BIOKDD04: Workshop on Data Mining in Bioinformatics
August 22nd, 2004
Seattle, WA, USA

in conjunction with

10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining

Mohammed J. Zaki
Computer Science Department
Rensselaer Polytechnic Institute
Troy, NY 12180, USA
zaki@cs.rpi.edu

Shinichi Morishita
Department of Computational Biology
University of Tokyo
Kashiwa City, Chiba, Japan
moris@k.u-tokyo.ac.jp

Isidore Rigoutsos, PhD
Manager, Bioinformatics & Pattern Discovery Group
IBM Thomas J Watson Research Center
Yorktown Heights, NY 10598
rigoutso@us.ibm.com

Opening Remarks

Bioinformatics is the science of managing, mining, and interpreting information from biological sequences and structures. Genome sequencing projects have contributed to an exponential growth in complete and partial sequence databases. The structural genomics initiative aims to catalog the structure-function information for proteins. Advances in technology such as microarrays have launched the subfield of genomics and proteomics to study the genes, proteins, and the regulatory gene expression circuitry inside the cell. What characterizes the state of the field is the flood of data that exists today or that is anticipated in the future; data that needs to be mined to help unlock the secrets of the cell.

While tremendous progress has been made over the years, many of the fundamental problems in bioinformatics, such as protein structure prediction or gene finding, are still open. Data mining will play a fundamental role in understanding gene expression, drug design and other emerging problems in genomics and proteomics. Furthermore, text mining will be fundamental in extracting knowledge from the growing literature in bioinformatics.

The goal of this workshop is to encourage KDD researchers to take on the numerous challenges that Bioinformatics offers. The workshop features an invited talks from noted expert in the field, and the latest data mining research in bioinformatics. We encouraged papers that propose novel data mining techniques for tasks such as:

- Gene expression analysis
- Protein/RNA structure prediction
- Phylogenetics
- Sequence and structural motifs
- Genomics and Proteomics
- Gene finding
- Drug design
- RNAi and microRNA Analysis
- Text mining in bioinformatics
- Modeling of biochemical pathways

These proceedings contain 10 papers (6 long and 4 short), out of 26 submissions that were accepted for presentation at the workshop. Each paper was reviewed by three members of the program committee. Along with a keynote talk, we were able to assemble a very exciting program.

We would like to thank all the authors, invited speaker, and attendees for contributing to the success of the workshop. Special thanks are due to the program committee for help in reviewing the submissions.

This workshop follows the previous three highly successful workshops: BIOKDD03, held in Washington, DC; BIOKDD02, held in Edmonton, Canada; and BIOKDD01 held in San Francisco, CA. We expect BIOKDD04 to be equally successful.

Workshop Co-Chairs

- Mohammed J. Zaki, Rensselaer Polytechnic Institute
- Shinichi Morishita, University of Tokyo
- Isidore Rigoutsos, IBM T.J. Watson Research Center

Program Committee

• **Srinivas Aluru**, Iowa State U., USA • **Alberto Apostolico**, Prudue U., USA • **Tatsuya Akutsu**, Kyoto U., Japan • **Charles Elkan**, UC San Diego, USA • **Jayant Haritsa**, Indian Inst. of Science, India • **Hasan Jamil**, Wayne State U., USA • **Andreas Karwath**, U. Freiburg, Germany • **George Karypis**, U. Minnesota, USA • **Ross D. King**, U. of Wales, UK • **Jinyan Li**, Inst. for Infocomm Research, Singapore • **Lance A. Liotta**, NIH/NCI, USA • **Ambuj Singh**, UC Santa Barbara, USA • **David Page**, U. Wisconsin, USA • **Srinivasan Parthasarathy**, Ohio State U., USA • **Jignesh M. Patel**, U. Michigan, USA • **Daniel E. Platt**, IBM TJ Watson, USA • **Luc De Raedt**, U. Freiburg, Germany • **Tobias Scheffer**, Humboldt U., Germany • **Karlton Sequeira**, RPI, USA • **Hannu Toivonen**, U. Helsinki, Finland • **Jason Wang**, NJIT, USA • **Wei Wang**, UNC Chapel-hill, USA • **Jiong Yang**, Case Western Reserve U., USA • **Aidong Zhang**, U. Buffalo, USA

Workshop Program & Table of Contents

8:50-9:00am: Opening Remarks

9:00-10:00am: Session I

10:00-10:30am: Coffee Break

10:30-11:15am: Keynote Talk

- Mark Boguski, M.D, Ph.D., Senior Director, Development and Research, Allen Institute for Brain Science; and affiliate faculty, Fred Hutchinson Cancer Research Center & Department of Medicine/Genetics, University of Washington.

11:15-11:55am: Session II

12:00-1:30pm: Lunch

1:30-3:00pm: Session III

3:10-3:30pm: Coffee Break

3:30-4:30pm: Session IV

A New Approach to Protein Structure Mining and Alignment*

Hongyuan Li, Keith Marsolo, Srinivasan Parthasarathy and Dmitrii Polshakov †
The Ohio State University
Columbus, Ohio, USA
li.274@osu.edu, {marsolo,srini}@cse.ohio-state.edu, dpolshak@chemistry.ohio-state.edu

ABSTRACT

One of the largest areas of focus in bioinformatic and data mining research has been on the protein domain. These research efforts have included protein structure prediction, folding pathway prediction, sequence alignment, *ab initio* simulation, structure alignment, substructure detection and many others. In this work, we deal with substructure detection and sequence alignment. Substructure detection is generally defined as the mining of a molecule's 3D structure in order to find interesting/frequent domains. Sequence alignment involves determining the similarity of two (or more) protein molecules based on the how well their amino acid sequences "match." There are potential pitfalls when trying solve both of these problems, however. In the case of substructure mining, focusing solely on structural information can lead to the discovery of biologically irrelevant substructures. With sequence alignment, the alignment results can vary greatly, depending on the substitution matrix used. In this paper we describe a method that combines the benefits of both substructure mining and sequence alignment in an attempt to determine the similarity between protein molecules. In the absence of biological information, our work will quickly and efficiently mine a protein molecule in order to determine frequent local structures. With the addition of biological sequence information, however, our algorithm provides a way to align proteins with similar local structures and sequence, yielding a global alignment between molecules. We present a novel structure mining/alignment algorithm as well as some additional work into a new clustering metric for amino acids based on several different physiochemical properties. This metric is used with our alignment algorithm in order to provide a mechanism for globally aligning protein molecules.

General Terms

Algorithms, Experimentation

*Work funded in part by NSF Career Grant IIS-0347662 (SP,KM)

†Department of Chemistry (HL, DP), Department of Computer Science and Engineering (KM, SP)
Note: Authors are listed in alphabetical order. Each contributed equally to this work.

Keywords

Protein structure alignment, sequence alignment, substructure discovery, multi-scale analysis

1. INTRODUCTION:

With the ever-increasing power and storage capacities of computers comes the ability to process larger amounts of information. Through endeavors such as the Human Genome Project [36] and the Sloan Digital Sky Survey [41], the amount of potential data has increased exponentially, to the point where new techniques are needed to analyze and comprehend it. One of the fastest-growing areas in computer science is that of data mining, or the process of deriving useful relationships and patterns from large stores of data. Data mining has been increasingly applied to problems in the scientific domain, especially bioinformatics, which involves the application of data mining techniques to biological datasets.

One of the richest research areas in bioinformatics has been in the protein domain. Proteins are often studied because they play an important role in a countless number of biological processes, yet there is still a great deal about proteins that is not understood. For instance, a protein can fold spontaneously and reproducibly into a three-dimensional structure when placed into aqueous solution. This transformation occurs in a fraction of a second, yet researchers still have not been able to determine the exact sequence of steps that cause a protein to fold. It is known that a protein's amino acid sequence uniquely determines its three-dimensional structure and that this structure influences the protein's biological function. Thus, if two proteins share a similar structure, they may have a similar biological function. While researchers have found that sequence influences structure, they have not yet determined the exact nature of the link between the two.

Substructure detection involves the mining of a protein's three-dimensional graph in order to find "interesting" (or possibly just frequent) structural motifs [4–6, 9, 12, 18, 25, 30, 33, 43]. By determining whether an previously unclassified protein contains certain structural motifs, one can make inferences as to the role it might play biologically. The problem with substructure detection algorithms is that the analysis methods are often quite complicated and require large amounts of time, memory, and computational resources to execute. With protein sequence mining, there has been a great deal of success in determining the similarity between proteins based on their amino acid sequence, yet through

evolution, it is possible for a protein's sequence to mutate. These mutations may not have any influence over a protein's structure or function, yet may lead to false notions of similarity between molecules. As a result, one would like to create a program that is able to combine the best of both worlds: have the ability to find interesting structural motifs within a protein, and then, using those motifs and a protein's amino acid sequence, construct an alignment between proteins that can be used to determine the similarity between molecules. In this paper we present work that is able to provide such functionality. By adding domain-specific extensions to a previously developed substructure mining algorithm [7, 8, 29, 37] our work makes the following contributions to research in the protein domain:

1. The ability to quickly and efficiently find local substructures within a protein molecule.

2. With the inclusion of biological sequence information, the ability to align local substructures to determine a global alignment between protein molecules.

3. The incorporation of a new classification for amino acids based on physio-chemical properties that allows for partial matching and partial alignment between molecules.

2. RELATED WORK AND BACKGROUND

2.1 Sequence Alignment

The idea of using alignment to determine protein similarity is not a new one. Programs like BLAST [15] and its refinements PSI-BLAST and gapped BLAST [16] have been used to align proteins based on their amino acid sequence. With the completion of the Human Genome Project [36] and other genome mapping initiatives, there is obviously a great need for such an alignment method. However, the number of new protein structures is growing enormously as well. The Protein Data Bank (PDB) [3] currently holds over 22,000 protein structures and is growing by almost 4,000 structures every year.

Much information about the structure of proteins can be found in the Structural Classification of Proteins (SCOP) database [34]. The SCOP database provides "a detailed and comprehensive description of the structural and evolutionary relationships of proteins," including information on a protein's secondary and tertiary structure. This information is derived by the visual inspection of the proteins in the PDB. The SCOP database is arranged into four different hierarchical levels: Class, Fold, Superfamily and Family. Proteins in the same Class share similar secondary structure information, while proteins within the same Fold have similar secondary structures that are arranged in the same topological configuration. Proteins within the same Superfamily show clear structural homology and proteins within the same Family exhibit a great deal of sequence similarity and are thought to be evolutionarily related.

2.2 Structure Alignment

There have been a number of methods proposed to compare protein structures. Some methods compare the secondary structures of the proteins; others try to align proteins based simply on their backbone configuration. A number of public tools exist that provide some type of alignment/similarity function, including DALI, STRUCTAL and LOCK. A brief description of each follows.

DALI [21] is based on the alignment of two-dimensional distance matrices, with the matrix values representing the distances between the C_α atoms of a protein. The algorithm attempts to find patterns of similar distances within two matrices. These patterns are combined with the intention of maximizing the number of atoms and minimizing the root mean square distance (RMSD) between them. DALI also uses a Monte Carlo optimization [32] to prevent the algorithm from quickly reaching a local minimum.

The STRUCTAL [17] algorithm uses an iterative dynamic programming [2] approach to align two proteins. The principal behind the algorithm is to minimize the RMSD between two protein backbones. First, the distance between all C_α carbons is computed. These distances are converted into a scoring matrix. Standard dynamic programming is employed to compute the optimal alignment of the two proteins. Since the solution to this algorithm depends heavily on the starting alignments of the two proteins, several different starting configurations are used.

