., Jianfeng, G., Zhongyuan, H. and Xinsong, X. (2008) Ordinal regression for information retrieval. Journal of Electronics (China). 25 (1), 120-124.

[8] Hou, Y., Hsu, W., Lee, M.L. and Bystroff, C. 2003. Efficient remote homology detection using local structure. Bioinformatics. 19 (17), 2294-2301.

[9] Hsu, C.W., Chang, C.C. and Lin, C.J. A practical guide to support vector classification, 2008. <http://csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf >.

[10] Ie, E., Weston, J., Noble, W.S. and Leslie, C. 2005. Multi-class protein fold recognition using adaptive codes, In Proceedings of the International Conference on Machine Learning (Bonn, Germany, August 7-11, 2005). ACM Press, New York, NY, 329-336. DOI = http://doi.acm.org/10.1145/1102351.1102393

[11] Jaakkola, T., Diekhans, M. and Haussler, D. 1999. Using the Fisher kernel method to detect remote protein homologies, In Proceedings of the International Conference on Intelligent Systems for Molecular Biology (August 6–10, 1999, Heidelberg, Germany). AAAI Press, 149-158.

[12] Jaakkola, T., M. Diekhans and D. Haussler. 2000. A discriminative framework for detecting remote protein homologies. Journal of Computational Biology. 7 (1-2), 95-114.

[13] Joachims, T. 2005. A support vector method for multivariate performance measures, In Proceedings of the International Conference on Machine Learning, (Bonn, Germany, 7-11 August, 2005). ACM Press, New York, NY, 377-384.

[14] Joachims, T. 2006. Training linear SVMs in linear time, In Proceedings of the ACM Conference on Knowledge Discovery and Data Mining (Philadelphia, USA, 20-23 August, 2006). ACM Press, New York, NY, 217-226. DOI= http://doi.acm.org/10.1145/1150402.1150429

[15] Krogh, A., Brown, M., Mian, I.S., Sjölander, K. and Haussler, D. 1994. Hidden Markov Models in Computational Biology: Applications to Protein Modeling. Journal of Molecular Biology. 235 (5), 1501-1531.

[16] Kuang, R., Ie, E., Wang, K., Siddiqi, M., Freund, Y. and Leslie, C. 2005. Profile kernels for detecting remote protein homologs and discriminative motifs, Journal of Bioinformatics and Computational Biology. 13, 21-23.

[17] Leslie, C., Eskin, E., Cohen, A., Weston, J., and Noble, W.S. 2004. Mismatch String Kernels for Discriminative Protein Classification. Bioinformatics. 20 (4), 467-476.

[18] Liao, L. and Noble, W.S. 2002. Combining pairwise sequence similarity and support vector machines for remote protein homology detection. In Proceedings of the Annual International Conference on Research in Computational Molecular Biology (Washington, USA, 18-21 April, 2002). ACM Press, New York, NY, 225-232. DOI= http://doi.acm.org/10.1145/1150402.1150429

[19] Liao, L. and Noble, W.S. 2003. Combining pairwise sequence similarity and support vector machines for detecting remote protein evolutionary and structural relationships. Journal of Computational Biology 10 (6), 857-868.

[20] Lorena, A. C. and Carvalho, A.C.P.L.F.d. 2008. Evolutionary tuning of SVM parameter values in multiclass problems. Neurocomputing. 71 (16-18), 3326-3334.

[21] Mangasarian, O. and Musicant, D. 2001. Lagrangian support vector machines. Journal of Machine Learning Research. 1 (1), 161-177.

[22] Melvin, I., Ie, E., Kuang, R., Weston, J., Stafford, N. and Leslie, C. 2007. SVM-Fold: a tool for discriminative multi-class protein fold and superfamily recognition. BMC Bioinformatics. (8:S2).

[23] Murzin, A. G., Brenner, S.E., Hubbard, T., and Chothia, C. 1995. SCOP: a structural classification of proteins database for the investigation of sequences and structures. Journal of Molecular Biology. 247 (4), 536-540.

[24] Park, J., Karplus, K., Barrett, C., Hughey, R., Haussler, D., Hubbard, T. and Chotia, C. 1998. Sequence comparisons

using multiple sequences detect twice as many remote homologues as pairwise methods. Journal of Molecular Biology. 284 (4), 1201-1210.

[25] Rangwala, H. and Karypis, G. 2005. Profile based direct kernels for remote homology detection and fold recognition. Bioinformatics. 21 (23), 4239-4247.

[26] Rangwala, H. and Karypis, G. 2006. Building multiclass classifiers for remote homology detection and fold recognition. BMC Bioinformatics. (7) 455.

[27] Runarsson, T.P. 2006. Ordinal Regression in Evolutionary Computation. Springer Berlin.

[28] Saigo, H., Vert, J. P., Ueda, N. and Akutsu, T. 2004. Protein homology detection using string alignment kernels. Bioinformatics. 20 (11), 1682-1689.

[29] Sch¨olkopf, B., Smola, A.J., Williamson, R.C. and Bartlett, P.L. 2000. New support vector algorithms. Neural Computation. 12 (5), 1207-1245.

[30] Schoelkopf, B. and Smola, A.J. 2002. Learning with kernels. MIT Press.

[31] Timothy, L. B., Nadya, W, Chris, M. and Wilfred, W.L. 2006. MEME: discovering and analyzing DNA and protein sequence motifs. Nucleic Acids Research. (34), 369-373.

[32] Tsochantaridis, I., Hofmann, T., Joachims, T. and Altun, Y. 2004. Support Vector Learning for Interdependent and Structured Output Space. In Proceedings of the International Conference on Machine Learning (Banff, Alberta, Canada, 4-8 July, 2004). ICML'04. ACM Press, New York, NY, 823-830. DOI= http://doi.acm.org/10.1145/1015330.1015341

[33] Tsochantaridis, I., Hofmann, T., Joachims, T. and Altun, Y. 2005. Large margin methods for structured and interdependent output variables. Journal of Machine Learning Research. (6), 1453–1484.

[34] Yuna, H., Wynne, H, Lee, M. L. and Bystroff, C. 2004. Remote homolog detection using local sequence-structure correlations. PROTEINS: Structure, Function, and Bioinformatics. 57 (3), 518-530.

[35] Zheng, S., Tang, H., Han, Z. and Zhang, H. 2006. Solving large-scale multiclass learning problems via an efficient support vector classifier. Journal of Systems Engineering and Electronics. 17 (4), 910-915.

HYBRID CLUSTERING SUPPORT VECTOR MACHINES BY INCORPORATING PROTEIN RESIDUE INFORMATION FOR PROTEIN LOCAL STRUCTURE PREDICTION

Rohayanti Hassan
Laboratory of Computational Intelligence and Biology
Universiti Teknologi Malaysia, 81310 UTM Skudai, MALAYSIA
+607-5599230
rohayanti@utm.my

Puteh Saad
Department of Software Engineering
Faculty of Computer Science and
Universiti Teknologi Malaysia, 81310 UTM Skudai, MALAYSIA
+607-5599230
puteh@utm.my

Razib M. Othman
Laboratory of Computational Intelligence and Biology
Universiti Teknologi Malaysia, 81310 UTM Skudai, MALAYSIA
+607-5599230
razib@utm.my

ABSTRACT
Protein local structure prediction can be described as prediction of protein secondary structure from protein subsequence. This protein subsequence or also known as protein local structure covers fragments of the protein sequence. In fact, it is easier to identify the sequence-to-secondary structure relationship using protein subsequence rather than use the whole protein sequence. Further, this relationship can be used to infer new protein fold, protein function and detect protein remote homolog. Due to its significance, a predictive algorithm named R-HCSVM is developed to predict protein local structure that works with following steps. Firstly, pre-process the input information for R-HCSVM. There are two types of input information needed namely protein residue score and protein secondary structure class. ResiduePatchScore information has been introduced as new method to pre-process protein residue score by combining protein conservation score that conserved rich functional information and protein propensity score that conserved rich secondary structural information. Hence, the protein residue score possess strength information that able to avoid bias scoring. Secondly, segment protein sequences into nine continuous length of protein subsequence. Next step which is highlighted another novel part in this study whereas a hybrid clustering SVM is introduced to reduce the training complexity. SOM and K-Means are integrated as a clustering algorithm to produce a granular input, while SVM is then used as a classifier. Based on the protein sequence datasets obtained from PISCES database, it is found that the R-HCSVM performs outstanding result in predicting protein local structure from a given protein subsequence compared to other methods.

Keywords
Protein local structure prediction, protein secondary structure, protein residue score, SOM K-Means, Support Vector Machines.

1. INTRODUCTION
Prediction of protein secondary structure using protein local structure has shown promising improvements [3], [41], [42]. Protein local structure primarily made up from segments of amino acid. In another words, protein local structures are also called as protein subsequence, protein fragments by Chen et al. [4], protein segments by Zhong et al. [41] and Zhong et al. [42] or protein local structural motifs by Karchin et al. [15] and Karchin et al. [16]. This protein local structure coded all information of native structure of a protein such as hydrophobicity, hydrophilicity, electrostatic and hydrogen bonds interaction. Furthermore, information or called knowledge of this protein local structure can be used to infer how the protein interacts with other molecules, predict its structure as well as function. In fact, this knowledge facilitates to drug design. For example, Hu and Hu [12] aimed at designing small-molecule compounds that restore the normal function of p53-MDM2 (two protein targets in cancer research) and consequently reduce or eliminated certain forms of specific cancer.

Indeed, supervised machine learning based method for protein local structure prediction have shown strong generalization capability in handling nonlinear classification such as works done using Support Vector Machine (SVM) [18], Neural Network (NN) [21] and Hidden Markov Model (HMM) [23]. Nevertheless, it is not favorable for large-scale datasets due to the convex quadratic programming property which is NP-complete in the worst case.

As a consequence, the training process will become decelerate. Several techniques have been proposed in order to solve this training complexity problem, for instance including chunking method [34], osuna decomposition method [26] and sequential minimal optimization method [29]. However, these techniques do not scale well of the training datasets. In related work, several techniques including random selection [2], bagging [37] and clustering analysis [35] are used as dataset selection to reduce the number of training datasets in order to accelerate the training process. Yet, the performance of training process is greatly depends on training datasets selection that may cause significance datasets are being overlooked. As a result, by decomposing a large-scale datasets into series of smaller datasets using clustering algorithm [19], [23], the training complexity can be reduced without overlook the significance dataset.

2. MOTIVATION
Determination of protein local structure by experimental methods such as X-ray crystallography, Nuclear Magnetic Resonance (NMR) and electron microscopy are tedious and expensive process such as done by Pauling et al. [27] who discovered H structure and Pauling and Corey [28] who discovered C structure. In fact, this method often involves difficulties inherent in protein synthesis, purification and crystallization which resulting to inaccurate assignment of protein residue to the corresponding secondary structural class. Consequently, many wet-lab methods have been developed by researcher and biologist to predict protein

local structure accurately such as works done by Levitt and Chotia [20] who first proposed to classify thirty one globular proteins into four structural classes. In 1990's, Liu and Chou [22] improved the definitions of structural classes by increasing the size of associated protein regions. In another related works, Wu and Kabat [39], Shenkin et al. [36] and Karlin and Brocchieri [17] have came up with varies method to quantify the residue conservation score in order to determine protein local structure accurately. However, these wet-lab methods needed a long time to execute all experiments and cost consuming.

Difficulties of determining protein local structures experimentally motivates researcher to come up with computational method. Machine learning algorithm is another dimension of computational method to predict protein local structure. On one hand, the superior of this method is depend on the information is being supplied and learned. Basically, there are two types of information are needed for this method to execute. One is known as feature vector and another one is known as feature class. Feature vector in a form of numerical value is represented by protein residue score. Meanwhile, feature class in a form of nominal value is represented by protein secondary structure class. A superior method is desired to pre-process these two features in order to ensure they are reliable and possess strong information. For instance, in order to quantify the protein residue score, it has to avoid from bias scoring as a result of sequence redundancy without losing the important evolutionary and structural information. Protein residue score can be quantified using propensity score that based on the proportion of predominant secondary structure such as done in Levitt and Chotia [20], Chou and Fasman [6] and Constantini et al. [7]. Recently, most protein residue score is quantified through Multiple Sequence Alignment (MSA) process that based on its evolutionary information which is more conserved functional information. This type of protein residue score also known as protein conservation score and example works such as done by Sander and Scheneider [32], Mirny and Shakhnovich [24] and Goldenberg et al. [9].

Recently, progress has been made in protein local structure prediction method in order to address several issues. Sander et al. [33] proposed two types of discriminative models for protein local structure prediction which are hybrid K-Means with SVM and hybrid K-Means with Random Forest (RF) in order to reduce the training complexity. Nevertheless, the proposed hybrid K-Means with RF may decelerate the training process as a result of randomly sampling the training dataset. Furthermore, the proposed hybrid K-Means with SVM which also has been proposed by Zhong et al. [42], suffered from poor initialization method to form a quality cluster.

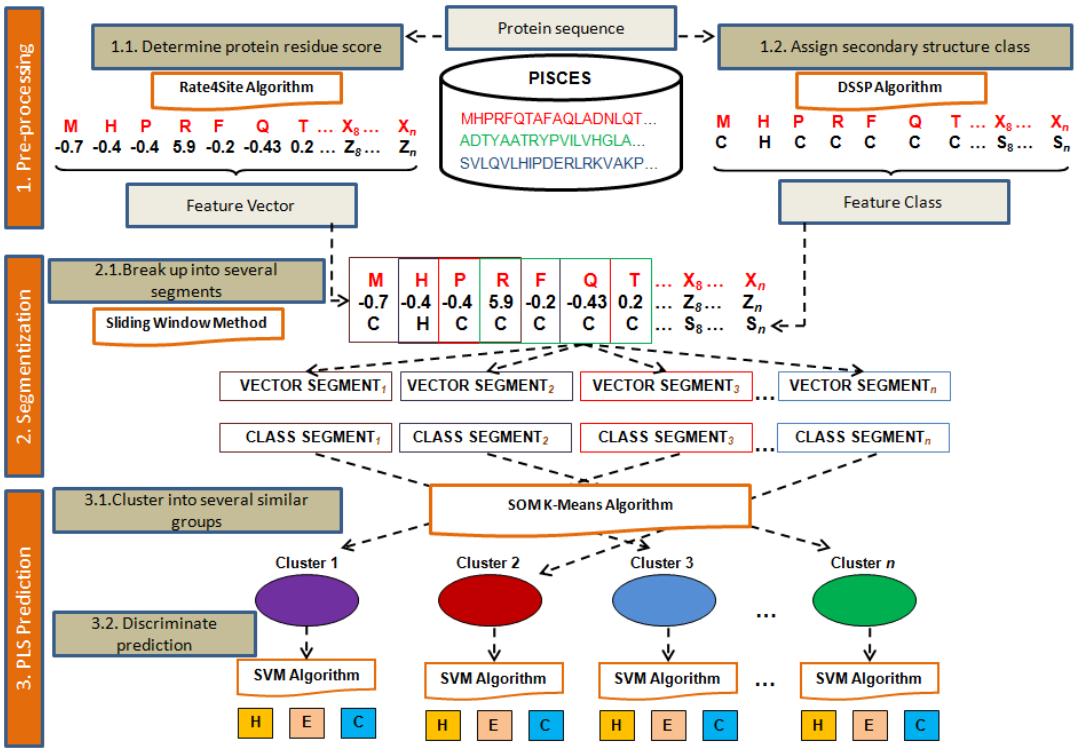


Figure 1. The proposed computational framework of R-HCSVM.