LOCK [39] attempts to align proteins by using hierarchical structure superposition. A protein is decomposed into its secondary structures, which are represented as a series of vectors. A scoring matrix is created based on the vectors of the two proteins being aligned. Dynamic programming is then used to find the best local alignment between the vectors. Next, the algorithm attempts to iteratively minimize the RMSD between pairs of nearest atoms. Finally, a core of well-aligned atoms is created and the algorithm attempts to minimize the RMSD of the core.

2.3 Substructure Analysis

Discovering important structures in molecular datasets has been the focus of many recent research efforts in scientific data analysis [4–6, 9, 12, 18, 25, 30, 33, 43]. These efforts have targeted substructure analysis in small molecules, material defect analysis in molecular dynamics simulations, and more recently in macromolecules such as proteins and nucleic acids [21, 24, 46].

Several methods for secondary level motif finding in proteins have been proposed in the past. An algorithm based on subgraph isomorphism was proposed in [33]; it searches for an exact match of a specific pattern in a database. The search for distantly related proteins using a graph to represent the helices and strands was proposed in [25]. An approach based on maximally common substructures between two proteins was proposed in [18]; it also highlights areas of structural overlap. SUBDUE [9] is an approach based on Minimum Description Length for finding patterns in proteins. Another graph based method for structure discovery, based on geometric hashing, was presented in [43]. Recent work on graph data mining is also related to this effort [9, 18, 25–27, 33, 43, 45].

2.4 MotifMiner Toolkit

Our own attempts at substructure detection have resulted in the development of an extensible prototype toolkit, MotifMiner [7,8,29,37], that detects frequently occurring structural motifs. We have conducted a fairly in-depth evaluation of MotifMiner on various datasets, from pharmaceutical data [7], to tRNA data, to protein data (from the

PDB) [7,37] to data obtained from molecular dynamics simulations [6]. As stated previously, MotifMiner was designed to be extensible and here we present several extensions to the toolkit, some domain-neutral, others targeted specifically to proteins.

3. ALGORITHMS

The work described here is based on the MotifMiner project introduced in Section 2.4. A discussion of the general algorithm can be found in Section 3.1. Several basic extensions to the MotifMiner toolkit have been implemented and are presented in Section 3.2. A number of domain-specific constraints for the mining and alignment of proteins have also been developed. They are discussed in Section 3.3.

3.1 Background

MotifMiner represents the interaction between a pair of nodes A_i and A_j, as a *mining bond*. A node can be an atom, an amino acid, a secondary structure, etc., depending on the resolution desired. A mining bond $M(A_iA_j)$ is a 3-tuple of the form:

$$M(A_iA_j) = \{A_itype, A_jtype, AttributeSet(A_i, A_j)\}$$

The information contained in $AttributeSet(A_i, A_j)$ can vary depending on the resolution of the structure being represented. For instance, if the resolution of the structure is at the atomic level, $AttributeSet(A_i, A_j)$ could contain the distances between atoms A_i and A_j. At the secondary structure level, $AttributeSet(A_i, A_j)$ might contain the secondary structure type (α-helix or β-sheet), the number of residues within the secondary structure and so forth. Using the above definition, a *k-nodeset* is a substructure containing k connected (within a user-specified range) nodes, and is represented as:

$$X = \{\mathbf{S_X}, A_1, A_2, \ldots, A_k\},$$

where A_i is the i^{th} node and $\mathbf{S_X}$ is the set of mining bonds describing the nodeset. By defining pairs of nodes with mining bonds, the graph is completely represented, such that two nodesets X and Y are considered to be the same substructure if $\mathbf{S_X} = \mathbf{S_Y}$. Since we only deal with atoms in this work, we will refer to nodesets as *atomsets*. In addition, atomsets with a similar set of mining bonds are said to belong to the same *atomset family*, or *motif*.

Additionally, MotifMiner uses the following principles to generate frequent substructures: 1) *Range pruning* to limit the search for viable strongly connected sub-structures, 2) *Candidate pruning* [1], for pruning the search space of possible frequent structures, 3) *Recursive Fuzzy Hashing* for rapid matching of structures (to determine frequency of occurrence), and finally 4) *Distance Binning and Resolution* to work in conjunction with recursive fuzzy hashing to deal with noise in the input data.

Range pruning and candidate pruning reduce the candidate search space, thereby reducing the memory footprint and significantly improving the scalability of the algorithm. The biological motivation behind range pruning is that even though molecules are made up of atoms that interact with one another, there is only a finite distance over which such an interaction can occur. At a larger distance, the interaction between two atoms is essentially negligible and two atoms can be considered independent. As a result, by having a user-specified range parameter, it is possible to cut down on the number of potential atomsets.

1. Prune *infrequent* atoms (*1*-atomsets)
2. Generate candidate *2*-atomsets from *frequent* atoms
3. Prune *infrequent 2*-atomsets
4. $k = 3$
5. **while** (| *frequent* k-atomsets |> 0)
6. Generate candidate k-atomsets from *frequent (k-1)*-atomsets
7. Prune *infrequent* k-atomsets
8. $k = k + 1$

Figure 1: Local substructure discovery algorithm

Recursive fuzzy hashing, which is similar in principle to geometric hashing [28, 44], was designed to efficiently handle noise effects in data [37]. The idea behind distance binning and resolution is the data mining principle of discretization [13]. The raw Euclidean distance between two atoms is discretized by binning; This task is accomplished by choosing a *resolution* value and dividing the inter-atom distance into equi-width bins based on this value, represented efficiently as bits in the mining bond. Binning of the data simplifies calculations and helps MotifMiner handle *minor* fluctuations in distance.

As shown in Figure 1, atomsets of size *(i+1)* are derived by combining two frequent atomsets of size i that differ by one atom. Once an atomset has been generated, its frequency is determined using the following metrics:

- *atomsetSupport*- The number of atomsets in the atomset family.
- *coverRate*- The percentage of molecules that contain at least one atomset from the atomset family.

The minimum support thresholds for both parameters can be specified by the user.

3.2 Basic Extensions

In this section we present several basic improvements to the original MotifMiner algorithm. These extensions are domain-independent and can be applied to bio-molecular data of any type. These extensions were borne out of experimental testing of the original MotifMiner algorithm. They represent ways to improve both the running time and the quality of the results.

3.2.1 Variable Resolution

In the original version of MotifMiner, the resolution parameter is used to handle noise in the input data. One drawback with resolution in the original version is that the parameter is not flexible and cannot handle modulation in structure. The differences between two substructures can be very small in the short range, but as the substructures become larger, those differences are accumulated and magnified. Thus, the resolution is now variable. As the distance between two atoms increases, so does the resolution. This allows for the identification of larger similar structures. Figure 2 shows an example of this principle. With a sliding resolution, it is possible to identify an α-helix and a smaller helix from a helix-bundle as similar. Without variable resolution, the

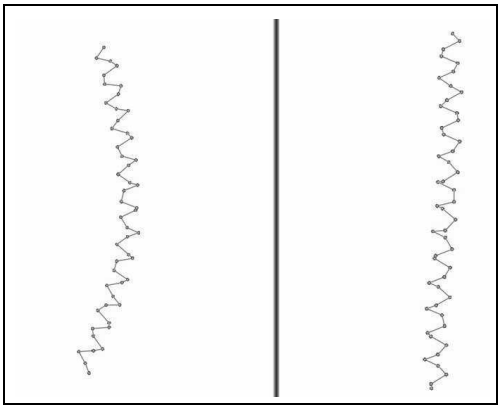


Figure 2: The twisted α-helices found in 1A02_J and 1A02_F (subunits of nuclear transcription complex).

```
1. Generate candidate atomset A
2. If atomCount_A ≤ minLinkage then
3.    ∀ atoms i ∈ A
4.       ∀ atoms j ∈ A (j ≠ i)
5.          if dist(atom i, atom j) > Range then
6.             discard A
7. Else
8.    ∀ atoms i ∈ A
9.       ∀ atoms j ∈ A (j ≠ i)
10.         count = count + 1
11.         if count < minLinkage then
12.            discard A
```

Figure 3: Local Structure Linkage Algorithm

bend in helix on left would have prevented it from being matched with the helix on the right.

3.2.2 Boundary Conditions

Another potential problem when dealing with noise in the input data is the handling of boundary conditions. For instance, if the range is specified to be 5Å, and the distance between two atoms i and j is 4.99Å, a mining bond will be created between the atoms. If the distance between them is 5.01Å, however, no bond will be created, which can cause problems when trying to determine substructure frequency. As a result, a mining bond will be created when the distance between two atoms is just over the range value.

3.2.3 Local Structure Linkage

The notion behind Local Structure Linkage is that an atomset should contain a minimum number of "close" points. In this case, "close" means that the distance between two atoms is less than the user-specified *Range* value. The minimum number of points is a user-specified parameter designated *minLinkage*. In most experiments, *minLinkage* was set to four.

The Local Structure Linkage algorithm is presented in Figure 3. The effect of the algorithm is to ensure that every atom in an atomset is within a distance *Range* of at least *minLinkage* atoms. Additionally, Local Structure Linkage makes use of another parameter: *initialRange*, where *initialRange* $\leq$ *Range* (see Figure 4). The effect of *initialRange* is to guarantee that each atom has at least one close neighbor and to eliminate many meaningless substructures. The results of an experiment testing the *minLinkage* algorithm are shown in Figure 5. Twenty-three molecules from each of the four major SCOP [34] fold classes (5-α, 7-β, 6-$\alpha+\beta$, 5-$\alpha\backslash\beta$) were mined looking for substructures with a minimum coverage of three molecules (a *coverRate* of 13%). Several substructures were found and are presented in Figure 5.

```
1. ∀ atomsets A with atomCount = 2:
2.    for atoms i, j ∈ A (j ≠ i)
3.       if dist(atom i, atom j) > initialRange then
4.          discard A
```

Figure 4: Effect of *initialRange* Parameter

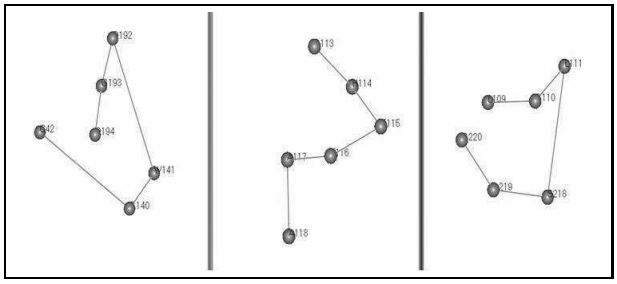


Figure 5: Examples of 6-atom local structures. Left, an unspecified local structure. Middle, partial α-helix. Right, partial anti-parallel β-sheet. Each of these three substructures occurs more than 100 times among the 23 molecules.

3.3 Domain Constraints

One of the most important contributions of this work is the incorporation of domain constraints into the original MotifMiner algorithm. Recall that MotifMiner was intended to be a general framework that could be used across multiple domains. By incorporating domain constraints, [35,40], it is possible to increase the utility of the original framework. Such constraints enable the researcher to interrogate the data while incorporating specific domain knowledge in the process. We have identified several such constraints which are described below.

3.3.1 Abstraction Using α-carbons

All proteins contain a backbone that is formed by peptide bonds. This substructure is very frequent and generates a number of trivial atomset families. Using the α-carbon of the amino acid as an abstraction of the peptide bond is a good way to reduce the number of atoms that need to be examined and enhance the speed of the algorithm. We do lose information about the chemical linkage of the peptide bond with such an abstraction, but compensate for the loss by including information about the amino acid sequence.

3.3.2 Sequence-based Pruning of Motifs

We also incorporate domain constraints that integrate sequence information. This has useful possibilities for the alignment of two proteins and also results in the detection of biologically relevant motifs. Specifically, the algorithm has an option wherein candidate atomsets can be pruned based

on the relative sequence ordering of the atomsets. In terms of implementation, this constraint is relatively straightforward. In addition to the mining bond information, information about the sequence order is maintained in the *Attribute-Set* parameter of the mining bond. When two substructures are compared, we first compare the relative sequence ordering and then match the substructures. We demonstrate the use of this constraint in our global alignment algorithm (Section 3.4).

3.3.3 Approximate Matching of Amino-Acids Based on Physio-Chemical Properties

As an additional extension, we now support approximate sequence matches where a particular amino acid is replaced by a label that represents amino acids which share similar physio-chemical properties (hydrophobicity, helical propensity, etc.). To implement this feature, we used a multi-dimension description of the amino acid space that included a large number (243) of physio-chemical properties that were collected from a number of different sources. In addition, we extended that list of physio-chemical properties with properties obtained from quantum chemical calculations. We used the Gaussian 98[1] program to compute properties such as ground state energy, dipole moment, and vibration frequency for all 20 amino acids.

To reduce the inherent redundancy in the physio-chemical property space, we relied on the technique of multi-scale analysis [42]. This method involves the multi-dimensional scaling of the high-dimension physio-chemical property space to a lower dimensional space using a PCA-style reduction [23]. We found that the first five Eigenvectors sufficed to capture more than 95% of the total inertia of the data. Figure 6 shows the projection of the amino acids on the first two principal-component dimensions (left), and the first and third principal-component dimension (right). After computing the eigenvectors, we used the K-means clustering algorithm [31] to group amino acids by Euclidean distance in 5D space. The result of clustering is shown in Table 1.

Some of the clusters in Table 1 are similar to the results found in [42]. When K=4, for example, residues I, V, L, F, and M fall into the same cluster. This cluster consists of hydrophobic amino acids. The cluster of amino acids W, Y and C consists of polar residues. At high K values (K >5) this cluster separates into two, one of which contains just the aromatic residues W and Y. Another noticeable cluster found in several different levels contains the small residues N, D, S, T, G and P. As shown in Table 1, residues G and P always fall into the same cluster. This result agrees with experimental observation, as it is known that residues G and P play an important role in the determining the 3D architecture of a protein [38]. They are frequently located in the linkage between secondary structures; for example, between two α-helices or between an α-helix and a β-sheet.

To model the cost of a replacement we use the following principle, depending on whether a coarse-grained or fine-grained cost model is desired. For a coarse-grained level, the cost of a replacement is 1 if two amino acids are in different clusters and 0 if they are in the same cluster. At a more fine-grained level we simply use the distances between amino acids in the scaled dimensional space to quantify the cost of replacement. A user-specified threshold determines whether a replacement or a series of replacements in a structure is acceptable or not.

[1] http://www.gaussian.com

1. Generate i-atomsets using the local substructure discovery algorithm and a *coverRate* of 100%
2. **If** there exists any *ambiguity* among atomsets **then**
3. Increment i, go to step 1 and repeat until the *ambiguity* is resolved.
4. **Else**
5. Begin alignment.

Figure 7: Ideal Alignment Preprocessing

1. Generate i-atomsets using the local substructure discovery algorithm and a *coverRate* of 100%.
2. **If** there exists any *ambiguity* among atomsets **then**
3. **If** *out-of-memory* **then**
4. Force alignment.
5. **Else**
6. Increment i, go to step 1 and repeat until the *ambiguity* is resolved or *out-of-memory*.
7. **Else**
8. Begin alignment.

Figure 8: Modified Alignment Preprocessing

3.4 Global Alignment

The most significant contribution of this work is the development of a global alignment method that aligns protein molecules based on their structure as well as their sequence. The alignment algorithm works by generating frequent local substructures and then, starting with the largest local structures discovered, attempts to assemble an alignment between two molecules.

Alignment Preprocessing

Before the global alignment of two molecules can occur, several preprocessing steps must be taken, starting with the generation of local substructures. A high-level presentation of the preprocessing steps is given in Figure 7.

In the ideal case, the substructure generation algorithm would be able to execute as shown in Figure 7. In practice, however, the number of frequent atomsets is usually very large, often to the point where they do not all fit into memory. When this occurs, we say that the local substructure discovery algorithm is *out-of-memory*. As a result, we must modify the preprocessing steps we take before the alignment can begin. The modified algorithm is shown in Figure 8.

In the algorithms presented in Figures 7 and 8, the term *ambiguity* is used to denote when there are atomsets from each molecule that belong to the same atomset family but do not contain exactly the same types of atoms (this can occur due to recursive fuzzy hashing). Thus, it is possible for a single atom in an atomset to align with multiple atoms in the other atomsets in the family. For example, given the 4-atomset families shown in the top table of Table 2, there is ambiguity when aligning atom D. It can align with either atom D' or atom E.

In order to solve this problem, the substructures at the next level are generated to see if they resolve the ambiguity. Since

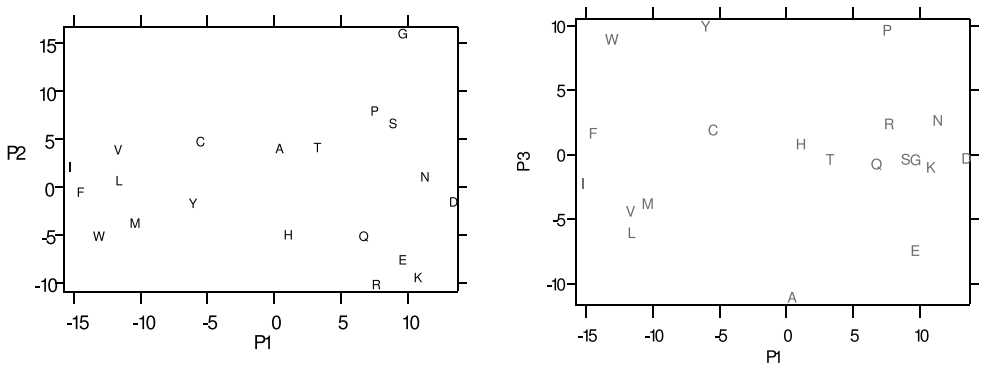


Figure 6: Multi-scale projection of amino acids on Components 1 and 2 (left) and Components 1 and 3 (right).

K	Clusters
2	{A,R,N,D,Q,E,G,H,K,P,S,T}{C,I,L,M,F,W,Y,V}
3	{A,R,Q,E,H,K}{N,D,G,P,S,T}{C,I,L,M,F,W,Y,V}
4	{A,R,Q,E,H,K}{N,D,G,P,S,T}{C,W,Y}{I,L,M,F,V}
5	{A,E}{R,Q,H,K}{N,D,G,P,S,T}{C,I,L,M,F,V}{W,Y}
6	{A,E}{R,Q,H,K}{N,D,S,T}{C,I,L,M,F,V}{G,P}{W,Y}
7	{A,E}{R,Q,H,K}{N,D,S,T}{C}{G,P}{I,L,M,F,V}{W,Y}
8	{A,E}{R,Q,H,K}{N,D,S,T}{C}{G,P}{I,M,V}{L,F}{W,Y}

Table 1: K-Means clustering of amino acids based on multi-dimensional scaling

Family 1	Family 2
ABCD	ABCD
A'B'C'D'	A'B'C'E'
Level 4	

Family 1
ABCDF
A'B'C'D'F'
Level 5

Table 2: Ambiguity between Families. In the 4-atomsets, atom D can possibly align with atoms D' and E. By growing the atomsets to the next level, the ambiguity is resolved.

the atom sequences of Family 2 differ, it will not be expanded at the next level. The atomsets of Family 1 *will* be used in the next level, however. After the next step of the the substructure discovery phase, suppose Family 1 now contains the atom sequences shown in the bottom table of Table 2. There is no longer any ambiguity. D can align with D' and F will align with F'. In this manner, the ambiguity is resolved. As shown in Figure 8, the alignment preprocessing algorithm runs until there are no ambiguous substructures between the molecules or until the program runs out of memory, whichever comes first. Once this point is reached, the assembly of the alignment between the molecules can begin.

Initial Alignment Assembly

When the alignment preprocessing algorithm finishes, we are left with two possible cases. In the first case, the algorithm is able to finish without any ambiguity among the atomsets. When this occurs, all of the atomsets at the highest level reached by the substructure generation algorithm (i.e. the largest frequent substructures discovered) are used as the basis for the starting alignment. This is considered to be *normal assembly*. The other case occurs when the algorithm runs out of memory before resolving all of the ambiguities between atomsets (i.e. atomsets contained in the same atomset family do not have exactly the same atom types). When this occurs, the algorithm is said to start with *forced assembly*. Before assembling the alignment, the algorithm attempts to find the atomset families that have the fewest conflicts with the other atomset families at the same level (in this case, the highest level reached by the substructure generation algorithm before running out of memory). The total number of conflicts is defined as the number of sequence conflicts between atomsets in the same family. For example, suppose an atomset family contained the 3-atomsets ABC and ABD. This family would contain one conflict: the conflict between atoms C and D. Given the family of 3-atomsets ABD and AEF, there would be two conflicts: atoms BD and atoms EF. The algorithm attempts to find the atomset families with the smallest number of conflicts and use them as the starting alignment.

Alignment Assembly

Once the initial alignment has been determined, the alignment assembly can begin. Suppose that the largest substructures found by the initial alignment algorithm are of size n. The assembly algorithm examines the atomsets at level n-1 and determines whether there are any conflicts (using the measure of conflict defined above) between those atomsets. If there are, the alignments with fewer conflicts are given a higher priority. Any candidate atomset (i.e. non-conflicting or a conflicting with a high priority) at this new level is

1. Determine the initial (existing) alignment from level-n atomsets.
2. $n = n - 1$.
3. **while** $n > minLevel$
4. Determine conflicts among level-n atomsets.
5. Remove any atomset that conflicts with the existing alignment.
6. Merge any remaining atomsets with the existing alignment. The resulting set becomes the new existing alignment.
7. $n = n - 1$
8. **Return** the set of atomsets as the global alignment.

Figure 9: Alignment Assembly Algorithm

checked against the existing alignment (i.e. the larger atomsets). If there is any conflict between the candidate atomset and an atomset in the existing alignment, the candidate atomset is removed. Once this step has completed, all of the remaining candidate atomsets are added to the existing alignment. The algorithm then examines the atomsets at the next lower level. These steps repeat until the algorithm reaches a lower limit of potential atomset size that is specified by the user. A pseudo-code description of the algorithm is shown in Figure 9.

4. VALIDATION

In the following examples, we present the results of several experiments that serve as a preliminary validation of our global alignment algorithm.

4.1 Alignment of FHA Domains

The FHA domain is a phospho-protein binding domain. It was originally identified using sequence alignment [20]. However, FHA domains have very few conserved residues (only three residues are completely conserved) and sequence alignment only detected the core region. Later, after the structures of FHA domains were solved, the full domain was demonstrated to cover a much larger region than the core region. We used our global alignment to align the proteins Rad53 and Chk2. The aligned result is very similar to those obtained through manual alignment [14]. The results are shown in Figure 10.