In fact, they adopted profile score from HSSP [32] database that only emphasized more on functional information to represent the protein residue score. On the other hand, Chen et al. [4] has proposed HYPLOSP to predict the local structure that based on Neural Network algorithm. Yet, they introduced high-scoring segment pairs for protein residue score that conserved more homology information rather than secondary structure information. Therefore, this study proposes a new algorithm to predict local structure named, R-HCSVM as shown in Figure 1. This R-HCSVM consists of two major components to (1) increase the strength of protein residue score and (2) reduce the training complexity. The R-HCSVM begins by determining the protein residue score using the first component named as ResiduePatchScore information. This ResiduePatchScore information ables to increase the strength information of protein residue score by combining protein conservation score that conserved rich functional information and protein propensity score that highly conserved secondary structure information. Subsequently, each of the protein sequence will be sliced into window segment to become feature vectors using sliding window method which has been implemented in Zhong et al. [42]. Next, DSSP method which was proposed by Kabsch and Sander [14] is used to assign secondary structure class to each protein residues. Besides, this study proposes granular SVM classification named as HCSVM in order to reduce the training complexity of predictive algorithm. Due to the large amount of protein subsequences being generated, Self-Organizing Map (SOM) is hybridized with K-Means to produce the granular input intelligently for SVM. This granular input allows SVM classification done in a series of tractable and simpler computationally problems. The detail explanation of the R-HCSVM can be found in the next section. Meanwhile, Receiver Operating Curve (ROC) and Segment Overlap (SOV) accuracy are used as metrics to evaluate the performance of the R-HCSVM in comparison to other similar algorithms. Experimental results show that R-HCSVM significantly improves the performance of protein local structure prediction.

The remainder of the paper consists of the detailed explanation of R-HCSVM (Section 2), description of the computational environment and data used in this study (Section 3), the results and discussion of experiments (Section 4), and the conclusions (Section 5).

3. METHODOLOGY

In this study, the proposed protein local structure prediction algorithm works as follows: (i) pre-process protein residue score information, (ii) pre-process protein secondary structure information, (iii) segmenting protein residue, (iv) classify protein subsequence for each granular input using SVM and (v) evaluate R-HCSVM using ROC and SOV.

3.1 Materials and Implementation

The dataset used in this study includes 2,000 protein sequences obtained from the PISCES [38] database. This dataset is the training dataset which is used to model the R-HCSVM. This protein database is bigger and more advanced than PDB-select-25 [11] that was used by Han and Baker [10]. Since PISCES uses PSI-BLAST [1] alignments to distinguish many underlying patterns below 40% identity, PISCES produces a more rigorous non-homologous database than PDB-select-25. In PISCES, the local alignment will not incorporate two proteins that share a common domain with sequence identity above the given threshold. This feature helps to overcome problems of PDBREPRDB [25] database which uses global alignment that may generate useless sequence similarities. Meanwhile, to avoid the bias testing dataset, the k-fold cross validation is implemented. In this study, kf=10 is applied. Besides, one of the vectors used in this study is extracted from protein residue conservation score in Consurf server database which is available at http://consurfdb.tau.ac.il. Each of protein residue conservation score in alignment is calculated using Rate4Site algorithm. The advantages of this score as a result of implementation of phylogenetic relations between the aligned proteins and the stochastic nature of the evolutionary process explicitly. In addition, Rate4Site algorithm [30] assigns a conservation level for each position in MSA using an empirical Bayesian Inference. Whereby, the clustering process has been executed for six times to obtain the stable output clusters.

3.2 Pre-process Protein Residue Score Information

As mentioned earlier, there are two types of protein residue score. One is determined by the propensity score based on the frequency occurrence of protein secondary structure. This score is outstanding in predicting protein secondary structure as a result of high structural conserved secondary structure information. To date, protein residue score is mostly determined based on its evolutionary history which is more functional conserved and known as protein conservation score. Besides, the advantage of this score is based on the superior Rate4Site algorithm that implements explicitly the phylogenetic relations between the aligned proteins and the stochastic nature of the evolutionary process through multiple sequence alignment in order to inherit highly conserved functional information and able to cater sequence redundancy**.** Therefore, this study is inspired to couple both protein residue score information named as ResiduePatchScore information in order to increase the strength of structural and functional conserved information. Further, the inaccurate prediction as consequence of bias protein residue score can be avoided. Four scores are employed to each protein residue. One is obtained from Consurf server database which is developed by Goldenberg et al. [9]. Meanwhile, the rest three scores are calculated based on its secondary structure propensity ratio in the whole dataset using Eq. 1. These secondary structure propensity scores clarify the degree of predominant role of H, E and C for each residue. Therefore, they were adopted in order to increase the strength of specified secondary structure information for each residue.

$$P_{ab} = \frac{(n_{ab} / n_a)}{(N_b / N_T)}, \quad (1)$$

here, n_{ab} is the number of residues of type a in structure of type b, n_a is the total number of residues of type a, N_b is the total number of residues in structure of type b and N_T is the total number of residues in the whole dataset.

3.3 Pre-process Protein Secondary Structure Class

There are several approaches of secondary structure assignment available such as DSSP [14], DEFINE [8] and STRIDE [31]. DSSP is selected in this study as it is the most widely used secondary structure definition program in recent studies. Basically, DSSP is able to recognize eight types of secondary structure depending on the pattern of hydrogen bonds that are H

(α-helix), G (3_{10}-helix), I (π-helix), E (β-strand), B (isolated β-bridge), T (turn), S (bend) and the rest. However, in this study DSSP assigns each of residues using three larger classes of secondary structure namely H for helices, E for sheets and C for coils. The encoding secondary structure class is based on the following assignment: (i) H, G and I to H, (ii) E to E and (iii) the rest states to C.

3.4 Segmenting Protein Residue

Sliding window method is used to generate protein subsequence from 2,000 protein sequences. Each of protein subsequence composes of nine continuous residues. Therefore, it will generate up to 50,000 protein subsequences. In addition, many local structure prediction method use protein subsequence rather than the whole sequence itself during the prediction process. According to Chen et al. [5], the formation of helical structure can be affected by residues that are up to 9 positions away in the sequence, while the formation of coils and strands can be affected by residues that are up to 3 and 6 positions away respectively. The shorter formation structure in protein subsequence can yield noticeably improved the clustering process. Thus, this study generates the protein local segments with length of 9 residues to be known as protein local structure.

3.5 Clustering and Discriminate Protein Local Structure

It is simpler and tractable to utilize SVM in multiple granular input spaces. Therefore, HCSVM contains two parts and works by: (1) group protein subsequences into several clusters using SOM K-Means and (2) classify protein subsequences in each clusters using SVM to identify the secondary structure class. The SOM is implemented first as a rough phase to reveal the similarity amongst protein subsequences. A vector quantization method in SOM able to simplify and reduces the training complexity in a SOM component plane as well as to discover the intrinsic relationship amongst protein subsequences. Next, K-Means is implemented as a refining phase on the learnt SOM to reduce the problem size of SOM cluster to the optimal number of *K*.

The SVM classifier is subsequently used to train the protein subsequences in each cluster. Assume that a training protein subsequence *S* is given as;

$$S = \{x_i, y_i\}, i = 1...n, \quad (2)$$

where each x_i is a feature vector and $y_i \in \{-1,+1\}$ corresponds to x_i label or feature class. The goal of SVM is to find the optimal hyperplane,

$$w.\phi(x_i) + b = 0, \quad (3)$$

in a high-dimensional space that able to separate the data from classes -1 and $+1$ with maximal margin. w is a weight vector orthogonal to the hyperplane, b is a scalar and ϕ is a function which maps the data into a high-dimensional space also named as feature space. One merit of SVM is to map the input vectors into a high dimensional feature space and thus can solve the nonlinear case. The capability of SVM in handling the nonlinear relationship amongst protein subsequence is based on the nonlinear kernel function. The RBF is used as the nonlinear kernel function and defined as follows:

$$K(x_i, x_j) = \exp\left(\frac{-r \| x_i - x_j \|^2}{2\sigma^2}\right), \quad (4)$$

where x_i and x_j are input vectors. The input vector will be the centre of the RBF and σ will determine the area of influence this input vector has over the data space. A larger value of σ will give a smoother decision surface and more regular decision boundary since the RBF with large σ will allow an input vector to have a strong influence over a larger area.

3.6 Evaluate Prediction

There are three secondary structure classes H, E or C will be determined or predicted for given protein subsequence. Meanwhile, the predictive algorithm in this study is based on binary classification which is presented in two classes for each secondary structure class. For example, to predict the protein subsequence as H class, positive class, +1 will be assigned to protein subsequence which is detected as H. Conversely, negative class, −1 will be assigned to protein subsequence which is detected as non H. Four possible outcomes will be generated from this classifier. The classification of these outcomes is described in contingency table 2x2 in Table 1.

Table 1. Contingency table 2x2 for binary classifier outcomes.

	Actual H	Actual non H
Predicted H	*True positives (TP)*	*False negatives (FN)*
Predicted non H	*False positives (FP)*	*True negatives (TN)*

Further, Table 2 explains the definition of variables that used in Table 1. Later, these variables derived to the used in ROCl formula.

Table 2. The definition of variable used in contingency table 2x2 for binary classifier outcomes.

Variable	Meaning
TP	Number of occurrence when both actual and predicted is positive class.
FN	Number of occurrence when actual is positive class and predicted is negative class.
FP	Number of occurrence when actual is negative class and predicted is positive class.
TN	Number of occurrence when both actual and predicted is negative class.

Basically ROC curve is used to visualize the performance of binary classifier in cartesian graph. Area under curve as shown in the following formula is another statistical index to describe the ROC measurement.

$$\text{ROC} = \frac{1}{2}(TPR)(FPR), \quad (7)$$

where TPR defines the proportion of correct predicted positive instances among all positive protein subsequence are being tested. FPR defines the proportion of incorrect positive results occur among all negative protein subsequence is being tested. To provide an indication of the overall performance of the predictive algorithm, we computed SOV. For example, the definition of the SOV measure for H is as follows:

$$\mathrm{SOV_H} = \frac{1}{N_H}\sum_{i=1}^{N_H}\frac{\min\{OV(s_i,s_{i+1})\}+\delta(s_i,s_{i+1})}{\max\{OV(s_i,s_{i+1})\}}, \quad (8)$$

here, s_i and s_{i+1} are the observed and predicted secondary structure of local segments in the H state. N_H is the total number of protein local segments in H conformation. $\min\{OV(s_i, s_{i+1})\}$ refers to the minimum length of the actual overlap of s_i and s_{i+1} and $\max\{OV(s_i, s_{i+1})\}$ is the maximum length of the total extent for which either of the segments s_i or s_{i+1} has a residue in H state. Furthermore, the definition of $\delta(s_i, s_{i+1})$ is as follows quoted by Zemla et al. [40]:

$$\delta(s_i,s_{i+1}) = \min\left\{\begin{array}{c}\max\{OV(s_i,s_{i+1})\}-\min\{OV(s_i,s_{i+1})\}\\ \min\{OV(s_i,s_{i+1})\}\\ \mathrm{int}(0.5(len(s_i)))\\ \mathrm{int}(0.5(len(s_{i+1})))\end{array}\right\}, \quad (9)$$

where, $\mathrm{len}(s_i)$ is the number of amino acid residues in segment. The similar calculation of SOV score in Eq. 8 will be applied to E and C state too.

4. RESULTS AND DISCUSSIONS

In this study, we test R-HCSVM and compare its performance with other methods such as SVM-light which is done by Joachims [13] that involves classifier alone, KCSVM which is introduced by Zhong et al. [42] that hybrid K-Means clustering algorithm and SVM classifier, R-KCSVM is a KCSVM with incorporates enriched protein residue score and HCSVM is a hybrid SOM K-Means clustering algorithm and SVM classifier without incorporates enriched protein residue score. Firstly, feature vector and feature class of R-HCSVM prediction method are pre-processed. Feature vector is represented using protein residue score which has been enriched by coupling the residue conservation score and residue propensity score based on secondary structure conserved information. On the other hand, feature class is represented by three states of secondary structure class which are generated using DSSP algorithm. Subsequently, all these feature vectors and classes are sliced in a window segment in prior to be discriminate using hybrid clustering SVM. Finally, the results generated by hybrid clustering SVM are evaluated. This evaluation provides a clear understanding of strengths and weaknesses of an algorithm that has been designed.

The datasets of protein sequences obtained from PISCES database that have been defined in the previous section are used to test and evaluate the R-HCSVM and other protein local structure prediction methods. As depicted in Table 3 and emphasizes in Figures 2─3, using classifier alone which is represented by SVM-light produces the lowest accuracy per segment of 60.2% and average ROC of 44%. This is due to the high complexity of dataset inherits influence noise. In contrary, prediction method which implemented clustering algorithm at first hand shows better performance accuracy. Hybrid clustering SVM shows tremendous improvement of prediction method by revealing the sequence-to-local structure relationship in a smaller and tractable dataset. This is proved by KCSVM that increase 10% higher in ROC and 5.3% higher in accuracy per segment compared to prediction using SVM alone. Furthermore, sequence-to-local structure relationship is revealed in two levels learning process in HCSVM, where the first level is using SOM K-Means clustering algorithm and the second level is continued using SVM classifier. As a result, the sequence-to-local structure relationship process is more focused and the ROC as well as SOV is much higher with 17% and 8.6% respectively compared to prediction using SVM alone. In addition, by enriching the information of protein residue score did improve the prediction method. This is due to the enriched protein residue score employed both high functionally and structurally conserved information which led to the increment of fraction score between the observed and predicted protein local segments. In R-KCSVM, the average ROC and SOV increased up to 16% and 11.38% respectively compared to prediction using KCSVM. Meanwhile, in R-HCSVM, the average ROC and SOV increased up to 17% and 10.86% respectively compared to prediction using HCSVM.

5. CONCLUSIONS AND FURTHER WORKS

This paper discussed a computational method which is developed one is to increase the strength of protein residue score information and another one is to solve the training complexity of prediction algorithm in order to boost up the performance accuracy of protein local structure prediction. In the proposed computational method, there are two major machine learning algorithms are employed. One is SOM K-Means which is used to break up the complex dataset of protein local structures into several granular inputs or subspaces. Further, SVM classifier is implemented to each of generated granular inputs to learn and predict the protein local structure. In order to increase the strength of input information to this prediction algorithm, the protein residue score has been introduced which integrates protein conservation score and protein propensity score based on secondary structure information. The results from the evaluation phase in previous section shown that hybrid clustering SVM did improve the performance accuracy significantly compared to prediction algorithm that using classifier alone. Meanwhile, hybrid clustering SVM with incorporated enriched protein residue score is much improved the performance accuracy rather than using hybrid clustering SVM only.

Table 3. Performance comparison between R-HCSVM with other protein local structure prediction methods.

Method	ROC	SOV (%)
R-HCSVM	0.78	79.76
R-KCSVM	0.70	76.88
HCSVM	0.61	68.90
KCSVM	0.54	65.50
SVM-Light	0.44	60.20

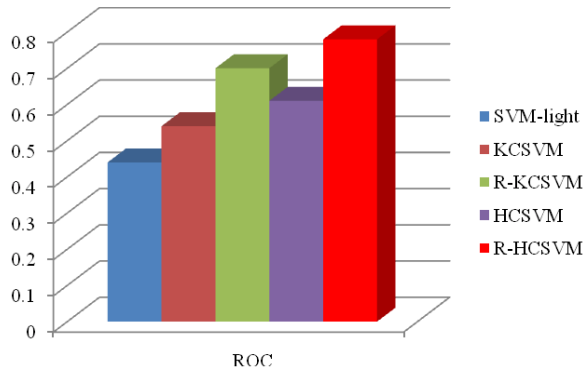


Figure 2. Performance comparison between R-HCSVM with other protein local structure prediction methods on ROC.

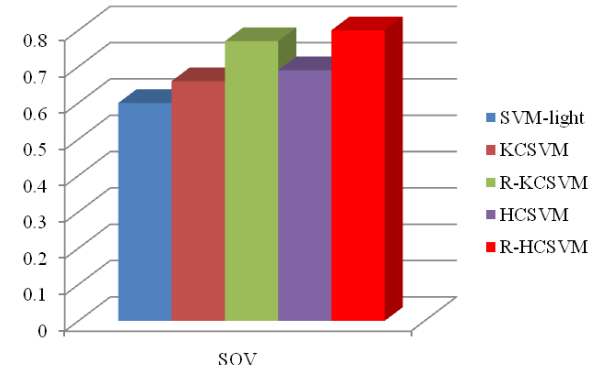


Figure 3. Performance comparison between R-HCSVM with other protein local structure prediction methods on SOV.

However, the performance accuracy specifically for sheets has a room to be improved. This study found that helices are the hardest to be captured in protein subsequence. One attempt to solve the problem is to enrich the secondary structure class information in order to capture more sheets occurrence. Besides, as a consequence of using binary classifier to predict three states of secondary structure class, unbalanced predicted class is occurred. Therefore, in future work, learning based secondary structure assignment will be proposed in order to capture more variability of secondary structure class and tertiary coding scheme will be integrated in order to solve the unbalanced predicted class.

6. ACKNOWLEDGMENTS

This work is funded by the Malaysian Ministry of Science, Technology and Innovation (MOSTI) under grant no. 01-01-06-SF0436. The authors sincerely thank reviewers for their comments on an earlier version of this manuscript.

7. REFERENCES

[1] Altschul, S. F., Madden, T. L., Schaffer, A. A., Zhang, J., Zhang, Z., Miller, W. and Lipman, D. J. (1997). Gapped BLAST and PSI–BLAST: A New Generation of Protein Database Search Programs. *Nucleic Acids Research*. 25(17): 3389–3402.

[2] Balcazar, J. L., Dai, Y. and Watanabe, O. (2001). Provably Fast Training Algorithms for Support Vector Machines. *Proceedings of the IEEE International Conference on Data mining*. November 29 – December 2, 2001. California, USA: IEEE Computer Society Press. 43–50.

[3] Bystroff, C., Thorsson, V. and Baker, D. (2000). HMMSTR: A Hidden Markov Model for Local Sequence–Structure Correlations in Proteins. *Journal of Molecular Biology*. 301(1): 173–190.

[4] Chen, C. T., Lin, H. N., Sung, T. Y. and Hsu W. L. (2006). Hyplosp: A Knowledge–Based Approach to Protein Local Structure Prediction. *Journal of Bioinformatics and Computational Biology*. 4(6): 1287–1308.

[5] Chen, K., Kurgan, L. and Ruan, J. (2006). Optimization of the Sliding Window Size for Protein Structure Prediction. *Proceedings of the IEEE Symposium on Computational Intelligence and Bioinformatics and Computational Biology*. September 28-29, 2006. Ontario, Canada: Blackwell Publishing. 1–7.

[6] Chou, P. Y. and Fasman, G. D. (1978). Prediction of the Secondary Structure of Proteins from their Amino Acid Sequence. *Journal of Advances in Enzymology and Related Areas of Molecular Biology*. 47(1): 145–148.

[7] Constantini, S., Colonna, G. and Facchiano, A. M. (2007). PreSSAPro: A Software for the Prediction of Secondary Structure by Amino Acid Properties. *Computational Biology and Chemistry*. 31(5-6): 389–392.

[8] Frishman, D. and Argos, P. (1995). Knowledge–Based Protein Secondary Structure Assignment. *Proteins*. 23(4): 566–579.

[9] Goldenberg, O., Erez, E., Nimrod, G. and Ben-Tal, N. (2009). The ConSurf-DB: Pre-Calculated Evolutionary Conservation Profiles of Protein Structures. *Nucleic Acids Research*. 37(Database Issue): D323-D327.

[10] Han, K. F. and Baker, D. (1996). Global Properties of the Mapping Between Local Amino Acid Sequence and Local Structure in Proteins. *PNAS*. 93(12): 5814–5818.

[11] Hobohm, U. and Sander, C. (1994). Enlarged Representative Set of Protein Structures. *Protein Science*. 3(3): 522–524.

[12] Hu, C. Q. and Hu, Y. Z. (2008). Small Molecule Inhibitors of the p53–MDM2. *Current Medical Chemistry*. 15(17): 1720–1730.

[13] Joachims, T. (1999). Making Large-Scale SVM Learning Practical. In: Scholkopf, B. and Burges, C. and Smola, A., Eds. *Advances in Kernel Methods: Support Vector Learning*. Cambridge, USA: MIT Press. 169–184.

[14] Kabsch, W. and Sander, C. (1983). Dictionary of Protein Secondary Structure: Pattern Recognition of Hydrogen–Bonded and Geometrical Features. *Biopolymers*. 22(12): 2577–2637.

[15] Karchin, R., Cline, M. and Karplus, K. (2004). Evaluation of Local Structure Alphabets Based on Residue Burial. *Proteins*. 55(3): 508–518.

[16] Karchin, R., Cline, M., Mandel–Gutfreund, Y. and Karplus, K. (2003). Hidden Markov Models that use Predicted Local

Structure for Fold Recognition: Alphabets of Backbone Geometry. *Proteins*. 51(4): 504–514.

[17] Karlin, S. and Brocchieri, L. (1996). Evolutionary Conservation of RecA Genes in Relation to Protein Structure and Function. *Journal of Bacteriology*. 178(7): 1881–1894.

[18] Karypis, G. (2006). YASSPP: Better Kernels and Coding Schemes Lead to Improvements in Protein Secondary Structure Prediction. *Proteins*. 64(3): 575-586.

[19] Kim, T. M., Chung, Y. J., Rhyu, M. G. and Jung, M. H. (2007). Inferring Biological Functions and Associated Transcriptional Regulators using Gene Set Expression Coherence Analysis. *BMC Bioinformatics*. 8(1): 453–465.

[20] Levitt, M. and Chotia, C. (1976). Structural Patterns in Globular Proteins. *Nature*. 261(5561): 552–558.

[21] Lin, K., Simossis, V. A., Taylor, W. R. and Heringa, J. (2005). A Simple and Fast Secondary Structure Prediction Method using Hidden Neural Networks. *Bioinformatics*. 12(12): 1041–1050.

[22] Liu, W. M. and Chou, K. C. (1999). Prediction of Secondary Structure Content. *Protein Engineering*. 21(2): 152–159.

[23] Lu, Y., Lu, S., Fatouhi, F., Deng, Y. and Brown, S. J. (2004). Incremental Genetic K-Means Algorithm and its Application in Gene Expression Data Analysis. *BMC Bioinformatics*. 5(1): 172–182.

[24] Mirny, L.A. and Shakhnovich EI. Universally conserved positions in protein folds: reading evolutionary signals about stability, folding kinetics and function. Journal of Molecular Biology 1999;291:177–196.

[25] Noguchi, T., Matsuda, H. and Akiyama, Y. (2001). PDB-REPRDB: A Database of Representative Protein Chains from the Protein Data Bank (PDB). *Nucleic Acids Research*. 29(1):219–220.

[26] Osuna, E., Freund, R. and Girosi, F. (1997). An Improved Training Algorithm for Support Vector Machines. *Proceedings of the IEEE Workshop on Neural Networks for Signal Processing*. September 24-26, 1997. Amelia Island Florida, USA: IEEE Press. 276–285.

[27] Pauling, L. and Corey, R. B. (1951). Atomic Coordinates and Structure Factors for Two Helical Configurations of Polypeptide Chains. *PNAS*. 37(5): 235–240.

[28] Pauling, L. and Corey, R. B. (1951). The Pleated Sheet, a New Layer Configuration of Polypeptide Chains. *PNAS*. 37(5): 251–256.

[29] Platt, J. (1999). Fast training of support vector machines using sequential minimal optimization. In: *Advances in Kernel Methods-Support Vector Learning*. Cambridge, USA: MIT Press. 185-208.

[30] Pupko, T., Bell, R. E., Mayrose, J., Glaser, F. and Ben, T. N. (2002). Rate4site: An Algorithmic Tool for Identification of Functional Regions in Proteins by Surface Mapping of Evolutionary Determinants within their Homologues. *Bioinformatics*. 18(Suppl 1): 71–77.

[31] Richards, F. M. and Kundrot, C. E. (1988). Identification of Structural Motifs from Protein Coordinate Data: Secondary Structure and First–Level Super Secondary Structure. *Proteins*. 3(2): 71–84.

[32] Sander, C. and Schneider, R. (1991). Database of Homology-Derived Protein Structures and the Structural Meaning of Sequence Alignment. *Proteins*. 9(1): 56–68.

[33] Sander, O., Sommer, I. and Lengauer, T. (2006). Local Structure Prediction using Discriminative Models. *BMC Bioinformatics*. 7(14): 1–13.

[34] Schoelkopf, B., Tsuda, K. and Vert, J. P. (2004). *Kernel Methods in Computational Biology*. MIT Press. 71-92.

[35] Shamir, R. and Sharan, R. (2002). Algorithmic Approaches to Clustering Gene Expression Data. In: Jiang, T., Smith, T., Xu, Y. and Zhang, M., Eds. *Current Topics in Computational Biology*. Cambridge, USA: MIT Press. 269–300.

[36] Shenkin, P. S., Erman, B. and Mastrandrea, L. D. (1991). Information-Theoretical Entropy as a Measure of Sequence Variability. *Proteins*. 11(4): 297–313.

[37] Valentini, G. and Dietterich,T. G. (2003). Low Bias Bagged Support vector Machines. *Proceedings of the International Conference on Machine Learning*. August 21-24, 2007. Washington DC, USA: AAAI Press. 752–759.

[38] Wang, G. and Dunbrack, R. L. (2003). PISCES: A Protein Sequence–Culling Server. *Bioinformatics*.19(12): 1589–1591.

[39] Wu, T. T. and Kabat, E. A. (1970). An Analysis of the Sequences of the Variable Regions of Bence Jones Proteins and Myeloma Light Chains and their Implications for Antibody Complementarity. *Journal of Experimental Medicine*. 132(2): 211–249.

[40] Zemla, A., Venclovas, C., Fidelis, K. and Rost, B. (1999). A Modified Definition of SOV, A Segment-Based Measure for Protein Secondary Structure Assessment. *Proteins*. 34(2): 220–223.

[41] Zhong, W., Jieyue, H. and Yi, P. (2007). Multiclass Fuzzy Clustering Support Vector Machines for Protein Local Structure Prediction. *Proceedings of the IEEE International Conference on Bioinformatics and Bioengineering*. April 1-5, 2007. Hawaii, USA: IDEA Group Publishing. 21–26.

[42] Zhong, W., Jieyue, H., Robert, W. H., Phang, C. T. and Yi, P. (2007). Clustering Support Vector Machines for Protein Local Structure Prediction. *Expert System with Application*. 32(2): 518–526.

Incorporating Gene Ontology with Conditional-based Clustering to Analyze Gene Expression Data

Shahreen Kasim
Intelligence and Bioinformatics Laboratory, Faculty of Computer Science and Information Systems, Universiti Teknologi Malaysia, 81310 UTM Skudai, Malaysia.
shahreen.kasim@gmail.com

Safaai Deris
Intelligence and Bioinformatics Laboratory, Faculty of Computer Science and Information Systems, Universiti Teknologi Malaysia, 81310 UTM Skudai, Malaysia.
safaai@utm.my

Razib M. Othman
Laboratory of Computational Intelligence and Biology, Industry Centre, Technovation Park, Universiti Teknologi Malaysia, 81310 UTM Skudai, Malaysia.
razib@utm.my

ABSTRACT
One of the purposes of the analysis of gene expression data is to cater for the cancer classification and prognosis. Currently, clustering has been introduced as a computational method to assist the analysis. However, these clustering algorithms focus only on statistical similarity and visualization presentation, thus neglecting the biological similarity and the consistency of the annotation in the cluster. Furthermore, there are still complexity issues and difficulty in finding optimal cluster. In this study, we proposed a clustering algorithm named BTreeBicluster to overcome those problems. The BTreeBicluster starts with the development of GO tree and enriching it with expression similarity from the Sacchromyces genes. From the enriched GO tree, the BTreeBicluster algorithm is applied during the clustering process. The BTreeBicluster takes subset of conditions of gene expression dataset using discretized data. Therefore, the annotation in the GO tree is already determined before the clustering process starts which gives major reflect to the output clusters. The results of this study have shown that the BTreeBicluster produces better consistency of the annotation.

Keywords
Biclustering, Discretization, Expression and biological similarity, Gene expression analysis, Gene ontology.

1. INTRODUCTION
Gene expression data has been widely used in the bioinformatics analysis. The analysis of gene expression profile is used to predict cancer classification for example Sotiriou et al. [1] have done the research for breast cancer classification and prognosis. Antonov et al. [2] have also done the classification concentrating on tumor samples based on microarray data. This procedure detects groups of genes and constructs models (features) that strongly correlate with particular tumor types. Meanwhile, Xiong and Chen [3] used the optimized kernel to increase the performances of the classifiers in classifying gene expression data.

Apart from classification, clustering is also a useful data-mining tool for discovering similar patterns in gene expression dataset, which may lead to the insight of significant connections in gene regulatory networks. Cheng and Church [4] introduced the concept of bicluster which captures the similarity of clustering of both genes and conditions. Meanwhile, Getz et al. [5] introduced Coupled Two-Way Clustering (CTWC) analysis on colon cancer and leukemia datasets. Lazzerroni and Owen [6] introduced plaid models which is similar as cluster analysis. These plaid models incorporate additive two way ANOVA models within the two-sided clusters of yeast gene expression datasets. However, all of these works only focus more on mathematical similarity of genes and conditions. These works did not pay attention to the biological process of each cluster. Lately, several biclustering methods have been introduced. The advantage of using biclustering is the genes in one cluster do not have to behave similarly through all conditions. Bicluster referred to subset of genes that behave similarly in a subset of conditions. Some of related works in bicluster is Samba [7]. Samba presented a graph-theoretic approach to biclustering in combination with a statistical data model. Iterative Signature Algorithm (ISA) [8] considers a bicluster to be a transcription module, for instance a set of co-regulated genes together with the associated set of regulating conditions. Meanwhile, in Order Preserving Submatrix Algorithm (OPSM) [9], a bicluster is defined as a submatrix that preserves the order of the selected columns for all of the selected rows. In the algorithm *xMotif* by Murali and Kasif [10], biclusters are sought for which the included genes are nearly constantly expressed - across the selection of samples. All of these methods are too complex to be solved which their optimization problems are NP-hard and did not bring optimal cluster result.