4.2 Alignment of Sequentially Distinct Proteins

Pair-wise structural alignment generates a number of possible sequence alignments that are very hard to align using just a scoring matrix. To give one example, we found that proteins pdb1a2y_B and pdb1a4j_L give an alignment of 49 α-carbons. These corresponding residues have a very similar substructure (Figure 11, top). The resulting sequence alignment of the substructure is shown in Figure 11, bottom. We attempted to align these proteins based on sequence information only, using the scoring matrices BLOSUM 62 [19] and PAM 250 [10, 11]. Neither scoring matrix was able to give a clear result, however (Results omitted due to space constraints).

Sequence alignment has two major difficulties: How to choose scoring matrix and how to estimate gap cost. These two

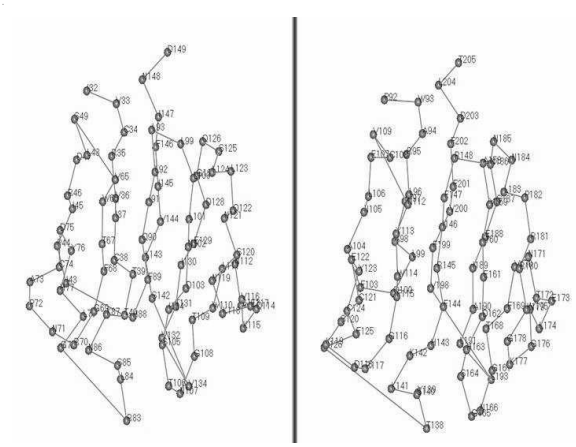
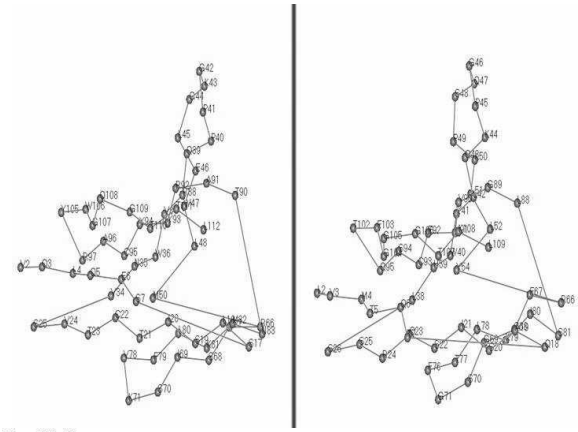


Figure 10: Structural alignment of two FHA domains. FHA1 of Rad53 (left) and FHA of Chk2 (right)



```
74_pdb1a2y_B
2-VQLQES^SLSITCTVS^VNWVRQPPGKGLEWL_M^RLSISK^VFLKMN^TARYYCAR^YWGQGTTL-112
113_pdb1a4j_L
2-LVMTQT^QASISCRSS^LHWYLQKPGQSPKLL_Y^RFSGSG^FTLKIS^LGVYFCSQ^TFGGGTKL-109
```

Figure 11: Sequence alignment of pdb1a2y_B and pdb1a4j_L "_" indicates one space, "^" indicates more than 2 spaces

problems no longer exist in structure-aided sequence alignment. We can give solid parameters to control the similarities of the structures and if there is any gap in the sequence, it is omitted by structure alignment. Thus, structure is more conserved than sequence since all amino acids share the backbone structure.

4.3 Alignment using Physio-Chemical Properties

Since the structural alignment algorithm in this paper uses α-carbons only, the side chain information is ignored to some extent. In many proteins, however, the side chain plays an important role in their activity. Adding amino acid constraints helps defray this loss and such constraints will also help identify residues that contribute more than just backbone linking in structure. Fewer α-carbons need to be aligned which usually speeds up the program.

As a final example, in Figure 12 we show the results of align-

Cluster	# Proteins	Function
1	23	Antibodies
2	11	Hydrolases (mainly serine proteases)
3	7	Transferases (mainly kinases)
4	6	Oxygen Transport (including heme proteins)
5	5	Oxidoreductase (Dehydrogenease)
5	5	Haloperoxidase

Table 3: Top six protein clusters based on the alignment of 312 non-redundant proteins and an alignment threshold of at least 56 atoms

ing two calcium-binding proteins, 1AHR and 5CPV, with the inclusion of physio-chemical properties (amino acid constraints) and without. Our alignment was comparable to one obtained through DALI and it should be noted that we able to verify the existence of the calcium-binding site in our results.

4.4 Alignment-Based Clustering

As an experimental test of our alignment algorithm, we ran an all-against-all alignment of 312 non-redundant (sharing less than 20 amino acids in sequence) proteins and then clustered the results based on the number of atoms that can be aligned between the molecules. We set an alignment threshold of 56 (meaning at least 56 atoms can be aligned between the molecules) and were left with 218 clusters. Most of the clusters contained a single protein molecule, however there were several clusters that contained multiple proteins, and even more striking, the cluster proteins showed functional similarity in addition to their structural similarity. The clusters containing more than five proteins are shown in Table 3. As mentioned above, the dataset uses only non-redundant proteins. Thus, most of the closely related proteins are removed from the dataset. However, antibodies are generated through gene shuffling, which leads to large sequence diversity. The clustering results shows that these antibodies, though different in sequence, still share structural similarity.

4.5 Comparison with DALI

In this work we present a new method for the alignment of protein molecules based on local substructures as well as sequence information. There are a number of other publicly-available protein alignment methods that work based on structure, DALI being one of the most popular. We reran several of our experiments using DALI and DaliLite (a stand-alone version of the DALI Server) [22] to see how our results compared to the results returned by those programs. We found that our results were comparable to what was returned by DaliLite, differing by only a few amino acid residues at most. Aligning proteins pdb1a2y_B and pdb1a4j_L (discussed in Section 4.2) with DaliLite yielded the same amino acid sequence that our program found (Figure 11, bottom). In addition, the running time of our algorithm was equivalent to the running time of DaliLite (or faster). Our algorithm runs in a completely different fashion than DALI (and DaliLite), so it is difficult to compare running times. DALI works by computing a pair-wise distance matrix for each protein and then uses a Monte Carlo optimization procedure to try and minimize the distance between the matrices. The resulting alignment is returned as the best alignment between proteins. Our algorithm works by mining local substructures and then using the underlying sequence information to determine conflicts between structures. Structures that do not conflict are aligned. Those that do are not. It is interesting to note that although both methods are orthogonal in nature, they produce consistent results, provided that we do not include physio-chemical properties. When we do include such properties, we achieve results that, while more concise, still retain their biological relevance.

5. CONCLUSIONS

In this work we present extensions to MotifMiner that allow for the efficient detection of substructures in protein molecules using both biological and structural information. These extensions enable us to detect substructures that vary due to the noise inherent in protein data and to approximate a molecule's amino acid sequence based on varying physio-chemical properties. We have tested our algorithm against a well-established structural alignment tool, DALI, and found that our work performs favorably, even providing some benefits not available in DALI. One benefit that our algorithm has over DALI is that we are able to handle the chirality inherent in some protein molecules. Chirality refers to the "handedness" of a protein. By only dealing with pair-wise distances, DALI is not able to distinguish between chiral proteins. Our algorithm can make such a distinction. In addition, we have the ability vary the physio-chemical properties in our cost analysis. With further testing, we hope to provide more examples as to the usefulness of our algorithm as well as a statistical metric that can be used to determine the quality of the results returned by our program.

6. ACKNOWLEDGEMENTS

We would like to thank the anonymous reviewers of this paper for their comments and suggestions.

7. REFERENCES

[1] R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In *20th VLDB Conf.*, September 1994.

[2] Alfred V. Aho and John E. Hopcroft. *The Design and Analysis of Computer Algorithms*. Addison-Wesley Longman Publishing Co., Inc., 1974.

[3] F. C. Bernstein, T. F. Koetzle, G. J. B. Williams, E. F. Meyer Jr., M. D. Brice, J. R. Rodgers, O. Kennard, T. Shimanouchi, and M. Tasumi. The protein data bank: A computer-based archival file for macromolecular structure. *J. Mol. Biol.*, 112:535–542, 1977.

[4] C. Borgelt and M. Berthold. Mining molecular fragments: Finding relevant substructures of molecules. In *IEEE International Conference on Data Mining*, Dec 2002.

[5] L. P. Chew, D. Huttenlocher, K. Kedem, and J. Kleinberg. Fast detection of common geometric substructures in proteins. *RECOMB*, 1999.

[6] M. Coatney, S. Mehta, A. Choy, S. Barr, S. Parthasarathy, R. Machiraju, and J. Wilkins. Defect detection in silicon and alloys. In *IEEE Workshop on Visualization in Bioinformatics and Cheminformatics*, 2002.

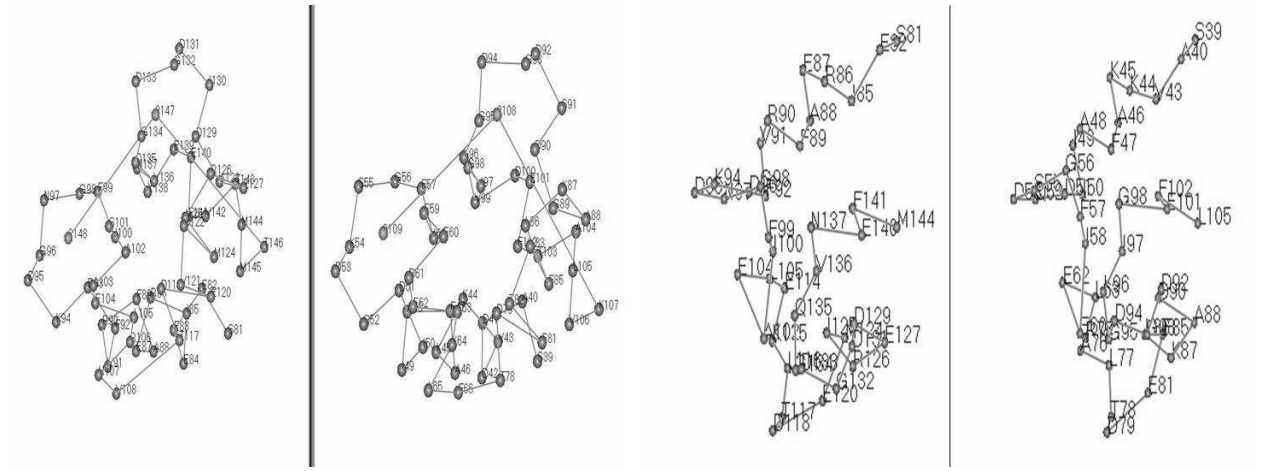


Figure 12: The result of structurally aligning two calcium-binding proteins. The two proteins (1AHR,5CPV) on the left were aligned using structure alignment. The same two proteins were then aligned using approximate matching based on physio-chemical properties. It is apparent that the alignment on the right is contained within the structure-based alignment, however, the structure on the right is, as a whole, more biologically active than the structure on the left. This gives credence to the idea that the incorporation of physio-chemical properties into the alignment algorithm produces more biologically relevant results without losing the important structural motifs.

[7] M. Coatney and S. Parthasarathy. Motifminer: A general toolkit for efficiently identifying common substructures in molecules. In *IEEE International Conference on Bioinformatics and Bioengineering (to appear)*, 2003.

[8] M. Coatney and S. Parthasarathy. Motifminer: Efficient discovery of common substructures in biochemical molecules. In *Knowledge and Information Systems, to appear*, 2003.

[9] D.J. Cook, L.B. Holder, S. Su, R. Maglothin, and I. Jonyer. Structural mining of molecular biology data. *IEEE Engineering in Medicine and Biology*, 20(4):67–74, 2001.

[10] M. O. Dayhoff and R. M. Schwartz. Matrices for detecting distant relationships. In M. O. Dayhoff, editor, *Atlas of Protein Sequence and Structure*, volume 5 of *3*, pages 353–358. Natl. Biomed. Res. Found., Washington, DC, 1978.

[11] M. O. Dayhoff, R. M. Schwartz, and B. C. Orcutt. A model of evolutionary change in proteins. In M. O. Dayhoff, editor, *Atlas of Protein Sequence and Structure*, volume 5 of *3*, pages 345–352. Natl. Biomed. Res. Found., Washington, DC, 1978.

[12] L. Dehaspe, H. Toivonen, and R. King. Finding frequent substructures in chemical compounds. In *The Fourth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1998.

[13] J. Dougherty, R. Kohavi, and M. Shami. Supervised and unsupervised discretization of continuous features. In *ICML*, 1995.

[14] D. Durocher, I. A. Taylor, D. Sarbassova, L. F. Haire, S. L. Westcott, S. P. Jackson, S. J. Smerdon, and M. B. Yaffe. The molecular basis of fha domain:phosphopeptide binding specificity and implications for phospho-dependent signaling mechanisms. *Mol Cell*, 6(5):1169–82, Nov 2000.

[15] S.F. Altschul et al. Basic local alignment search tool. *J. of Mo. Biol.*, 1990.

[16] S.F. Altschul et al. Gapped blast and psi-blast: A new generation of protein database search programs. *Nucleic Acids Research*, 1997.

[17] M. Gerstein and M. Levitt. Using iterative dynamic programming to obtain accurate pair-wise and multiple alignments of protein structures. In *Proc. Fourth Int. Conf. on Intell. Sys. for Mol. Biol*, pages 59–67, Menlo Park, CA, 1997. AAAI Press.

[18] H.M. Grindley, P.J. Artymiuk, D.W. Rice, and P. Willett. Identification of tertiary resemblence in proteins using a maximal common subgraph isomorphism algorithm. *J. of Mol. Biol.*, 229(3):707–721, 1993.

[19] S. Henikoff and J. G. Henikoff. Amino acid substitution matrices from protein blocks. In *Proc. Natl. Acad. Sci*, volume 89, pages 10915–10919, USA, 1992.

[20] K. Hofmann and P. Bucher. The fha domain: A putative nuclear signalling domain found in protein kinases and transcription factors. *Trends in Biochemical Science*, 20:347–349, 1995.

[21] L. Holm and C. Sander. Protein structure comparison by alignment of distance matrices. *J. Mol. Biol.*, 233:123–138, 1993.

[22] Liisa Holm and Jong Park. Dalilite workbench of protein structure comparison. *Bioinformatics*, 16(6):566–567, 2000.

[23] I. T. Jolliffe. *Principal Component Analysis*. Springer-Verlag, New York, NY, 1986.

[24] I. Jonassen, I. Eidhammer, D. Conklin, and W. Taylor. Structure motif discovery and mining the pdb. In *German Conference on Bioinformatics*, 2000.

[25] I. Koch, T. Lengauer, and E. Wanke. An algorithm for finding maximal common subtopologies in a set of protein structures. *J. of Comp. Biol.*, 3(2):289–306, 1996.

[26] M Kuramochi and G. Karypis. Frequent subgraph discovery. In *IEEE International Conference on Data Mining*, Nov 2001.

[27] M Kuramochi and G. Karypis. Discovering frequent geometric subgraphs. In *IEEE International Conference on Data Mining*, Dec 2002.

[28] Y. Lamdan and H. Wolfson. Geometric hashing: a general and efficient model-based recognition scheme. In *ICCV*, 1988.

[29] H. Li and S. Parthasarathy. Automatically deriving multilevel protein structures through data mining. In *HiPC Conference Workshop on Bioinformatics and Computational Biology*, Hyderabad, India, 2001.

[30] R. Machiraju, S. Parthasarathy, D. S. Thompson, J. Wikins, B. Gatlin, T. S. Choy, D. Richie, M. Jiang, S. Mehta, M. Coatney, S. Barr, and K. Hazzard. Mining Complex Evolutionary Phenomena. In H. Kargupta *et al.*, editor, *Data Mining for Scientific and Engineering Applications*. MIT Press, 2003.

[31] J MacQueen. Some methods for classification and analysis of multivariate observation. In L.M. Le Cam and J. Neyman, editors, *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 281–297, Berkeley, 1967. University of California Press.

[32] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller. Equation of state calculations by fast computing machines. *J. Chem. Phys.*, 21:1087–1092, 1953.

[33] E.M. Mitchell, P.J. Artymiuk, D.W. Rice, and P. Willett. Use of techniques derived from graph theory to compare secondary structure motifs in proteins. *J. Mol. Biol.*, 212:151–166, 1990.

[34] A. G. Murzin, S. E. Brenner, T. Hubbard, and C. Chothia. Scop: a structural classification of proteins database for the investigation of sequences and structures. *J. Mol. Biol*, 247:536–540, 1995.

[35] Raymond T. Ng, Laks V. S. Lakshmanan, Jiawei Han, and Alex Pang. Exploratory mining and pruning optimizations of constrained associations rules. In *SIGMOD*, pages 13–24, 1998.

[36] U.S. Department of Energy. Human genome program. *Genomics and Its Impact on Science and Society: A 2003 Primer*, 2003.

[37] S. Parthasarathy and M. Coatney. Efficient discovery of common substructures in macromolecules. In *IEEE International Conference on Data Mining*, 2002.

[38] J. S. Richardson and D. C. Richardson. Principles and patterns of protein conformation. In G. D. Fasman, editor, *Prediction of protein structure and the principles of protein conformation*, pages 43–75. Plenum, 1989.

[39] A. P. Singh and D. L. Brutlag. Hierarchical protein structure superposition using both secondary structure and atomic representations. In *Proc. Fifth Int. Conf. on Intell. Sys. for Mol. Biol*, pages 284–293, Menlo Park, CA, 1997. AAAI Press.

[40] R. Srikant and R. Agrawal. Mining generalized association rules. In *21st VLDB Conf.*, 1995.

[41] Alexander S. Szalay, Peter Z. Kunszt, Ani Thakar, Jim Gray, Don Slutz, and Robert J. Brunner. Designing and mining multi-terabyte astronomy archives: the Sloan Digital Sky Survey. In *Proc. ACM SIGMOD*, pages 451–462. ACM Press, 2000.

[42] M. S. Venkatarajan and W. Braun. New quantitative descriptors of ami no acids based on multidimensional scaling of large number of physical-chemical properties. *Journal of Molecular Modeling*, 7:445–453, 2001.

[43] X. Wang, J.T.L. Wang, D. Shasha, B.A. Shapiro, I. Rigoutsos, and K. Zhang. Finding patterns in three-dimensional graphs: Algorithms and applications to scientific data mining. *IEEE Transactions on Knowledge and Data Engineering*, 14(4):731–749, jul/aug 2002.

[44] H. Wolfson and I. Rigoutsos. Geometric hashing: An overview. In *In IEEE Computational Science and Engineering*, Oct 1997.

[45] X. Yan and J. Han. gspan: Graph based substructure pattern mining. In *IEEE International Conference on Data Mining*, Dec 2002.

[46] X. Zheng and T. Chan. Chemical genomics: A systematic approach in biological research and drug discovery. *Current Issues in Molecular Biology*, 2002.

A Novel Approach for Prediction of Protein Subcellular Localization from Sequence Using Fourier Analysis and Support Vector Machines

Zhengdeng Lei
Department of Bioengineering
University of Illinois at Chicago
851 S. Morgan Street
Chicago, IL 60607, USA
zlei2@uic.edu

Yang Dai*
Department of Bioengineering
University of Illinois at Chicago
851 S. Morgan Street
Chicago, IL 60607, USA
yangdai@uic.edu

ABSTRACT

A novel method is presented for the prediction of protein subcellular localization from sequence using Fourier analysis and support vector machines. To extract the features of a protein sequence, each amino acid is replaced by a value representing its scale of hydrophobicity and then a fast Fourier transform is applied to the numerically encoded sequence. The transformed sequence data are then used as the input for the training of support vector machines to predict subcellular localization. The motivation for this method of encoding resides fundamentally on (1) the fact that periodicities are critically important factors in protein structure and (2) the ability of this method to capture information about long-range correlations and global symmetries which are completely missed by approaches based on global amino acid composition. Our method is evaluated against the integrated system PSORT-B for the prediction of subcellular localizations of proteins in Gram-negative bacteria. It is demonstrated that the new method outperforms PSORT-B in prediction for the inner membrane, the outer membrane, and extra cellular localizations in a 5-fold cross-validation. It is expected that integrated systems such as PSORT-B may benefit from inclusion of the advanced individual predictor presented in this paper.

Keywords

Protein Subcellular Localization, Gram-negative bacteria, Fourier Transform, Support Vector Machine.

1. INTRODUCTION

Advances in proteomics and genome sequencing are generating enormous numbers of genes and proteins. The development of automated systems for the annotation of protein structure and function has become extremely important. Since many cellular functions are compartmentalized in specific regions of the cell, subcellular localization of a protein is biologically highlighted as a key element in understanding its function. Specific knowledge of subcellular localization can inform and direct further experimental studies of proteins.

Several methods and systems have been developed during the last decade for the predictive task of protein localization. Machine learning methods such as Artificial Neural Networks, the k-nearest neighbor method, and Support Vector Machines (SVM) have been utilized in conjunction with various modalities of feature extraction from protein sequences. Most of the early approaches employed the amino acid composition and the di-peptide frequency [7; 13; 26] to represent sequences. This method may miss the information on sequence order and the inter-relationships between the amino acids. In order to overcome this shortcoming, it has been shown that motifs, frequent-subsequences, and functional domains, which are obtained from various databases (SMART, InterPro, PROSITE) or extracted using Hidden Markov Models and data mining techniques, can be used for the representation of protein sequences for the prediction of subcellular localizations [2; 3; 6; 28; 29]. Methods have also been developed based on the use of the N-terminal sorting signals [1; 5; 10; 20; 22; 23; 24] and sequence homology searching [21].

It has become clear that no single method of prediction can achieve high predictive accuracy for all localizations. Therefore, most robust methods adopt an integrative approach by combining several methods, each of which may be a suitable predictor for a specific localization or a generic predictor for all localizations. PSORT is an example of such a successful system. Developed by Nakai and Kanehisa [23], PSORT, recently upgraded to PSORT II [12; 22], is an expert system that can distinguish between different subcellular localizations in eukaryotic cells. It also has a dedicated subsystem PSORT-B for bacterial sequences [9]. Obviously, further improvement of the quality of such an integrated system relies on advances in the individual predictors, namely, improvements that arise from the employ of sophisticated protein encoding schemes and powerful machine learning and data mining techniques.

In this study, we describe a new approach for the prediction of protein subcellular localization from protein sequences using Fourier analysis as the feature extracting tool and sup-

*Corresponding author.

port vector machines as the learning framework. In order to extract the features from a given protein sequence, each amino acid is replaced by a value representing its scale of hydrophobicity and a fast Fourier transform is subsequently applied to the numerically encoded sequence. These transformed data are then trained by support vector machines.