Fang et al. [11] developed biclustering method which incorporates Gene Ontology (GO) [12] in the expression data. GO has been applied in many works, for example Liu et al. [13] had incorporated GO information in its Smart Hierarchical Tendency Preserving clustering (SHTP-clustering). Hvidsten et al. [14] induced predictive rule models for functional classification of gene expressions which are also taken from the GO. Moreover, there are softwares based on GO for performing statistical determination, interpretation and visualization of function profiles such as GOMiner [15], GOTree Machine [16], Onto-Tools [17], GO::TermFinder [18] and FunSpec [19]. However, all of these software uses the knowledge in GO only to evaluate the clustering results rather than to improve the clustering itself.

Therefore, in order to solve complexity problems and to achieve optimal cluster, we developed a new clustering method named BTreeBicluster, which applies fundamental biclustering method and at the same time integrating GO in the analysis of gene expression data. Our method differs from the conventional clustering techniques such as hierarchical clustering (HCL) [20], as it allows genes in the same cluster not to respond similarly across all experimental conditions. Instead, it is defined as a subset of genes that shows similar expression patterns over a subset of conditions. This is useful to find processes that are

active in some but not all samples. The BTreeBicluster also uses discretized data which will bring a comprehensive result. Furthermore, the BTreeBicluster eschew random interference caused by masked bicluster in Cheng and Church [4]. More importantly, the BTreeBicluster is based on similarity measures which expression profiles and biological functions are taken before clustering. This step is a major difference from other clustering methods which does the annotation after the clustering process.

The detailed explanation of our method is in the following sections. In Section 2, the clustering algorithm is illustrated. In Sections 3, the results of the BTreeBicluster on two realworld datasets [20], [21] and some comparison with other published methods are presented. Finally, in Section 4, the BTreeBicluster is discussed, and thus some conclusions are drawn. The paper ends with perspectives for other potential applications and suggestions for further improvements.

2. METHODS

The GO is applied in the construction of hierarchical tree before mapping the GO tree with the gene expression data. Based on this GO tree, it is traversed from one node to another node, from top to bottom using level-by-level traversal method. In the GO tree, unmapped nodes are excluded. Gene is mapped to a node and its descendant nodes form an initial matrix cluster. Their expression similarity is then calculated, and the matrix cluster with high similarity will be the output that will be excluded in the next calculation. If the high expression similarity is not obtained, no action will be taken. The process is repeated until the whole GO tree has been visited. The BTreeBicluster produces a set of clusters using improved biclustering method. These clusters are enriched with expression and functional similarities. Then, these clusters are evaluated to check its reliability and the consistency of annotation. In the following sections, we further illustrate our method in detail. The BTreeBicluster is shown in Figure 1.

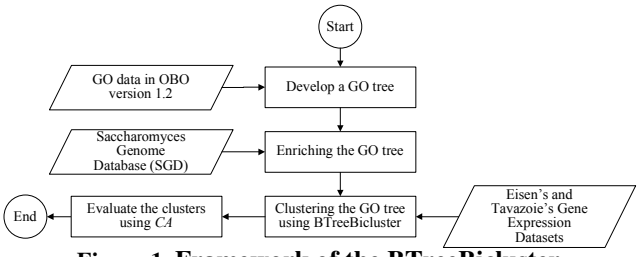


Figure 1. **Framework of the BTreeBicluster**

2.1 Develop the GO Tree

The GO tree is constructed using GO OBO version 1.2. Currently in the GO website (http://www.geneontology. org), there are nearly 25,231 terms which refers to the controlled vocabulary used to describe gene and gene product attributes in any organism. These terms are classified as only one of the three ontologies: cellular component, biological process or molecular function. Each term in these ontologies is structured as a Directed Acyclic Graph (DAG). There are many types of GO data formats such as OBO-XML, RDF-XML, OBO version 1.2, MySQL, OWL and flat file. In this paper, we have chosen GO OBO version 1.2 as our GO data. The purpose of choosing this data is due to the neatly arrangement of the terms thus easy to read.

We parsed the GO OBO version 1.2 format by reading the file line by line and then compare the string values to extract each GO term and its attributes. Then, each term and its relationship information such as "is-a" and "part-of" are put as a node into the linked list. A complete linked list is built when the process of reading and adding each term from the GO OBO version 1.2 file is finished.

Starting from the first node in the complete linked list, a root node of a tree is created and removed from the linked list. Using the concept of binary tree, each node in our tree has two pointers which are left pointer and right pointer. The left pointer of a tree node points to its first child while the right pointer points to its next sibling. After the root node has been created, the process continues with the next available node in the linked list. By using pre-order traversal method, the GO tree is traversed recursively where each node in a tree is compared with the node from the linked list. A node is said to be a child of another node when it has "is-a" or "part-of" relationship. By using information in the node from the linked list, if a node in the tree is found to have a parent-child relationship with it, the node is then added to the tree. Then, the node is removed from the linked list. The process of GO tree construction continues until there are no more nodes in the linked list to be processed. A complete GO tree is now constructed and the example is illustrated by Figure 2.

2.2 Enriching the GO Tree

In mapping genes with the GO tree, we used Saccharomyces Genome Database (SGD) [22]. The GO terms obtained from the SGD will be mapped to the developed GO tree structure. Beginning with the root node of the GO tree, each node in the GO tree is visited using level-order traversal method. Every time a new node is visited, the SGD format file is looked into for all matches of the GO terms. If a match is found, its matching gene information is then saved in the respective GO tree node. The example of mapped GO tree with SGD genes is shown in Figure 3.

2.3 Clustering the GO Tree Using BTreeBicluster

The BTreeBicluster has interesting advantages compared to other clustering methods. The BTreeBicluster did not take the whole set of conditions in clustering process and did not apply the average trend constraint in the clustering process as been used in Fang et al. [11]. The clustering over the whole set of conditions may separate the biologically related genes from each other. Therefore, by applying the BTreeBicluster, the running time and memory complexity of the whole clustering process can be reduced. Apart from that, Prelic et al. [23] demonstrated that using discretization will improved the clustering result. Therefore, in this method, we apply discretization to our datasets. The discretization process will change the data into binary values where each cell in the matrix is set to value 1 whenever gene *i* show reaction in the condition *j*.

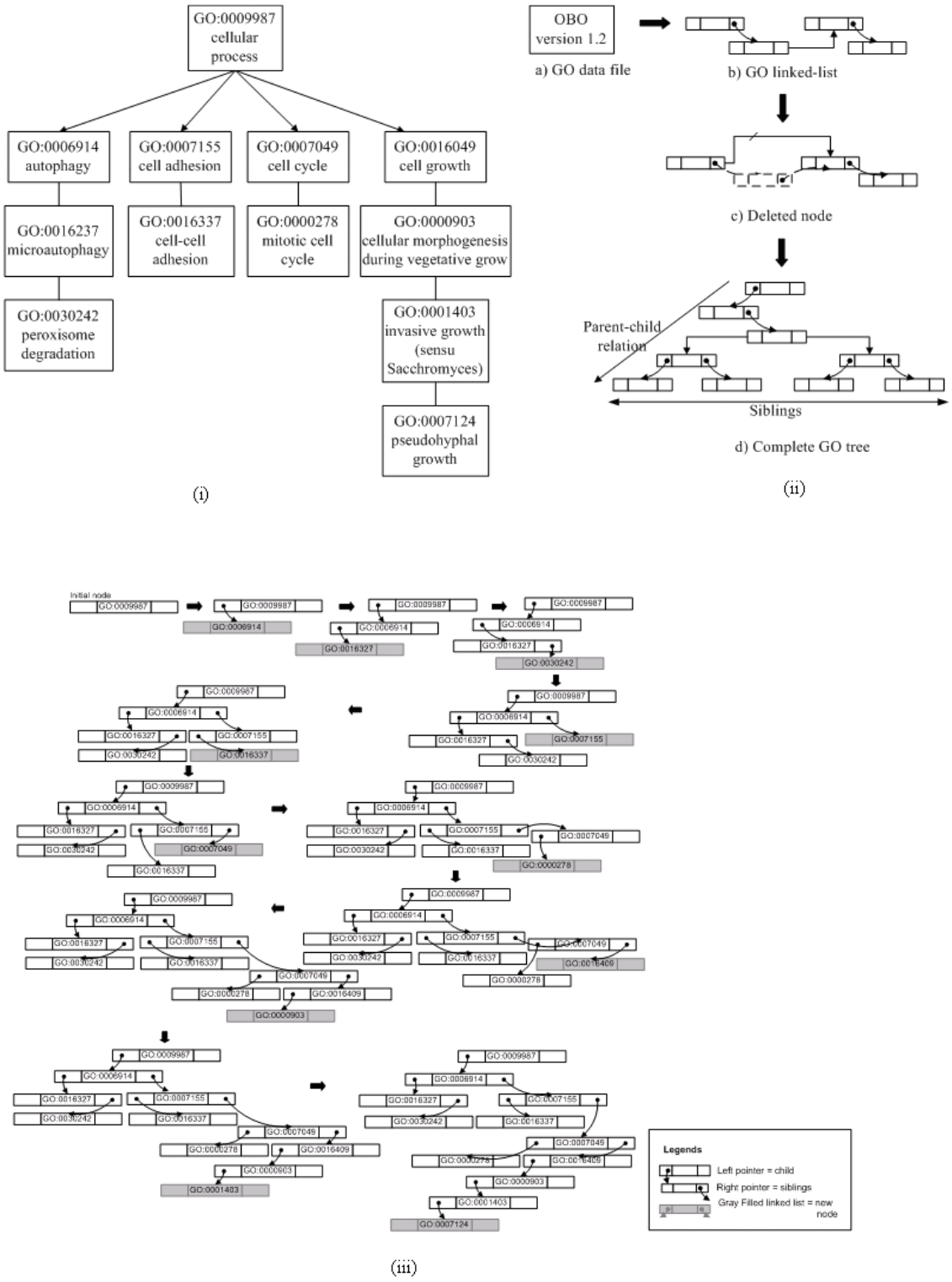


Figure 2. (i) The example of GO tree, (ii) The process of construction of the GO tree using linked list, and (iii) The constructed GO tree using linked list.

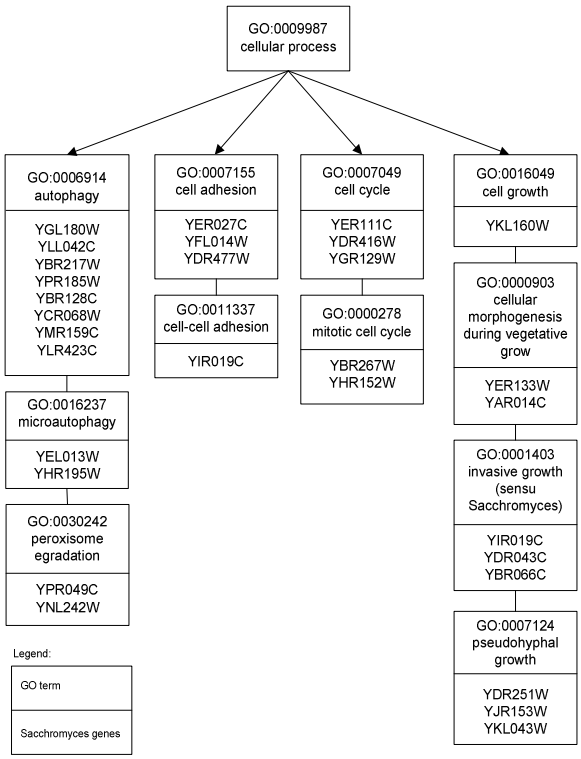


Figure 3. The example of enriched GO tree with Sacchromyces genes.

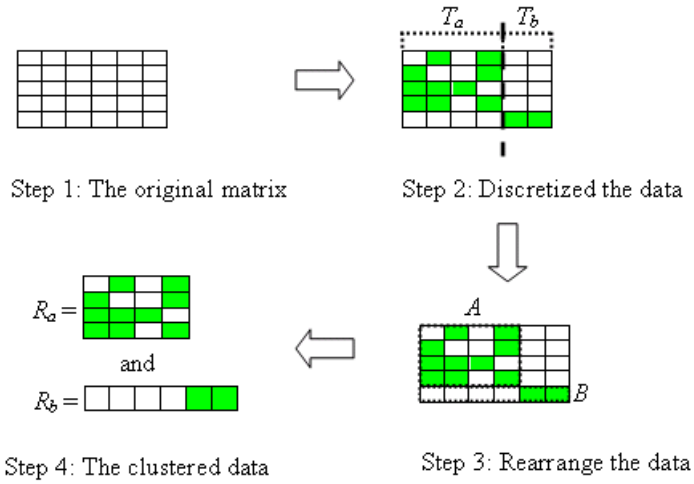


Figure 4. Steps involved in the BTreeBicluster.

1	**Algorithm :** BTreeBicluster
2	**Input :** GT : GO tree
3	EP : Expression profiles
4	IM : Input matrix
5	YD: Yeast data
6	**Output:** R=(R_1,R_2,R_3,…,R_n): gene biclusters
7	
8	***begin***
9	***for*** i=1 to max level of GT ***do***
10	***for*** j=1 to max node index of level i ***do***
11	***if*** the node j in GT is clustered, ***then*** continue.
12	***else***
13	Find all genes in YD corresponding to node j on GT.
14	Find all descendant nodes of j on GT with their mapped gene set P.
15	
16	Select the expression profile of genes in P from EP and build an input matrix
17	
18	$S(P,Q)$.
19	***if*** all cells in IM contains only 1's, ***then***
20	bicluster of genes is output
21	***else***
22	Divide S to T_a, T_b
23	***while*** a bicluster is not yet found ***do***
24	Divide S to T_a, T_b
25	***if*** all cells in S contains only 1's, ***then***
26	bicluster of genes is output.
27	***end-if***
28	***end-while***
29	***end-if***
30	***end-if***
31	***end-for***
32	***end-for***
33	***end***

Figure 5. Algorithm of the BTreeBicluster.

The process of BTreeBicluster starts as follows. Beginning with GO tree most top (root) node down to the bottom node(s), we use level-by-level tree traversal method to visit each node in the GO tree. For every level we traverse, we start from the most left node of the level. During a visit to a node, we check for nodes mapped with genes. Nodes which are not mapped with genes will not be considered in our further process. We proceed to the next node if a particular node is already marked with 'clustered'. Otherwise, all its descendant nodes will be selected.