Fourier analysis has been used for (1) the recognition of protein folds [27] and gene-encoding regions of DNA sequences [8; 30] and (2) the detection of periodic patterns and tandem repeats of residues in both DNA and protein sequences [25]. The motivation for this method of encoding resides fundamentally on the observation that periodicities are critically important factors in protein structure [27]. The approach based on the Fourier transform analysis is capable of capturing information about long-range correlations and global symmetries; both are completely missed by approaches based on global amino acid composition. For comparison, we also present another encoding method based on the tri-peptide frequency. This encoding scheme is an extension of the method using the amino acid decomposition and has been used for the prediction of protein folds [18].

Our method is evaluated against PSORT-B for the prediction of subcellular localizations for Gram-negative bacteria [9]. It is demonstrated by the result of a 5-fold cross-validation that the new method outperforms PSORT-B predictions associated with the outer membrane, the inner membrane, and extra cellular localizations. It is expected that PSORT-B may benefit from the integration of this new predictor into the system.

2. METHOD

This section introduces two sequence encoding methods. One is the encoding method based on the Fourier analysis of protein sequences; the other is based on the tri-peptide frequency. The latter approach has been used in protein fold recognition [18], but has never been evaluated for the prediction of subcellular localizations. We also present a short description of support vector machines, the machine learning method used in this study.

2.1 Feature Extraction based on the Fourier Transform

There are many ways to describe amino acids, most of which are correlated to some degree. For example, the AAindex database contains indices representing 434 different physicochemical and biological properties of amino acids [16]. We concentrate on the amino-acid hydrophobicity in this work, as it is the one of major properties influencing the structure and function of a protein [14]. A simple three-state hydrophobicity scale is used to map hydrophobic residues to 1, hydrophilic residues to -1, and "neutral" residues to 0 [27]. More precisely,

$$(A, C, F, I, L, M, V) \rightarrow 1,$$

$$(D, E, H, K, N, Q, R) \rightarrow -1,$$

and

$$(G, P, S, T, W, Y) \rightarrow 0.$$

Once a protein sequence has been encoded into the above numerical format, it is converted to a sequence in the frequency domain with a Fourier transform. A common use of the Fourier transform is the identification of frequency components of a weak time-dependent signal buried in noise. Prior to the application of the Fourier transform, the numerical sequences have to be lengthened by padding with zeros, since the length of the input sequences is required to be a power of two. Let $n = 2^M$ denote the smallest number that is greater than or equal to the length of the longest protein sequence in a given set, where M is some integer. Let $\{x(1), ..., x(n)\}$ be the numerically encoded sequence of a protein according to the three-state hydrophobicity scale after padding. The Fast Fourier Transform (FFT) will transform the encoded sequence into another sequence $\{X(1), ..., X(n)\}$ in the frequency domain. The procedure of the FFT used in this research is based on the algorithm of Masters [19], which is an implementation of the discrete Fourier transform (DFT) given by

$$X(f) = \sum_{t=1}^{n} x(t) \exp[i(2\pi t f/n)] \quad (f = 1, ..., n),$$

and

$$x(t) = \frac{1}{n} \sum_{f=1}^{n} X(f) \exp[-i(2\pi t f/n)] \quad (t = 1, ..., n).$$

Figures 1-3 present the encoded sequences before and after the application of the FFT for two representative proteins from extra cellular, inner membrane, and outer membrane localizations, respectively. The sequences obtained from the FFT display enhanced characteristics for each localization in comparison with the sequences before the use of the FFT.

Another advantage of the FFT based feature extraction is that the number of extracted features is almost the same as the length of the longest protein sequence in the data. This is a compact representation for protein sequences in contrast to the features extracted based on the tri-peptide frequency described below.

2.2 Feature Extraction based on the Tri-peptide Frequency

In order to evaluate the FFT encoding method presented above, an approach based on the tri-peptide frequency for feature extraction has also been considered. This encoding method extends the concept of the amino acid composition and di-peptide frequency encoding methods. These have been used intensively for the representation of protein sequences in numerous applications. These are, for example, the prediction of (1) protein secondary structures, (2) protein folds, and (3) subcellular localizations, and the efficacy of these encoding methods has been established.

In order to encode a protein sequence with the tri-peptide frequency, a vector of $21^3 = 9261$ dimensions is required. Each entry of the vector is associated with a possible pattern of three amino acids. Since the symbol "X" may appear in some sequences, it is added to the set of the original 20 symbols of the amino acids to give a total of 21. A window with a length of three is moved along the sequence from the first amino acid to the third amino acid from the end. Every 3-letter pattern that appears in the window is recorded with increments of 1 in the corresponding entry of the vector. Upon the termination of this procedure, the vector provides the tri-peptide frequency of the sequence.

The final vector is normalized by dividing the number of window positions associated with that sequence. Note that the resulting vector is sparse, as only a small collection of the possible 3-letter patterns will appear in each protein sequence.

2.3 Support Vector Machine

Suppose that we are given a set of m points $\boldsymbol{x}_i$ $(1 \leq i \leq m)$ in an n-dimensional space. Each point $\boldsymbol{x}_i$ is labeled by $y_i \in \{1, -1\}$ denoting the membership of the point. An SVM is a learning method for binary classification. Using a nonlinear transformation ϕ, it maps the data to a high dimensional feature space in which a linear classification is performed. It is equivalent to solving the quadratic optimization problem:

$$\begin{array}{ll} \min\limits_{\boldsymbol{w},b,\xi_1,\ldots,\xi_m} & \frac{1}{2}\boldsymbol{w}\cdot\boldsymbol{w} + C\sum\limits_{i=1}^{m}\xi_i \\ \text{subject to} & y_i(\phi(\boldsymbol{x}_i)\cdot\boldsymbol{w}+b) \geq 1-\xi_i \quad (i=1,...,m), \\ & \xi_i \geq 0 \quad (i=1,...,m), \end{array}$$

where C is a parameter. The decision function is defined as $f(\boldsymbol{x}) = \phi(\boldsymbol{x}_i)\cdot\boldsymbol{w}+b$, where $\boldsymbol{w} = \sum_{i=1}^{m}\alpha_i\phi(\boldsymbol{x}_i)$ and α_i $(i = 1, ..., m)$ are nonnegative constants determined by the dual problem of the optimization defined above. Therefore, the function is

$$f(\boldsymbol{x}) = \sum_{i=1}^{m}\alpha_i\phi(\boldsymbol{x}_i)\cdot\phi(\boldsymbol{x}) + b = \sum_{i=1}^{m}\alpha_i K(\boldsymbol{x}_i, \boldsymbol{x}) + b$$

through the definition of the appropriate kernel function K. For details of SVMs refer to Cristianini and Shawe-Taylor [4].

3. RESULTS AND DISCUSSION

We employed the SVMs in conjunction with the features extracted by the methods described above for training and testing. The evaluation of the methods was conducted on the following dataset.

3.1 Dataset

The set of proteins from Gram-negative bacteria used in the evaluation of PSORT-B [9] was considered (available at `http://www.psort.org/`) in this experiment. It consists of 1443 proteins with experimentally determined localizations. The dataset comprises 1302 proteins resident at a single localization site: 248 cytoplasmic, 268 inner membrane, 244 periplasmic, 352 outer membrane, and 190 extra cellular; it additionally contains a set of 141 proteins resident at multiple localization sites: 14 cytoplasmic/inner membrane, 50 inner membrane/periplasmic, and 77 outer membrane/extracellular. In our experiment, we considered only the 1302 proteins possessing a single localization. The longest protein sequence in this dataset is about 4000 amino acids, so the length of the final FFT encoded sequences is approximately, 2000.

3.2 Experiment and Results

We have compared the performance of our new methods with that of PSORT-B, a powerful tool for the prediction of protein subcellular localization for Gram-negative bacteria.

The system PSORT-B was designed to seek precision other than recall to allow for confident predictions, and prevents the propagation of erroneous predictions. It utilizes six modules for the generation of an overall prediction of a localization site:

(1) BLAST search based predictor *SCL-BLAST* for all localizations [21];

(2) Motif based predictor *Motif* for all localization sites [23];

(3) Hidden Markov Model based predictor *HMMTOP* for the inner membrane localization [28; 29];

(4) Motif based predictor *OPT Motif* for the outer membrane localization [9];

(5) Amino acid composition based predictor *SubLocC* for the cyctoplasmic localization [13];

(6) Signal peptide based predictor *Signal peptides* for the non-cyctoplasmic localization [9; 24].

Based on the output from each module, the system uses a Bayesian network to generate a final probability value for each localization site. The system achieved an overall prediction accuracy of 75% for all localizations, a significant improvement over the previous results of PSORT I.

Besides the tri-peptide and the FFT based methods, we also implemented the method based on the amino acid composition. The experiment was carried out using a 5-fold cross-validation for each specific localization. Each time, the relevant dataset consisting of the proteins with the specific localizations was designated as the positive set; the remainder of the proteins was denoted as the negative set. The radial basis function was chosen as the kernel function for the SVM, since a preliminary experiment indicated this kernel exhibited better performance.

As the sizes of the positive and negative sets are substantially different, the performance of SVM was evaluated for precision (or sensitivity):

$$\text{precision} = \frac{tp}{tp+fp},$$

and recall (or positive prediction value):

$$\text{recall} = \frac{tp}{tp+fn},$$

where tp (resp. tn) is the number of the predicted positive (resp. negative) proteins which are true positive (resp. negative), and fp (resp. fn) is the number of the predicted positive (resp. negative) proteins which are true negative (resp. positive). The precision and recall of the 5-fold cross-validation were computed as the averages of the values from 5 folds.

The generalization performance of an SVM is controlled by the following parameters:

(1) the trade-off C between the training error and the class separation;

(2) the parameter g in the radial basis function, i.e., $\exp(-g\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2)$;

(3) the biased penalty J for error from positive and negative training points.

Table 1: Results obtained from four different methods for the proteins from Germ-negative bacteria. (The numbers represent percentages.

Method	Composition		tri-peptide		FFT		PSORT-B	
Localization	Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall
Cytoplasmic	83.38	69.22	83.43	50.53	61.20	68.00	97.6	69.4
Inner membrane	98.65	83.57	99.52	80.75	96.12	87.30	96.7	78.7
Periplasmic	91.36	54.56	90.37	50.34	50.00	54.20	91.9	57.6
Outer membrane	87.21	84.12	95.28	83.66	95.70	94.30	98.8	90.3
Extra cellular	88.38	53.68	92.57	50.53	92.10	80.70	94.4	70.0

Composition : the method using SVM with the features from the amino acid composition ;
tri-peptide : the method using SVM with the features from the tri-peptide frequency;
FFT : the method using SVM with the features from the FFT of hydrophobicity encoding;
PSORT-B : the integrated predictor in [9]. The results are from Gardy *et al.* [9].

The values of precision and recall of a 5-fold cross-validation were computed for each triplet (C, g, J). The choices of the parameters in the experiment for the composition and tri-peptide encoding sequences are given as follows:

C: from 1 to 150 with an incremental size of 10;

g: 1 to 100 with an incremental size of 10;

J: from 0.1 to 3.0 with an incremental size of 0.2.

The FFT encoded sequences are dense, therefore, they demand an intensive training time. Accordingly, a search over the full range of parameters would be prohibited. In order to deal with this problem, a two-step strategy for searching was employed. In the first round, the procedure scanned through all triplets (C, g, J) determined as follows.

C: from 2^{-8} to 2^{7} with $c = 2 * c$ for each step;

g: from 2^{-8} to 2^{7} with $g = 2 * g$ for each step;

J: from 0.1 to 3.0 with $j = j + 0.2$ for each step.

After identifying the best g value g^* from the first round, a more intensive search localized around g^* was performed. More precisely, it searched all triplets determined as follows.

C: from 1 to 21 with $C = C + 3$ for each step;

g: from $2^{g^*} - 1$ to $2^{g^*} + 1$ with $g = g + 0.003$ for each step;

J: from 0.1 to 3.0 with $J = J + 0.2$ for each step.

The SVMLight package was used as the SVM solver [15]. The best values of precision and recall for each method are given in Table 1, where the results for PSORT-B are taken from Gardy *et al.* [9]. Note that we compare the performance of the single predictor against the integrated predictive results from PSORT-B.

The FFT based method demonstrated superior performance over that of PSORT-B for the prediction of all three localizations: the inner membrane, the outer membrane, and the extra cellular case. While maintaining similar levels of precision, the improvement on the corresponding recall is from 78.7 to 87.3 for the inner membrane localization, from 90.3 to 94.3 for the outer membrane localization, and from 70.0 to 80.7 for the extra cellular localization. The FFT based method achieved substantial improvement in recall for the inner membrane and extra cellular localizations as compared with the remaining three methods. However, the FFT based approach provided inferior findings for the cytoplasmic and periplasmic localizations.

On the other hand, the tri-peptide based method demonstrated good predictive power for the inner membrane localization as compared with PSORT-B. However, its ability for the other localizations did not surpass that of PSORT-B. Notably, the prediction of the periplasmic localization seems to be the hardest for all methods.

Although the FFT encoding method generates a compact set of features, we experienced longer times for training and testing in comparison with the tri-peptide encoding method, even though the tri-peptide frequency approach has a significantly larger number of features. We propose the following interpretation of this behavior. The FFT encoded sequences have a full dense structure while the tri-peptide encoded sequences are very sparse, although the lengths are longer. A feature selection scheme using a cut-off value to discard lower frequency features in the FFT encoded sequences may be able to achieve a similar level of predictive quality.

4. CONCLUSIONS

This work has introduced a novel Fast Fourier Transform based method for the feature extraction of protein sequences in conjunction with the use of support vector machines for the prediction of subcellular localizations. In addition, a tri-peptide based encoding method was considered in parallel.

The performances of these methods were empirically evaluated on a set of proteins with experimentally determined localizations from Germ-negative bacteria. Compared with the integrated system PSORT-B, the experimental results demonstrated that the SVM learned from the FFT encoded sequences exhibited superior performance for the prediction of the inner membrane, the outer membrane, and the extra cellular localizations, but was inferior for the prediction of cytoplasmic and periplasmic localizations. This implies that the hydrophobicity alone can not properly represent the sequence information which characterizes these two localizations. Combination with the tri-peptide based method may improve the predictive performance. This can be realized by using a kernel that combines the information from the FFT encoded sequences and the tri-peptide encoded sequences. The use of a different hydrophobicity index of amino acids, for example, the index shown in Table 2 [17], may also improve the quality of prediction.

Table 2: Hydrophobicity index of amino acids in Kyte and Doolittle.

amino acid	I	V	L	F	C	M	A	Z	T	S
index	4.5	4.2	3.8	2.8	2.5	1.9	1.8	-0.4	-0.7	-0.8
amino acid	W	Y	P	H	E	Q	D	N	K	R
index	-0.9	-1.3	-1.6	-3.2	-3.5	-3.5	-3.5	-3.5	-3.9	-4.5

5. ACKNOWLEDGMENTS

This research is partially supported by National Science Foundation (EIA-022-0301) and Naval Research Laboratory (N00173-03-1-G016).

6. REFERENCES

[1] Bannai, H., Tamada, Y., Maruyama, O., Nakai, K. and Miyano, S. (2002) Extensive feature detection of N-terminal protein sorting signals. *Bioinformatics*, 18, 298-305.

[2] Cai, Y.D. and Chou, K. C. (2003) Predicting subcellular localization of proteins in a hybridization space. *Bioinformatics*. 20, 1151-1156.

[3] Chou, K.C. and Cai, Y.D. (2002) Using functional domain composition and support vector machines for prediction of protein subcellular location. *J. Biol. Chem.*, 277, 45765-4576.

[4] Cristianini, N. and Shawe-Taylor, J. (2000) *An Introduction to Support Vector Machines*, Cambridge University Press.

[5] Emanuelsson, O., Nielsen, H., Brunak, S. and von Heijne, G. (2000) Predicting subcellular localization of proteins based on their N-terminal amino acid sequence. *J. Mol. Biol.*, 300, 1005-1016.

[6] Emanuelsson, O. (2002) Predicting protein subcellular localisation from amino acid sequence information. *Brief. Bioinform.*, 3, 361-376.

[7] Feng, Z.P. (2001) Prediction of the subcellular location of prokaryotic proteins based on a new representation of the amino acid composition. *Biopolymers*, 58, 491 - 499.

[8] Fickett, J.W. and Tung, C.S. (1992) Assessment of protein coding measures. *Nucleic Acids Res.*, 20, 6441-50.

[9] Gardy, J.L. et al. (2003) PSORT-B: improving protein subcellular localization prediction for Gram-negative bacteria. *Nucleic Acids Res.*, 31, 3613-3617.

[10] von Heijne, G. (1994) Signals for protein targeting into and across membranes. *Subcell. Biochem.*, 22, 1-19.

[11] Hofmann, K., Bucher, P., Falquet, L. and Bairoch, A. (1999) The PROSITE database, its status in 1999. *Nucleic Acids Res.*, 27, 215-219.

[12] Horton, P. and Nakai, K. (1999) PSORT: a program for detecting sorting signals in proteins and predicting their subcellular localization. *Trends Biochem. Sci.*, 24, 34-36.

[13] Hua, S. and Sun, Z. (2001) Support vector machine approach for protein subcellular localization prediction. *Bioinformatics*, 17, 721-728.

[14] Irbäck,A. and Sandelin,E. (2000) On hydrophobic correlations in protein chains. *Biophysical Journal*, 79, 2252-2258.

[15] Joachims, T. (1999) *Making Large Scale SVM Learning Practical. Advances in Kernel Methods-Support vector learning*. MIT Press, Cambridge.

[16] Kawashima, S. and Kanehisa, M. (2000) AAindex: amino acid index database. *Nucleic Acids Res.*, 28, 374.

[17] Kyte, J. and Doolittle, R.F. (1982) A simple method for displaying the hydrophobic character of a protein. *J. Mol. Biol.*, 157, 105.

[18] Leslie, C., Eskin, E., Cohen, A., Weston, J. and Noble, W. (2002) Mismatch String Kernels for Discriminative Protein Classification. [Journal version of NIPS 2002 paper.] To appear in Bioinformatics.

[19] Master, T. (1994) *Signal and Image Processing with Neural Networks : a C++ Sourcebook*. Bew York : John Wiley & Sons.

[20] Menne, K.M.L., Hermjakob, H. and Apweiler, R. (2000) A comparison of signal sequence prediction methods using a test set of signal peptides. *Bioinformatics*, 16, 741-742.

[21] Nair, R. and Rost, B. (2002) Sequence conserved for subcellular localization. *Protein Sci.*, 11, 2836-2847.

[22] Nakai, K. (2000) Protein sorting signals and prediction of subcellular localization. *Adv. Protein. Chem.*, 54, 277-344.

[23] Nakai, K. and Kanehisa, M. (1991) Expert system for predicting protein localization sites in Gram-negative bacteria. *Proteins*, 11, 95-110.

[24] Nielsen, H., Engelbrecht, J., Brunak, S. and von Heijne, G. (1997) A neural network method for identification of prokaryotic and eukaryotic signal peptides and prediction of their cleavage sites. *Int. J. Neural Syst.*, 8, 581-599.

[25] Pasquier, C.M., Promponas, V.I., Varvayannis, N.J. and Hamodrakas,S.J. (1998) A web server to locate periodicities in a sequence *Bioinformatics*, 14, 749-704.

[26] Reinhardt, A. and Hubbard, T. (1998) Using neural networks for prediction of the subcellular location of proteins. *Nucleic Acids Res.*, 26, 2230-2236.

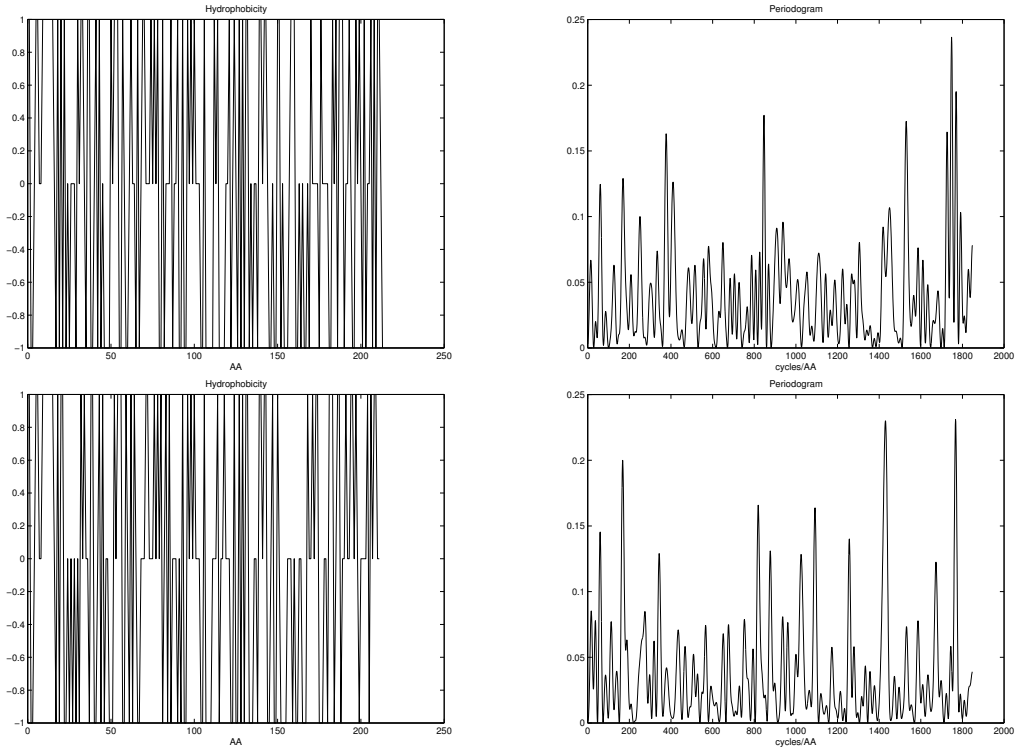


Figure 1: The encoded sequences for two extra cellular proteins before (left) and after (right) the fast Fourier transform.

[27] Shepherd, A.J., Gorse, D. and Thornton, J.M. (2003) A novel approach to the recognition of protein architecture from sequence using Fourier analysis and neural networks. *PROTEINS: Structure, Function, and Genetics*, 50, 290-302.

[28] Tusnady, G.E. and Simon, I. (1998) Principles governing amino acid composition of integral membrane proteins: application to topology prediction. *J. Mol. Biol.*, 283, 489-506.

[29] Tusnady, G.E. and Simon, I. (2001) The HMMTOP transmembrane topology prediction server. *Bioinformatics*, 17, 849-850.

[30] Yan, M., Lin, Z.S. and Zhang, C.T. (1988) A new Fourier transform approach for protein coding measure based on the format of the Z curve. *Bioinformatics*, 14, 685-690.