Given the set of genes which are mapped to the node and its descendant nodes is P, we define the gene expression profiles of P as a matrix, $S(P,Q)$. Thereafter, subsequent process is all based on the input matrix S. In the clustering process we used a fast divide and conquer. The algorithm of BTreeBicluster rearranges the data taken from previous step into two subsets. This is in order to divide the input matrix into two smaller, possibly overlapping sub-matrices A and B. First, the set of columns is divided into two subset columns T_a and T_b, by taking the first row as a guide template. Then, the rows of S are rearranged. All genes that respond to conditions in T_a are arranged first. Then, it arranges

those genes that respond to conditions in T_a and T_b. Finally the arrangement of genes that respond to conditions T_b is taken place. The corresponding sets of genes R_a and R_b are then defined in combination with T_a and T_b resulting sub-matrices A and B which are decomposed recursively. R_a will take place if there is overlapping between A and B. The recursion ends if a bicluster matrix is found that is contains only 1s. The two matrices, A and B can be processed independently from each other if they do not share any rows and columns of input matrix S. Otherwise, a very specific process is necessary to get those biclusters in B that has at least share one common column with those in T_b. The steps involved in the algorithm are shown in Figure 4 and the algorithm is shown in Figure 5.

3. RESULTS AND DISCUSSION

The experiments were implemented using Java on Intel Core Duo CPU T2450 computer with 1GB RAM and 2GHz processor. The GO data format used in this experiment is OBO version 1.2: revision 5.483. The SGD revision 1.1381 used in the experiment was downloaded from http://www.yeastgenome.org. The gene expression profiles are taken from two popular datasets [21], [22].

The purpose of this study is to get the optimum cluster result using BTreeBicluster. Therefore, to check the consistency of the annotation in the output cluster, we evaluate it by the definition given below:

$$CA = 1 - \frac{m}{n}, \quad (1)$$

where *CA* is the consistency for the cluster. For every cluster *R*, *m* refers to number of genes annotated by certain term while *n* refers to total number of genes in *R*. By using this evaluation, the smaller of *CA* produces better consistency. The value of zero *CA* is where all the genes in any each of the clusters holding the same annotation. Therefore, the higher number of zero *CA* is resulted to high consistency. For the average of *CA*, the total of every calculated *CA* for each cluster is divided by total number of clusters. Thus, the smaller value of average *CA* confirmed the cluster results to the high consistency.

As shown in Table 1, we compared the consistency values between BTreeBicluster method with Eisen's and Fang's methods using Eisen's dataset. For this comparison, there are 2467 genes and 79 conditions in the dataset. We tested this data by setting the threshold = 2.0. The BTreeBicluster has shown 3367 clusters which gave the best result of *CA* where it has 2424 of the zero *CA* and 0.0071 of the average *CA*. Fang's method showed that there are 423 clusters with 258 of zero *CA* and 0.1839 of the average *CA*. Meanwhile, Eisen's method showed there are 9 clusters with 1 of zero *CA* and 0.3508 of the average *CA*. Based on this comparison, BTreeBicluster proved that the consistency value of annotation in our clusters is better than Fang's and Eisen's methods. This is due to the fast divide and conquer approach in BTreeBicluster which has been done to the mapped GO tree.

We also evaluated our BTreeBicluster with Tavazoie's dataset by setting the threshold = 2.0 as shown in Table 2. By using this dataset, there are 6601 genes and 17 conditions. The BTreeBicluster has shown 564 clusters which gave the best result of *CA* where it has 424 of the zero *CA* and 0.0420 of the average *CA*. Fang's method showed that there are 513 clusters with 394 of zero *CA* and 0.0905 of the average *CA*. Meanwhile, Tavazoie's showed there are 30 clusters with 0 of zero *CA* and 0.2799 of the average *CA*. Based on this comparison, BTreeBicluster proved that the consistency value of annotation in our clusters is better than Fang's and Eisen's methods, due to the divide and conquer approach in BTreeBicluster that the mapped GO tree confirmed the similarity in clusters for both expression and biological.

Table 1. Comparison of *CA* values using Eisen's dataset

Clusters	No. of cluster	No. of zero *CA*	Average *CA*
Eisen's	9	1	0.3508
Fang's	423	258	0.1839
BTreeBicluster	3367	2424	0.0071

Table 2. Comparison of CA values using Tavazoie's dataset

Clusters	No. of cluster	No. of zero *CA*	Average *CA*
Tavazoie's	30	0	0.2799
Fang's	513	394	0.0905
BTreeBicluster	564	424	0.0420

Based on the result as shown in Table 3, the bigger the number of threshold is, the smaller number of clusters can be obtained. On the other hand, through this experiment, by using a smaller threshold, the more number of clusters is obtained. The best threshold value is 2.0 where the number of clusters produced is 126 for Eisen's dataset and 9 for Tavazoie's dataset. The stability and CPU performance of the BTreeBicluster can be seen in Table 4 where results of 5 separate runs are compared by taking threshold = 2.0 for each run. The results show that in Eisen's dataset,the number of clusters is 126 and the average of CPU performance is 11,532 seconds. Meanwhile, in Tavazoie's dataset, the number of clusters is 9 and the average of CPU performance is 5,700 seconds.

Table 3. Comparison of clusters produced with different setting of thresholds for Eisen's and Tavazoie's datasets

Threshold	No. of cluster (Eisen's dataset)	No. of cluster (Tavazoie's dataset)
2.0	126	9
2.2	87	2
2.4	67	0
2.5	56	0
2.6	50	0
2.8	49	0
3.0	38	0

Table 4. Comparison of CPU running time and number of cluster produced for Eisen's and Tavazoie's datasets.

Run	Eisen's dataset		Tavazoie's dataset	
	CPU time (seconds)	No. of cluster	CPU time (seconds)	No. of cluster
Run1	11,460	126	5,400	9
Run 2	11,700	126	6,060	9
Run 3	11,400	126	5,460	9
Run 4	11,520	126	5,700	9
Run 5	11,580	126	5,880	9

4. CONCLUSIONS

The aim of this work is to get optimal cluster for better consistency of annotation in the GO tree. This study has shown that clustering the gene expression can be done by developing the GO tree. Then, the process is continued by enriching the GO tree with the SGD genes. Furthermore, the BTreeBicluster is used to cluster the GO nodes from the GO tree that has relationship with the genes. The experiments showed that BTreeBicluster outperformed the existing methods by producing clusters with expression and biological similarity.

Unlike any other methods, the BTreeBicluster can prove the annotation before the clustering starts. This process can determine the expression and biological similarity in the first phase before the clustering process starts. This process also can avoid genes being clustered with dissimilar functions. Furthermore, the BtreeBicluster used discretization data. The main advantage of using discretized data is that human beings naturally are more easy dealing with discrete data rather than in continuous quantities and also the discrete data is generally better received by classifiers in the classification process [24]. In addition, the BTreeBicluster only take subsets of the dataset where the areas that contain 0s are excluded.

Although the experiments have shown that BTreeBicluster produces good results, future research in the quality of the GO itself should be done. It is assumed that a more updated GO version will improve the GO tree thus improve the clustering results. The processing time can be also be reduced by using high performance computing and parallel algorithms for future research.

5. ACKNOWLEDGMENTS

This project is funded by the Malaysian Ministry of Science, Technology, and Innovation (MOSTI) under ScienceFund grant no. 02-01-06-SF0068. The authors sincerely thank reviewers for their comments on an earlier version of this manuscript.

6. REFERENCES

[1] Sotiriou C., Neo S.Y., McShane L.M., Korn E.L., Long P.M., Jazaeri A., Martiat P., Fox S.B., Harris A.L., Liu E.T. Breast cancer classification and prognosis based on gene expression profiles from a population-based study. In: Proceedings of the National Academy of Sciences of the United States of America; 2003. p. 10393-8.

[2] Antonov A.V., Tetko I.V., Mader M.T., Budczies J., Mewes H.W. 2004. Optimization models for cancer classification: extracting gene interaction information from microarray expression data. Bioinformatics. 20:644-52.

[3] Xiong H, Chen X.W. Optimized kernel machines for cancer classification using gene expression data. In: Proceedings of the IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology; 2005. p. 1-7.

[4] Cheng Y, Church G.M. Biclustering of expression data. In: Proceedings of the International Conference on Intelligent Systems for Molecular Biology; 2000. p. 93-103.

[5] Getz G, Levine E, Domany E. Coupled two-way clustering analysis of gene microarray data. In: Proceedings of the National Academy of Sciences of the United States of America; 2000. p. 12079-84.

[6] Lazzeroni L., Owen A. 2002. Plaid models for gene expression data. Statistica Sinica;12:61-86.

[7] Tanay A., Sharan R., Shamir R. 2002. Discovering statistically significant biclusters in gene expression data. Bioinformatics;18:S136-44.

[8] Ihmels J., Bergmann S., Barkai N. 2004. Defining transcription modules using large-scale gene expression data. Bioinformatics;20:1993-2003.

[9] Ben-Dor A., Chor B., Karp R, Yakhini Z. 2003. Discovering local structure in gene expression data: the order-preserving sub-matrix problem. Journal of Computational Biology;10:49-57.

[10] Murali T.M and Kasif S. Extracting conserved gene expression motifs from gene expression data. In: Proceedings of the Pacific Symposium on Biocomputing; 2003. p. 77-88.

[11] Fang Z., Li Y., Luo Q, Liu L. 2006. Knowledge guided analysis of microarray data. Journal of Biomedical Informatics;39:401-11.

[12] The Gene Ontology Consortium. 2000. Gene ontology: tool for the unification of biology. Nature Genetics;25:25-29.

[13] Liu J., Wang W., Yang J. Gene ontology friendly biclustering of expression profiles. In: Proceedings of the IEEE Computational Systems Bioinformatics Conference; 2004. p. 436-7.

[14] Hvidsten T.R., Komorowski J., Sandvik A.K., Laegreid A. Predicting gene function from gene expressions and ontologies. In: Proceedings of the Pacific Symposium on Biocomputing World Scientific; 2001. p. 299-310.

[15] Zeeberg B.R., Feng W., Wang G., Wang M.D., Fojo A.T., Sunshine M, et al. 2003. GoMiner: a resource for biological interpretation of genomic and proteomic data. Genome Biology;4:R28.

[16] Bing Z., Denise S., Stefan K., Jay S. 2004. GOTree Machine (GOTM): a web-based platform for interpreting sets of interesting genes using Gene Ontology hierarchies. BMC Bioinformatics;5:16.

[17] Sorin D., Purvesh K., Pratik B., Abhik S., Stephen A.K., Michael A.T. 2003. Onto-Tools, the toolkit of the modern biologist: onto-express, ontocompare, onto-design and onto-translate. Nucleic Acids Research;31:3775-81.

[18] Boyle E.I., Weng S., Gollub J., Jin H., Botsterin D., Cherry J.M., Sherlock G. 2004. GO::TermFinder-open source software for accessing gene ontology information and finding significantly enriched gene ontology terms associated with a list of genes. Bioinformatics;20:3710-15.

[19] Robinson M.D., Grigull J., Mohammad N., Hughes T.R. 2002. FunSpec: a web-based cluster interpreter for yeast. BMC Bioinformatics;3:35.

[20] Eisen M.B., Spellman P.T., Brown P.O., Botstein D. Cluster analysis and display of genome-wide expression patterns. In: Proceedings of the National Academy of Sciences of the United States of America; 1998. p. 14863-8.

[21] Tavazoie S., Hughes D., Campbell M.J., Cho R.J., Church G.M. 1999. Systematic determination of genetic network architecture. Nature Genetics;22:281-5.

[22] Cherry J.M., Adler C., Ball C., Chervitz S.A., Dwight S.S., Hester E.T., Jia Y, et al. 1998. SGD: saccharomyces genome database. Nucleic Acids Research;26:73-80.

[23] Prelic A., Bleuler S., Zimmermann P., Wille A., Buhlmann P., Gruissem W., Hennig L., et al. 2006. A systematic comparison and evaluation of biclustering methods for gene expression data. Bioinformatics;22:1122-9.

[24] Waldron M., Penaloza M. Genetic algorithms as a data discretization method. In: Proceedings of the Midwest Instruction and Computing Symposium; 2005. <http://www.micsymposium. org/ mics_2005/ papers/paper16.pdf >

IMPROVING PROTEIN-PROTEIN INTERACTION PREDICTION BY A FALSE POSITIVE FILTRATION PROCESS

Rosfuzah Roslan
Laboratory of Computational
Intelligence and Biology
Faculty of Computer Science and Information Systems
Universiti Teknologi Malaysia, 81310 UTM Skudai, MALAYSIA
+607-5599230
eternal_exquisite@yahoo.com

Razib M. Othman
Laboratory of Computational
Intelligence and Biology
Faculty of Computer Science and Information Systems
Universiti Teknologi Malaysia, 81310 UTM Skudai, MALAYSIA
razib@utm.my

ABSTRACT

Protein-protein interactions (PPI) play a significant role in many crucial cellular operations such as metabolism, signaling and regulations. The computational prediction methods for PPI have shown tremendous growth in recent years, but problem such as huge false positive rates has contributed to the lack of solid PPI information. We aimed to enhance the overlap between computational predictions and experimental results with the effort to partially remove the false positive pairs from the computational predicted PPI datasets. The usage of protein function prediction based on shared interacting domain patterns named PFP() for the purpose of aiding the Gene Ontology Annotation (GOA) is introduced in this study. We used GOA and PFP() as agents in the filtration process to reduce the false positive in computationally predicted PPI pairs. The functions predicted by PFP() which are in Gene Ontology (GO) IDs that were extracted from cross-species PPI data were used to assign novel functional annotations for the uncharacterized proteins and also as additional functions for those that are already characterized by GO. As we know, GOA is an ongoing process and protein normally executes a variety of functions in different processes, so with the implementation of PFP(), we have increased the chances of finding matching function annotation for the first rule in the filtration process as much as 20%. The results after the filtration process showed that huge sums of false positive pairs were removed from the predicted datasets. We used signal-to-noise ratio as a measure of improvement made by applying the proposed filtration process. While strength values were used to evaluate the applicability of the whole proposed computational framework to all the different computational PPI prediction methods.

Keywords

False Positive Filtration, Gene Ontology, Interaction Rules, Protein-Protein Interaction Predictions, Shared Interacting Domain Patterns.

1. INTRODUCTION

PPI play critical roles in the control of most cellular processes and act as a key role in biology since they mediate the assembly of macromolecular complexes, or the sequential transfer of information along signaling pathways. Many proteins involved in signal transduction, gene regulation, cell-cell contact and cell cycle control require interaction with other proteins or cofactors to activate those processes [1–4]. In recent years, high throughput technologies have provided experimental methods to identify PPI in large scale, generating tremendous amount of PPI data such as yeast two hybrid (Y2H) and mass spectrometry of coimmunoprecipitated complexes (Co-IP) [5]. Several methods have been previously used to identify true interactions in high-throughput experimental data like paralogous verification methods [6] structurally known interactions [7] and by using an interaction generality measure [8]. Advances in experimental methods are paralleled by rapid development of computational methods designed to detect vast number of protein pairs on wide genome scale. The major limitation in both the computational and experimental approaches is their lack of confidence in the identification of PPI, with high false positive and false negative rates [5], [9]. Most efforts in computational approaches focused on predicting more PPI by the means of various approaches that identify true positives. The results from these approaches are higher or of huge volume of predicted PPI datasets that contains not only more true positive predictions but also numerous false positive predictions.

Experimental PPI detection methods attempt to discover direct physical interactions between proteins while computational PPI prediction often refer to functional interactions [10]. Efforts and researches in enhancing true positive fraction of computationally predicted PPI datasets has not been adequately investigated. A lot of other researchers have focused on improving computational method in producing better result of predicted datasets in terms of its accuracy which means low false positive by means of refinement of a particular computational method [11–13] or an integration of several types of computational methods such as joint observation method (JOM) [14], [15] that calculates the accuracy and coverage for the PPI that were predicted by at least one, two, three or four methods using three positive datasets (KEGG, EcoCyc and DIP). Those methods are Phylogenetic Profiles (PP), Gene Cluster (GC), Gene Fusion (GF) and Gene Neighbourhood (GN). STRING [16] that integrate combined scores for each pair of proteins and InPrePPI [17] that integrates

the scores of each protein pair obtained by the four methods. While other researchers focused on improvement in computational methods area, Mahdavi and Lin [18] have proposed a filtering algorithm solely using GOA [19]. The removal of false positive depends on whether the predicted pairs satisfy the heuristic rules that were developed based on the concept of PPI in cellular systems observation. The result after the filtration process differs among different types of computational methods that were implemented. GOAs that were used as a common ground for the filtration process has been a popular and reliable source for several research which concern validation or evaluation of a certain results such as in Patil and Nakamura [20]. GOAs were used as one of the means to assign reliability to the PPI in yeast determined by high-throughput experiments. GO has appeared to be utilized in several studies concerning PPI. GO [21] terms had been used by Rhodes et al. [22] to assess associations between proteins in a pair while Wu et al. [23] constructed a PPI network for yeast by measuring the similarity between two GO terms with a relative specificity semantic relation. In the meantime, Hsing et al. [24] used GO term for predicting highly-connected 'hub' nodes in PPI networks. While Dyer et al. [25] used GO to provide functional data to protein interactome sets that also revealed interactions of human proteins with viral pathogens. From the GO analysis, it indicated that many different pathogens target the same processes in the human cell, such as regulation of apoptosis, even if they interact with different proteins. On the other hand, GO structural hierarchy was used to evaluate functional associations by Lord et al. [26].

Although GO shows tremendous usage in recent studies, GO suffers from inconsistency within and between genomes. This is because ontology annotation is an ongoing process, thus it is considered incomplete and does not contain full or complete annotations. Problems that could arise from this limitation are, one protein is assigned a term that represents a broad type of activity, and its interacting partner is assigned a more specific term. There are some cases where some proteins have not even been assigned all three ontologies which make the interaction assessments more difficult. There is also a possibility that a substantial portion of most genomes are still unannotated such as *D. melanogaster* and *H. sapiens* and some proteins are still uncharacterized. Chen et al. [27] has stated that only about 54% among the current list of *D. melanogaster* genes that were downloaded from FlyBase [28] as on November 2006 are annotated with molecular function terms in GO.

In this paper, we aimed to enhance the overlap between computational predictions and experimental results through a confidence level which reflects the agreement of a link between both the experimental results and computational predictions. Therefore, we proposed a computational framework to filter false positive of the predicted PPI pairs so that it will increase true positive fraction of the computationally predicted PPI dataset. Using GO as a common ground in the filtering process, we also implemented protein function assignment based on the shared interacting domain patterns extracted from cross-species PPI data to assign novel functional annotations for the uncharacterized proteins and predict extra functions for proteins that are already annotated in the GO. The involved species in PPI data that were used to infer the uncharacterized or incomplete functions are *S. cerevisiae*, *C. elegans*, *D. melanogaster* and *H. sapiens.* In order to evaluate the improvement made by the proposed filtration process, the Signal-to-Noise Ratio [29] was employed while value strength [18] was calculated to show the effect of the rules applied.

A series of steps was conducted in a framework to refine the computationally predicted datasets. First, a set of *S. cerevisiae* PPI datasets with high confidence were prepared for the experimental dataset and one set of each newly updated PPI dataset consist of four species (*C. elegans*, *D. melanogaster*, *H. sapiens* and *S. cerevisiae*). Second, GOAs with the aid of GO functions predicted by the shared interacting domain patterns extracted from cross-species PPI data were utilized to identify keywords which represent general functions of the proteins. Third step was to establish interaction rules. It is established to be satisfied by the predicted interacting proteins. Next, four computational PPI prediction methods were selected to use in this study. Those methods are the conventional Phylogenetic Profiles (PP) [30], Gene co-Expression (GE) [31], Mutual Information (MI) [32] and Maximum Likelihood Estimation (MLE) [33]. For each of these computational methods, predicted PPI datasets were obtained. Then, the false positive pairs that exist in the predicted datasets were removed by applying the interaction rules. If the predicted interacting pair satisfies the rules, then it is considered as a true positive pair, otherwise the pair is assume as a false positive pair and removed from the dataset. The result of the filtered datasets were statistically evaluated and compared.

2. METHODS

In this study, a computational framework for the refinement of computationally predicted datasets is proposed. Basically the predicted protein pairs are filtered according to the matching keywords that represent general molecular functions and the matching cellular component of both proteins. The ‘keywords’ are the top ranking keywords resulting from the keywords identification process based on GO molecular function of the interacting proteins in experimental datasets. The assignment of GO molecular functions to the associated interacting proteins directly from cross-reference assignment of GO and InterPro [34] are being aid with protein function prediction based on cross-species (*C. elegans*, *D. melanogaster*, *H. sapiens* and *S. cerevisiae*) interacting domain patterns. The justification for both of this rules and the concept of the protein function prediction based on the interacting domain patterns will be explain further in their respective sub-sections. The results of the filtered datasets and raw datasets (unfiltered datasets) are compared in order to evaluate the significant effect of the proposed algorithm. The proposed computational framework is as shown in Figure 1.

2.1 Protein Function Prediction

We applied a procedure named PFP() that predicts the interacting proteins based on the concept that interaction between protein pairs in diverse species having the shared domain patterns. It produced assignments of appropriate GO functional annotations to proteins by finding modular domains that are likely to possess similar functions. The underlying hypothesis for this procedure is that similarity in functions for the proteins exists when proteins in the two PPI pairs share similar modular domains in which the PPI

pairs contain a common interacting domain pattern. Figure 2 demonstrates the function annotation scheme based on this hypothesis. The figure shows two PPI pairs, the first one is protein A that interacts with protein B and the second pair is protein C that interacts with protein D. Proteins A and C contain the same modular domain X that interact with the modular domain Y in proteins B and D. Therefore, it is concluded that the two PPI pairs share a common interaction domain pattern in which proteins A and C share similar functions whereas proteins B and D.

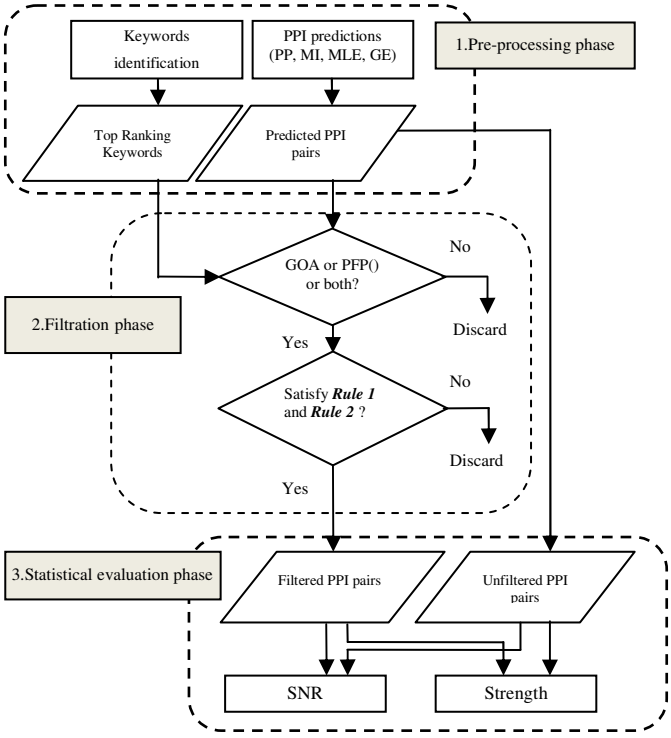


Figure 1. Proposed computational framework for false positive reduction and true positive sustainment.

This procedure has to be trained in order to produce a lookup table of significant interacting modular domain patterns from the interaction pairs that contain domain patterns and associated functional assignments. It finds groups of protein interaction pairs across different organisms with similar functions. During the training phase, groups of protein pairs with similar functions were formed. Each PPI pair in the training dataset serves as a centroid to the formation of these groups. The remaining pairs are compared against the centroid interaction pair. Then χ^2 statistics has been applied to derive interacting domain patterns from the PPI group. A list of function terms that stem from the PPI pairs involved with the creation of the domain patterns are then enlisted to the lookup table along with the corresponding domain patterns. The equation for χ^2 is as follows:

$$\chi^2 = \frac{N(AD-CB)^2}{(A+C)(B+D)(A+B)(C+D)}. \quad (1)$$

N as indicated in equation above represents the total number of PPI pairs in the reference dataset. Variable *A* refers to the number of PPI pairs in the group that contain the particular 'pattern', while *B* is the number of remaining PPI pairs outside the group that contain the 'pattern'. Variable *C* indicates the number of PPI pairs that do not contain the 'pattern' in the group. While variable *D* is the number of PPI pairs that do not contain the 'pattern' in the remaining samples outside the group. The domain patterns that will be adopted in the lookup table for function annotation are the deduced interacting domain patterns with the highest χ^2 value.

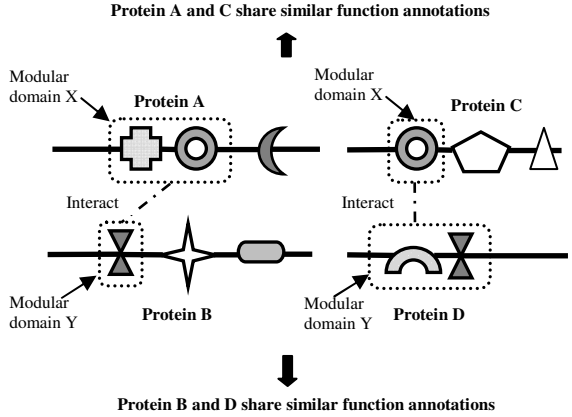


Figure 2. PFP() underlying hypothesis in predicting function annotations.

2.2 Keywords Identification

This step is where we define or identify associated keywords that represent the entire interacting proteins in our experimental dataset. First, protein pairs in the experimentally verified dataset were submitted to UniProt [35]. Then we retrieved GO and InterPro cross-reference assignments of the proteins. All InterPro entries were mapped to GO terms using "*interpro2go*" dated 2 July 2008, that were retrieved from GO website. The GO terms of each protein were then searched using AMIGO term search engine [36]. After collecting the searched GO term information of each protein, the redundant information was removed. The GO terms information on molecular function annotation were compiled and used as a training dataset. The GO molecular function for the interacting proteins in the experimental dataset was acquired based on function prediction resulted from cross-species shared interacting domain patterns. The results retrieved are in the form of GO IDs and later the GO terms were extracted from the flatfile of "*interpro2go*". After collecting all of the GO term information, it was then added to the training dataset. Redundant information in the training dataset was removed. We produced the final dataset into several groups that were clustered according to their general molecular activities.

The number of occurrences (*n*) of a word in a cluster was counted in order to determine representative keyword in a cluster. The calculation using Poisson distribution was conducted to find the probability of finding that word in the training dataset. The formula for Poisson distribution is as follows,

$$p(n) = e^{-\lambda}\left(\frac{\lambda^n}{n!}\right), \quad (2)$$

λ in the equation above is the result of *N.f*, in which *N* refer to the total number of words in a cluster, while *f* refer to the relative frequency of that word in the whole training dataset. In order to avoid floating point errors, we implemented *n*! based on Stirling's approximation, resulting in

$$\ln p(n) = -\lambda + n(\ln n) - n(\ln n) + n - 1. \quad (3)$$

In order for this calculation to be valid, the total number of words in the training dataset has to be much greater than *N* or when *f* is small. All enzyme activities were considered as "*ase activity*" since biochemistry literature introduced enzymes with "*ase*" suffix. The representative keyword in each cluster is the word with the most negative ln *p* (*n*) value.

2.3 Interaction Rules

Interaction rules that were established here are based on PPI in cellular systems, based on the observation that two proteins are more likely to interact to perform the same function and that proteins are required to exist in close proximity to interact physically [37]. These rules were applied in the filtration process where the predicted interacting protein pairs have to satisfy both rules,

***Rule 1*:** Both of predicted proteins in the pair should match one of the trained 'top ranking keywords' that represent a function or functions that the pair carries.

***Rule 2*:** Both of predicted proteins in the pair should be in the same GO cellular components.

2.4 PPI Predictions

Four PPI prediction methods from different categories that were used in this study as shown in Table 1 will be briefly explained. In the PP method, PP of all proteins in the experimental dataset was gathered from PLEX database [38]. Once all PP were constructed, we grouped the proteins that shared similar profile, then we paired them with each other within the group and considered the pairs interacting. The next method is MI which utilizes MI function as a measure of similarity between two PP. After profile for each protein is constructed, we used MI value to assess the confidence level of the link between the two proteins of each protein pair. The candidate interactions are identified by setting the value of threshold of mutual information (TMI). When the MI value between two proteins is higher than the TMI, we regarded it as interacting.

In the implementation of GE method, SMD [39] was used to obtained normalized expression levels of *S. cerevisae* that corresponds to a different microarray experiment (100 experiments). CLICK algorithm in EXPANDER program [40] was used for clustering the matrix supplied. Genes that are in the same cluster are the co-expressed genes and thus considered interacting with each other. Meanwhile, for the MLE method we followed the underlying hypothesis which is two proteins are considered interacting if and only if at least one pair of domains from the two proteins interact based on the understanding that in order to perform the necessary functions, protein domains physically interact with one another. All datasets resulted from these prediction methods will be used as the testing datasets.

Table 1. Computational PPI prediction methods that were selected and their respective categories.

Method	Category
Conventional Phylogenetic Profiles (PP)	Utilization of genomics information to predict protein interactions.
Mutual Information (MI)	Rely on statistical scoring functions with the purpose to enrich conventional genomics methods.
Maximum Likelihood Estimation (MLE)	Domain-based approach.
Gene co-Expression (GE)	Prediction through integration of microarray data in different biological conditions.

2.5 Filtration of False Positive

After obtaining the predicted PPI datasets for each computational method, we executed the filtration process for the purpose of reducing the rates of false positive as many computational PPI prediction methods suffer from mass false positive predictions. By satisfying the interaction rules, the predicted false positive pairs are discarded or removed from the predicted datasets resulted to a dataset that contain higher true positive fraction compared to before it was filtered. Based on the filtration phase illustrated in Figure 1, predicted PPI pairs from each of the four predicted PPI datasets were examined sequentially by the algorithm to determine if the proteins in the particular pair possess the molecular function annotations from GO or from the shared interacting domain patterns or both. It also examined the GO cellular component annotations. If both annotations are present, such pair is checked with the proposed rules. This protein pair is required to satisfy Rule 1 and Rule 2 to be considered as an interacting pair. The final output that contains predicted interacting protein pairs, are called filtered predicted PPI dataset. Then, we assess the level of agreement of the predicted PPI with the experimentally obtained dataset by comparing them both.

2.6 Statistical Evaluation

The purpose for this statistical evaluation is to measure the significant effect or improvement made by applying the filtering process to the predicted PPI datasets. We employed SNR that measure signal strength relative to background noise. SNR in bioinformatics is translated to the ratio of capability of a computational method in creating protein pairs to pairing proteins on a random basis. For this statistical evaluation, we define SNR as follows,

$$SNR = \frac{(matched\_pairs / total\_pairs)_{predicted\_dataset}}{(matched\_pairs / total\_pairs)_{random\_dataset}}. \quad (4)$$

The random dataset has to be randomly selected with the same sample size. Matched pairs in the equation above means the matched protein pairs with the experimental dataset and the total pairs is the total number of pairs in the same dataset. It is also defined as the true positive fraction of the dataset. SNR was calculated according to two circumstances, the raw dataset and the filtered dataset. Raw dataset means the predicted PPI pairs before applying the rules. While the filtered dataset means the predicted PPI pairs after applying the rules. After calculating SNR of raw and filtered datasets for all four PPI predicted datasets, we find strength, ratio of SNR filtered dataset to SNR raw dataset in order to measure the effect of the application of the false positive filtration rules. The equation for strength is as follows,

$$S = \frac{SNR_{filtered\_dataset}}{SNR_{raw\_dataset}} . \quad (5)$$

3. MATERIALS

3.1 Experimental Dataset

We used the experimentally obtained protein pairs from CYGD [41] database to extract the functional keywords from the GO annotations. The CYGD is a frequently used public resource for yeast related information that was generated by the European consortium and serves as a reference for the exploration of fungi and higher eukaryotes. 15453 experimentally verified *S. cerevisiae* protein pairs consisting of 4748 interacting proteins were used in this study.

3.2 PPI Datasets for Function Prediction

For the protein function prediction based on cross-species shared interacting domain patterns, two datasets were involved. The first dataset is called training PPI dataset. This dataset was collected from the DIP [42], BioGRID [43] and MINT [44] databases for the organisms *S. cerevisiae*, *C. elegans* and *D. melanogaster*, while for organism *H. sapiens,* the interaction data were obtained from HPRD [45] database. The final training PPI dataset which does not contain any uncharacterized proteins consist of 11151, 231, 7709 and 13596 interaction pairs from *S. cerevisiae*, *C. elegans*, *D. melanogaster* and *H. sapiens* respectively. Protein domain information were extracted from PFAM [46] database. Pfam-A and Pfam-B domains were considered for each protein. Total of 493 unique Pfam domains were found to be in common between the four species. There are a total of 2972 unique GO annotated molecular function terms obtained from GO database within the dataset.

The second dataset is called interaction information dataset. This collection of PPI is used to enhance the probability of finding a pattern in the lookup table generated from the process of the training PPI dataset. We used a newly updated PPI dataset retrieved in March 2008 that consist interaction pairs of four well-studied eukaryotic species. The dataset consist of 77006 of *S. cerevisiae,* 6853 of *C. elegans* and 25300 of *D. melanogaster* that were acquired from the BioGRID and the DIP databases, while 43527 of *H. sapiens* interaction data was acquired from the HPRD database.

3.3 GO and Annotations

GOA provides high-quality electronic and manual annotations to the UniProt Knowledgebase that consist of UniProtKB/Swiss-Prot and UniProtKB/TrEMBL using the GO standardized vocabulary. Annotations in both GO and UniProt databases are updated on a regular basis. We used UniProt Knowledgebase release 14.0 (July 2008) and the GO database August 2008 release.

4. RESULTS AND DISCUSSIONS

4.1 Predicted Protein Function

Using the interaction information dataset for the purpose of enhancing the probability of finding a pattern in the lookup table of PFP(), we managed to identify functional predictions for 1393 proteins. After analyzing the result of the PFP(), we discovered that PFP() compliments the function annotations for our interacting proteins in which it increases the number of function annotations received. Table 2 shows some of the additional functional terms predicted by PFP() that have enriched the function annotations of the existing GO annotated proteins. Meanwhile, Table 3 shows new function annotations produced by the PFP() to the proteins that currently do not have GO molecular function annotation. Figure 4 shows the effect of PFP() to GOA from the overall view which means all of the proteins involved in the experimental dataset whereas Figure 5 uses a sample of 20 *S. cerevisiae* proteins to give a closer view. Both of the new and additional annotations produced by PFP() are showed in these figures to provide a better look at the significant effect of PFP() to the amount of GO function annotations for the experimental dataset used in this study. If we were to use only the current GO functional annotations, it will restrict the function information extraction for the proteins involved thus weakens the result in functional keyword matching process which is in the first filtration phase. As high quality PPI data becomes more available, so does the performance of PFP() in the quest to assign specific and accurate function annotations.

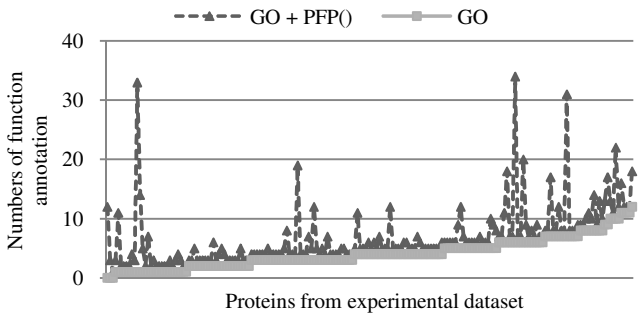


Figure 3. Effect of PFP() to GOA based on the experimental dataset (overall view).

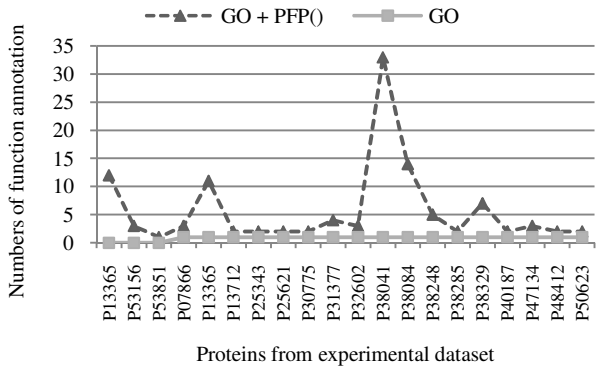


Figure 4. Effect of PFP() to GOA using a sample of 20 *S. cerevisiae* proteins from the experimental dataset.

Table 2. Examples of proteins and their additional function annotations.

Proteins	Predicted GO molecular functions
Cytochrome c oxidase subunit 1 [Swiss-Prot: P00401]	**GO:0005515** Protein binding
Syntaxin-8 [Swiss-Prot: P31377]	**GO:0032266** Phosphatidylinositol 3-phosphate binding
Protein phosphatase PP2A regulatory subunit A [Swiss-Prot: P31383]	**GO:0003823** Antigen binding **GO:0008601** Protein phosphatase type 2A regulator activity **GO:0046982** Protein heterodimerization activity
Dolichyl-phosphate-mannose--protein mannosyltransferase 2 [Swiss-Prot: P31382]	**GO:0000287** Magnesium ion binding

Table 3. Examples of proteins that have no GO molecular function annotation but were assigned newly predicted function annotations.

Proteins	Predicted GO molecular functions
G1/S-specific cyclin CLN3 [Swiss-Prot: P13365]	**GO:0016538** Cyclin-dependent protein kinase regulator activity **GO:0016251** Cyclin-dependent protein kinase regulator activity **GO:0005515** general RNA polymerase II transcription factor activity **GO:0004672** Protein kinase activity **GO:0019209** Kinase activator activity **GO:0003711** Transcription elongation regulator activity **GO:0008353** RNA polymerase subunit kinase activity **GO:0019901** Protein kinase binding **GO:0000043** 4-hydroxybenzoate octa
	prenyltransferase activity
Uncharacterized protein YGL081W [Prot: P53156]	**GO:0004864** Protein phosphatase inhibitor activity **GO:0003729** mRNA binding **GO:0004865** Type 1 serine/threonine specific protein phosphatase inhibitor activity
Protein TEX1 [Swiss-Prot: P53851]	**GO:0051018** Protein kinase A binding

4.2 Identified Keywords

The keywords identification process serves to be highly beneficial as it groups proteins through their general functions criteria rather than using the exact GO functional terms. It increased the possibility of finding proteins that conduct the same processes or functions. We gathered 1100 non-redundant GO term information based on the 4748 *S. cerevisiae* proteins in the experimental dataset. The GO term information was further clustered into several clusters resulting to 31 keywords. The frequency of appearance of the keywords in the training dataset was identified and 10 keywords that showed high frequency were chosen out of 31 keywords. The top ranking keywords represent functions that the proteins in the experimental dataset mostly commit to. In Table 4, we listed the 10 top ranking keywords and the rest remaining 21 keywords that were classified under remaining keywords called 'RK'. We believe that these 10 top ranking keywords that have been picked are the keywords that best reflect the overall general functions of the *S. cerevisiae* proteins in the experimental dataset. To support this, we performed the sensitivity and specificity analysis. First, we calculated the percentage of strength of a certain keyword on the protein pairs in training dataset which is the sensitivity analysis. Then, we performed the specificity analysis which is conducted on the predicted datasets to find the percentage of strength of a certain keyword on the protein pairs in testing datasets. Sensitivity is calculated as follows:

$$\text{Sensitivity}=\frac{\text{number of pairs represented by the keyword}}{\text{total number of pairs in the training dataset}}\times 100=\frac{1}{x}\sum_{i=1}^{x} n_i\times 100 \quad (6)$$

The total number of pairs in the training dataset (experimental dataset) is represented by x. When a keyword represent two proteins in pair i, then n_i=1 and n_i=0 if it is otherwise. Specificity is similarly calculated as equation 7 in which y indicate the total number of pairs in the testing dataset (predicted dataset),

$$\text{Specificity}=\frac{\text{number of pairs represented by the keyword}}{\text{total number of pairs in the test dataset}}\times 100=\frac{1}{y}\sum_{i=1}^{y} n_i\times 100 \quad (7)$$

Table 4. Frequency of keywords that were identified from the experimentally obtained dataset.

Keywords	Frequency
ase activity	3731
Binding	3613
Porter activity	355
Transcription activity	167

Ribosome	126
Carrier activity	78
Receptor activity	59
Exchange activity	39
Dimerization activity	38
Translation activity	34
Remaining keywords (21 keywords)	114

Sensitivity variations among identified keywords are illustrated in Figure 5, with abbreviations for keywords as follows: AS (ase activity), BI (binding), PO (porter activity), TC (transcription activity), RI (ribosome), CA (carrier activity), RE (receptor activity), EX (exchange activity), DI (dimerization activity), TL (translation activity) and RK (remaining keywords). We concluded that the percentage for each identified keywords for the sensitivity analysis as satisfying considering that the average percentage received is 60.78%. 'RK' which refer to the 21 remaining keywords, seems to impose relatively insignificant contribution to the experimental dataset (training dataset) because of its low sensitivity result that is with 50.25%. We further examined the significance of the identified keywords by conducting similar analysis on the four predicted datasets (testing datasets) which we refer as specificity analysis. The predicted PPI datasets that were used in this analysis will be explained in section 4.3. Figure 6 illustrates specificity of the top ranking keywords and also all the 21 keywords that were classified as 'RK'. Specificity varies from 22.74% in PP dataset to 88.43% in MLE dataset. Meanwhile, 'RK' in all four predicted datasets display the lowest specificity compared to the other keywords from 16.01% in PP dataset to 55.78% in MLE dataset. The average percentage (all four datasets) of specificity received is 52.18% which represents the recovery power of the keywords towards all four predicted datasets. Although it seems as both average of sensitivity and specificity did not give a highly confident result, it is still acceptable given the deficiencies suffered by current annotations and also experimental techniques.

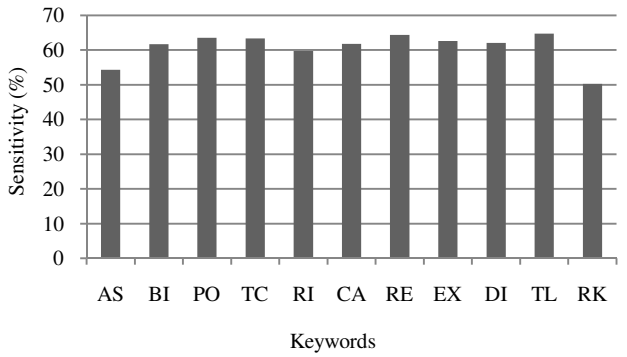


Figure 5. Sensitivity of identified keywords is being presented by each column.

4.3 Predicted PPI

We received 177427 predicted pairs for PP method. Meanwhile, for MI method we managed to gather 343922 non-duplicated predicted PPI pairs. The results of the non-duplicated PPI as well as the amount of DDI for MLE method are 414768 and 15404, respectively. Lastly, for GE method we obtained 7 clusters altogether with 0.538 for the overall average homogeneity. Genes within the same cluster are paired with each other based on the reason that they shared similar co-expressed pattern. We listed the number of proteins in each cluster and the total predicted PPI pairs in Table 5. Results after predictions showed that depending on method applied, the amount of the predicted PPI pairs varied.

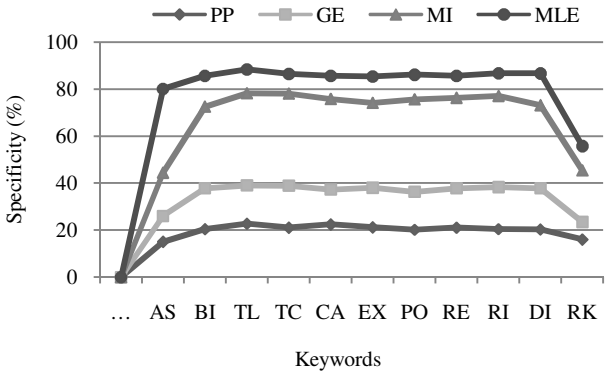


Figure 6. Specificity of the trained identified keywords for all four computational predicted PPI datasets are represented by data points.

Table 5. Result of predictions for each cluster and the total PPI prediction obtained for GE method.

Cluster	Number of proteins in cluster	Predicted PPI pairs
1	1807	1631721
2	1526	1163575
3	613	187578
4	356	63190
5	294	43071
6	91	4095
7	86	3655
Total		**3096885**

4.4 Filtered Datasets

After executing the filtration process where we match the proteins in the PPI predicted pairs according to the interaction rules, the results had showed a huge sum of reduction in every computational prediction methods indicating that the false positive pairs have been partially removed. The results of computational PPI predictions that we received are high especially for the GE method since the proteins were paired to each other within their cluster, followed by MLE, MI and PP methods. Figure 7 presents both raw and filtered datasets. The effect of the filtration process will be statistically evaluated and analysed in the next section.

4.5 Statistical Evaluation Analysis

After conducting the statistical analysis of SNR and strength to all four computational prediction methods, we received results as seen in Table 6 and Figure 8. The results varied among different methods indicating that this proposed computational framework does not exerts the same strength towards different categories of computational predicting methods. From the results, we witnessed the robustness of the prediction from MLE method when the strength showed the lowest among the other methods used in this study. However, it does contribute to the purpose of reducing the false positive that contain in the predicted dataset. PP method showed the strongest strength with 10.0257 and this means that the proposed computational framework gave a strong influence on improving PPI pairs predicted by this method and most likely by methods from this category that is the category that utilized genomics information.

In the efforts to produce better PPI prediction results, we analysed the true positive fraction (TPF) and the false positive fraction (FPF) of the raw and filtered prediction datasets such as in Figure 9. The figure illustrates trendlines for both raw and filtered datasets where we display the TPF and FPF that were resulted from comparisons with the PPI data in experimental dataset. Here, we witnessed that all four computational prediction methods that were conducted based on the proposed computational framework resulted to a much lower FPF value and a much higher TPF value compared to the prediction results that did not apply the proposed computational framework.

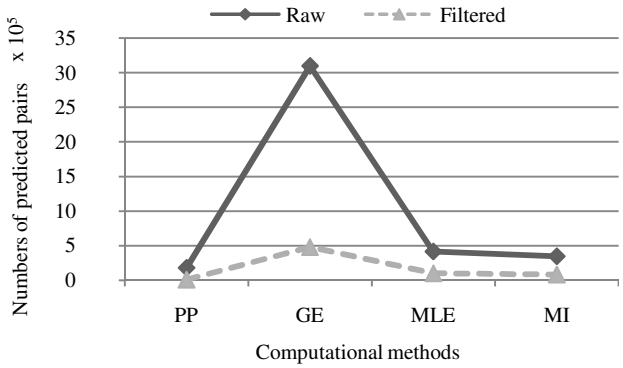


Figure 7. Results of raw datasets (unfiltered datasets) and the filtered datasets of all the computational prediction methods.

Table 6. Results from the statistical evaluation phase, SNR and strength of all the computational prediction methods.

Methods	SNR		Strength
	Raw/Unfiltered	Filtered	
PP	2.3813	23.8741	10.0257
GE	1.8941	8.9035	4.7007
MI	3.2045	12.7193	3.9691
MLE	13.5841	32.7963	2.4143

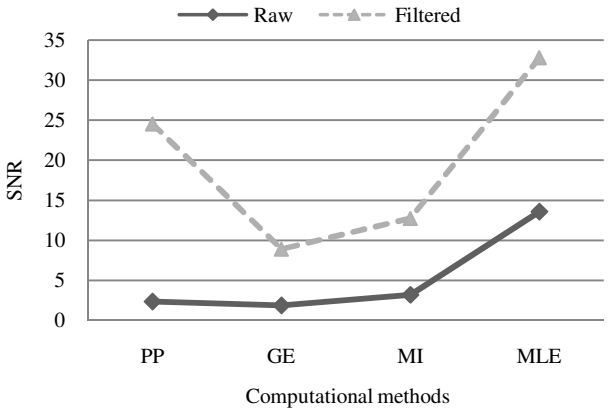


Figure 8. Results of SNR for all four PPI prediction methods.

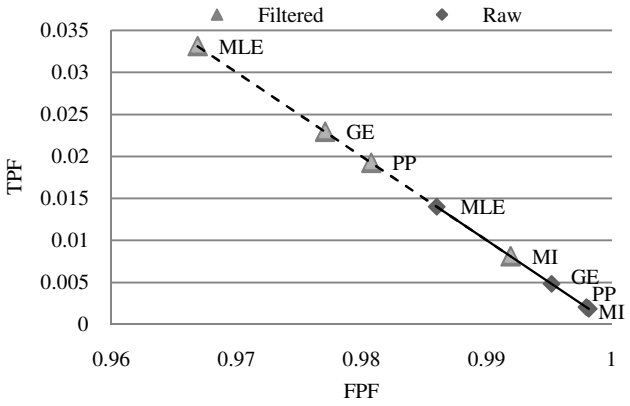


Figure 9. True positive fraction (TPF) and false positive fraction (FPF) in all the computational methods for unfiltered (raw) and after filtered (filtered).

5. CONCLUSIONS

Protein function assignment based on shared interacting domain patterns have been utilized with the purpose of aiding the inconsistencies of GO in order to filter false positive that stem from PPI predictions based on PP, GE, MI and MLE methods. By manipulating similar domain patterns from *S. cerevisiae*, *C. elegans*, *D. melanogaster* and *H. sapiens* species, the method managed to enrich GO molecular function annotations and even assign new annotations for the proteins (*S. cerevisiae*) in the experimental dataset. The information has contributed to the probability of finding matching functional keywords in a pair thus resulted to a higher chance of finding true positive pair in the predicted datasets. The points that represent the TPF and FPF for each computational method appeared to be situated at better positions after filtering. This means that the quality and the reliability of the predicted PPI datasets have increased where huge sums of false positive pairs were successfully discarded. Ultimately, we managed to enhance the confidence level of the datasets by the reduction of false positives which then improves the robustness of the PPI data. The effect of the proposed computational framework will continue to improve as more genes are assigned to GOA and PPI data increase in terms of its quantity

and quality. This computational framework which produced an improved PPI prediction results, will serves as an effective post-prediction process with the goal to reduce false positive in computational PPI predictions.

6. ACKNOWLEDGMENTS

This project is funded by Malaysian Ministry of Higher Education (MOHE) under Fundamental Research Grant Scheme (project no. 78186).

7. REFERENCES

[1] Papin J., and Subramaniam S. 2004. Bioinformatics and cellular signaling. Current Opinion in Biotechnology. 15, 78-81.

[2] Tucker, C.L, Gera, J.F., and Uetz, P. 2001. Towards an understanding of complex protein networks. Trends Cell Biology. 11, 102-26.

[3] Wang, J. 2002. Protein recognition by cell surface receptors: physiological receptors versus virus interactions. Trends in Biochemical Sciences. 27, 122-6.

[4] Reš I., Mihalek, I., and Lichtarge, O. 2005. An evaluation based classifier for prediction of protein interface without using protein structures. Bioinformatics. 21, 2496-501.

[5] von Mering, C., Krause, R., Snel, B., Cornell, M., Oliver, G.S., Fields, S., et al. 2002. Comparative assessment of large scale data sets of protein-protein interactions. Nature. 417, 399-403.

[6] Deane, C.M., Salwinski, L., Xenarios, I., and Eisenberg, D. 2002. Protein interactions: two methods for assessment of the realibility of high-throughput observation. Molecular and Cellular Proteomics. 1, 349-56.

[7] Edwards, A.M., Kus, B., Jansen, R., Greebaum, D., Greenblatt, J., and Derstein, M. 2002. Bridging structural biology and genomics: assessing protein interaction data with known complexes. Trends in Genetics. 18, 529-36.

[8] Saito, R., Suzuki, H., and Hayashizaki, Y. 2003. Construction of reliable protein-protein interaction networks with a new interaction generality measure. Bioinformatics. 19, 756-63.

[9] Qi, Y., Joseph, Z.B., and Seetharaman, J.K. 2006. Evaluation of different biological data and computational classification methods for use in protein interaction prediction. PROTEINS: Structure, Function, and Bioinformatics. 63, 490-500.

[10] Valencia, A., and Pazos, F. 2002. Computational methods for the prediction of protein interaction. Current Opinion in Structural Biology. 12, 368-73.

[11] Wu, J., Kasif, S., and DeLisi, C. 2003. Identification of functional links between genes using phylogenetic profiles. Bioinformatics. 19, 1524-30.

[12] Sun, J., Xu, J., Liu, Z., Liu, Q., Zhao, A., Shi, T., et al. 2005. Refined phylogenetic profiles method for predicting protein-protein interactions. Bioinformatics. 21, 3409-15.

[13] Huang, C., Morcos, F., Kanaan, S.P., Wuchty, S., Chen, D.Z., and Izaguirre, J.A. 2007. Predicting protein-protein interactions from protein domains using a set cover approach. IEEE/ACM Transactions on Computational Biology and Bioinformatics. 4, 1-10.

[14] Marcotte, E.M., Pellegrini, M., Thompson, M.J., Yeats, T.O., and Eisenberg, D. 1999. A combined algorithm for genome wide prediction of protein function. Nature. 402, 83-6.

[15] Chen, Y., and Xu, D. 2003. Computational analyses of high-throughput protein-protein interaction data. Current Protein and Peptide Science. 4, 159-81.

[16] von Mering, C., Huynen, M., Jaeggi, D., Schmidt, S., Bork, P., and Snel, B. 2003. STRING: a database of predicted functional associations between proteins. Nucleic Acids Research. 31, 258-61.

[17] Sun, J., Sun, Y., Ding, G., Liu, Q., Wang, C., He, Y., et al. 2007. InPrePPI: an integrated evaluation method based on genomic context for predicting protein-protein interactions in prokaryotic genomes. BMC Bioinformatics. 8, 414.

[18] Mahdavi, M.A., and Lin, Y. 2007. False positive reduction in protein-protein interaction predictions using Gene Ontology Annotation. BMC Bioinformatics. 8, 262.

[19] Camon, E., Magrane, M., Barrell, D., Lee, V., Dimmer, E., Maslen, J., et al. 2005. The Gene Ontology Annotation (GOA) database: sharing knowledge in UniProt with Gene Ontology. Nucleic Acids Research. 32, D262-6.

[20] Patil, A., and Nakamura, H. 2005. Filtering high-throughput protein-protein interaction data using a combination of genomic features. BMC Bioinformatics. 6, 100.

[21] The Gene Ontology Consortium. 2000. Gene Ontology: tool for the unification of biology. Nature Genetics. 25, 25-9.

[22] Rhodes, D.R., Tomlins, S.A., Varambally, S., Mahavisno, V., Barrette, T., Kalyana-Sundaram, S., et al. 2005. Probabilistic model of the human protein-protein interaction network. Nature Biotechnology. 23, 951-9.

[23] Wu, J., Kasif, S., and DeLisi, C. 2006. Prediction of yeast protein-protein interaction network: insights from the Gene Ontology and annotations. Nucleic Acids Research. 19, 2137-50.

[24] Hsing, M., Byler, K.G., and Cherkasov, A. 2008. The use of Gene Ontology terms for predicting highly-connected 'hub' nodes in protein-protein interaction networks. BMC Systems Biology. 2, 80.

[25] Dyer, M.D., Murali, T.M., and Sobral, B.W 2008. The landscapes of human proteins interacting with viruses and other pathogens. PLOS Pathogens. 4(2), e32.

[26] Lord, P.W., Stevens, R.D., Brass, A., and Goble, C.A. 2003. Investigating semantic similarity measures across the Gene Ontology: the relationship between sequence and annotation. Bioinformatics. 19, 1275-83.

[27] Chen, X., Liu, M., and Ward, R. 2008. Protein function assignment through mining cross-species protein-protein interactions. PLOS ONE. 3(2), e1562.

[28] Crosby, M.A., Goodman, J.L., Strelets, V.B., Zhang, P., and Gelbart, W.M. 2007. FlyBase: genomes by dozen. Nucleic Acids Research. 35, D486-91.

[29] Fujimori, T., Miyazu, T., and Ishikawa, K. 1974. Evaluation of analytical methods using signal-noise ratio as a statistical criterion. Microchemical Journal. 19, 74-85.

[30] Pellegrini, M., Marcotte, E.M., Thompson, M.J., Eisenberg, D., and Yeates, T.O. 1999. Assigning protein functions by comparative genome analysis: protein phylogenetic profiles. PNAS. 96(8), 4285-88.

[31] van Noort, V., Snel, B., Huynen, M.A. 2003. Predicting

gene function by conserved co-expression. TRENDS in Genetics. 19, 238-42.

[32] Date, S.V., Marcotte, E.M. 2003 Discovery of uncharacterized cellular systems by genome-wide analysis of functional linkages. Nature Biotechnology. 21, 1055-62.

[33] Deng, M., Mehta, S., Sun, F., and Chen, T. 2002. Inferring domain-domain interactions from protein-protein interactions. Genome Research. 12, 1540-8.

[34] Mulder, N.J., Apweiler, R., Attwood, T.K., Bairoch, A., Bateman, A., Binns, D., et al. 2002. InterPro: an integrated documentation resource for protein families, domains and functional sites. Briefing in Bioinformatics. 3, 225-35.

[35] Apweiler, R., Bairoch, A., Wu, C.H., Barker, W.C., Boeckmann, B., Ferro, S., et al. 2004. UniProt: the universal protein knowledgebase. Nucleic Acids Research. 32, D262-6.

[36] The Gene Ontology Consortium. AMIGO GO database release 21-10; 2008. DOI= http://amigo.geneontology.org.

[37] Nooren, I.M.A., and Thornton, J.N. 2003. Structural characterization and functional significance of transient protein-protein interactions. Journal of Molecular Biolog. 325, 991-1018.

[38] Date, S.V, and Marcotte, E.M. 2005. Protein function prediction using the Protein Link Explorer (PLEX). Bioinformatics. 21, 2558-9.

[39] Sherlock, G., Boussard, T.H., Kasarski, A., Binkley, G., Matese, J.C., Dwight, S.S., et al. 2001. The Standford Microarray Database. Nucleic Acids Research. 29, 152-5.

[40] Sharan, R., Katz, A.M., and Shamir, R. 2003. CLICK and EXPANDER: a system for clustering anzzzyd visualizing gene expression data. Bioinformatics. 19, 1787-99.

[41] Güldener, U., Münsterkötter, M., Kastenmüller, G., Strack, N,. van Helden, J., Lemer, C., Richelles, J. 2005. CYGD: the comprehensive yeast genome database. Nucleic Acids Research. 33, D364-8.

[42] Salwinski, L., Miller, C.S., Smith, A.J., Pettit, F.K., Bowie, J.U., and Eisenberg, D. 2004. The database of interacting proteins: 2004 update. Nucleic Acids Research. 32, D449-51.

[43] Stark, C., Breitkreutz, B.J., Reguly, T., Boucher, L., Breitkreutz, A., and Tyers, M. 2006. BioGRID: a general repository for interaction datasets. Nucleic Acids Research. 34, D535-9.

[44] Chatr-aryamontri, A., Ceol, A., Palazzi, L.M., Nardelli, G., Schneider, M.V., Castagnoli, L., et al. 2007. MINT: the molecular interaction database. Nucleic Acids Research. 35, D572-4.

[45] Peri, S., Navarro, J.D., Amanchy, R., Kristiansen, T.Z., Jonnalagadda, C.K., Surendranath, V., et al. 2003. Development of human protein reference database as an initial platform for approaching systems biology in human. Genome Research. 13, 2363-71.

[46] Bateman, A., Birney, E., Cerruti, L., Durbin, R., Etwiller, L., Eddy, S.R., et al. 2004. The Pfam protein families database. Nucleic Acids Research. 32, D138-41.