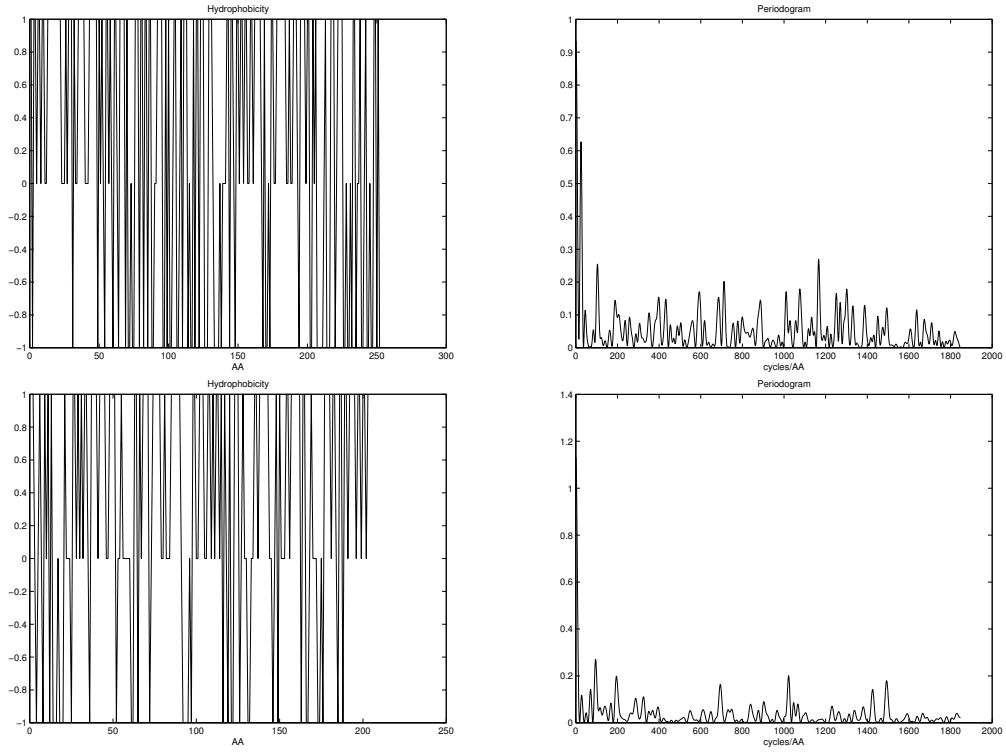


Figure 2: The encoded sequences for two inner membrane proteins before (left) and after (right) the fast Fourier transform.

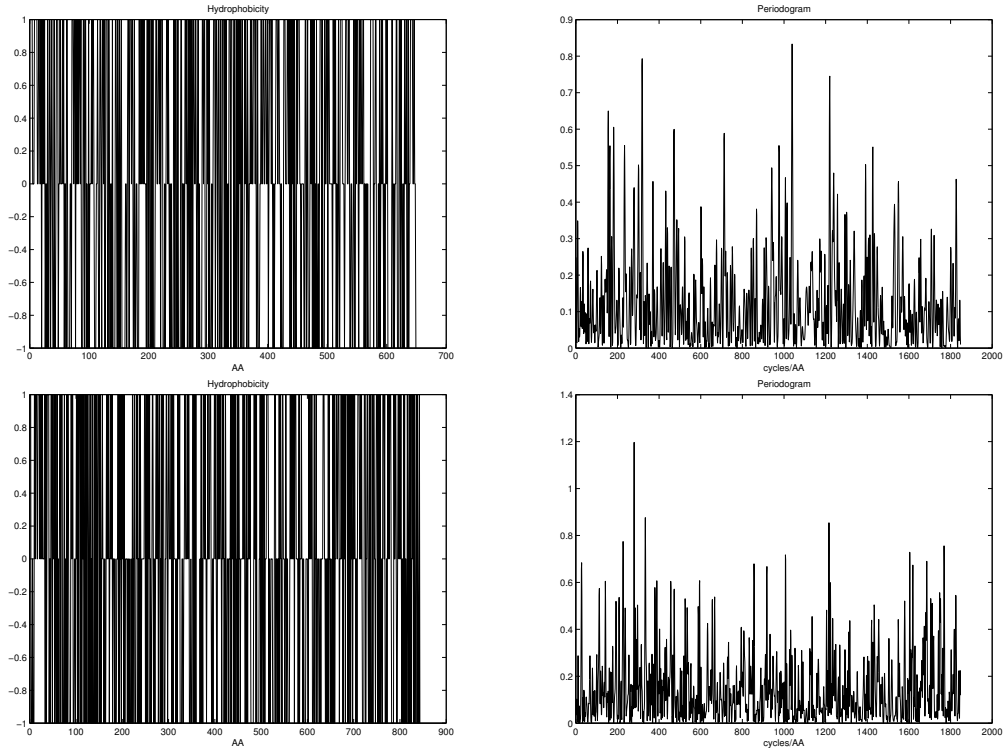


Figure 3: The encoded sequences for two outer membrane proteins before (left) and after (right) the fast Fourier transform.

High-throughput Protein Interactome Data: Minable or Not

Jake Y. Chen
Indiana University School of Informatics
Purdue School of Science Department of Computer and Information Science
Indianapolis, IN 46202
jakechen@iupui.edu

Andrey Y. Sivachenko
Prolexys Pharmaceuticals, Inc.
2150 W. Dauntless Ave
Salt Lake City, UT 84116
asivache@prolexys.com

Lang Li
Division of Biostatistics
Indiana University School of Medicine
Indianapolis, IN 46202
lali@iupui.edu

ABSTRACT

There is an emerging trend in post-genome biology to study the collection of thousands of protein interaction pairs (protein interactome) derived from high-throughput experiments. However, high-throughput protein interactome data, especially when derived from the Yeast 2-Hybrid (Y2H) method, have been generally believed to be irreproducible and unreliable, with an estimated high "noise ratio" of more than 50%. In this work, we performed a comprehensive study on approximately 70,000 protein interactions derived from a systematic yeast 2-hybrid (SY2H) method. We performed a comprehensive analysis of biases, reproducibility, statistical significance, and biologically significant patterns in this data set. Surprisingly, we found these protein interactions have a much higher quality. The data represented a comprehensive survey of the entire human proteome with no chromosomal location bias. The reproducibility rate of interactions among replicated searches was quite good, i.e., at 78.5%. The false positive rate, 5.5e-5, was two orders of magnitude better than that reported elsewhere. We further developed several statistical measures and concluded that a protein interaction only needs to appear in two different SY2H searches to become significant. We also developed techniques to show supporting evidence that "promiscuous" protein interactions were not random noises; instead, they could be "network hubs" of the cell signaling network. We also attributed the low noise in our data to the adoption of standard control in the experimental data generation process.

Keywords

Protein Interaction, Systematic Yeast 2-Hybrid, Reproducibility, Significance, Data Mining.

1.INTRODUCTION

In post-genome systems biology, the study of **protein interactomes**—comprehensive collections of all the expressed proteins and their interactions within cells of model organisms, has gained increasing popularity. Several protein interactome mapping projects, including those of *H. pylori* [1], *S. cerevisiae* [2, 3], *D. melanogaster* [4], *H. sapiens* [5], and *C. elegans* [6], have reported significant progress in recent years. In these projects, novel high-throughput experimental techniques, e.g., high-throughput yeast 2-hybrid (Y2H) screenings [7], protein arrays, and mass spectrometry, have been developed to measure physical bindings between proteins in parallel. This results in a steady influx of protein interaction data in the public domain. By understanding how proteins regulate each other through interaction, biologists can compile novel molecular pathway models, which they cannot normally derive from genomics techniques. The collection of thousands of protein interactions also enable system biologists to understand protein functions in a molecular network context, through which they may identify protein biomarkers or drug targets for diagnosing and treating human genetic diseases [8].

Nonetheless, there is a prevalent belief among many researchers that experimental protein-protein data generated from the high-throughput Y2H method equate to "high errors" and "poor reproducibility". Much doubt about Y2H data might have originated from a comparative analysis by Mrowka *et al* [9], who suggested that high-throughput Y2H experiments may have a false positive rate of greater than 50%. In a similar study, Bader *et al* analyzed high-throughput protein interaction data obtained from several sources and also concluded that these methods do not show enough internal consistency to warrant complete acceptance of the result [10]. Even more grim opinions exist [11]. Whether perceived or real, the high data "noise" has presented immense challenges for computational scientists to "mine" for biologically significant protein interactions and for biologists to trust data mining results from these efforts. Therefore, an imminent question for any researcher who will study the protein interactome data becomes,

(1) Can I trust the high-throughput protein interactome data at all?

(2) If so, how do I mine for significant protein interactions?

In this work, we restore confidence in high-throughput protein interactome data and the mining efforts, by investigating the biases, reproducibility, statistical significance, and functionally significant patterns of a human protein interactome data set. This data set consists of approximately 7,500 human proteins and 70,000 protein interactions, which was generated from a high-throughput **Systematic Yeast 2-Hybrid Method** (**SY2H**, refer to the Method section) [5]. Some of us have been curating and applying this data to biological discoveries for two years. We will show that by using a systematic method (SY2H), in which experimental conditions are enforced by standard protocols and the same robots, one can achieve reasonably good data reproducibility, keep false positive rate low, design reliable statistical hypothesis tests, discover statistically significant "interaction network hub proteins", and identify biologically significant interacting protein groups. We also show that "promiscuous" protein interactions should perhaps be regarded as "network hubs" instead of random noises—another explanation for the discrepancy between our analysis and the widely-held beliefs elsewhere. Our results may restore the confidence in similar high-throughput protein interactome data sets, and promote their application in subsequent molecular function studies. In Table 1, we have summarized some key features of the SY2H method by comparing it with the standard Y2H method. For a detailed description of this method, refer to the next section.

Table 1. A summary of comparisons between two Yeast 2-Hybrid (Y2H) methods. Refer to the Method section for an explanation of "baits", "preys", "searches", and "positives".

	Standard Y2H	Systematic Y2H
Bait Known Prior to a Search	Yes	No
Bait Sequence Enlisted in a Search	Whole or partial sequences	Short sequence fragments
Bait/Interaction Selection Bias	Yes (by design)	No (random sampling)
Possible Replicated Preys in a Search	Yes	Yes
Possible Replicated Same-bait Searches	No	Yes
Sequences to be Identified from Positives	Prey only	Bait and Prey
Global Assessment of Interactions	No	Possible

2.METHODS

Systematic Yeast 2-Hybrid (SY2H). First, two Y2H cDNA libraries from cDNA library samples from an organism are prepared using random internal primers. The hybrid proteins, which are derived by fusing a sample cDNA fragment with the yeast transcription factor DNA-binding domain or with the yeast transcription factor activation domain, are called "*bait*" and "*prey*", respectively. Second, haploid yeast bait and prey cDNA libraries are isolated into individual colonies, each containing a single bait or prey. Third, two types of haploid yeast cultures are mixed, one containing single bait colonies and the other containing colonies of the entire prey library, to allow mating to happen. Each such an experiment is called a "*search*". Fourth, mated diploid yeast cultures are placed on dishes that contain selective medium, which allows the mated yeast to grow only if bait and prey interact. Each grown diploid yeast colony is called a "*positive colony*", or a "*positive*". Fifth, up to a certain number of positive colonies is selected for picking ("*picked positives*"). Positives that are not picked are discarded. Sixth, DNA sequences from picked positives are amplified by PCR and DNA sequencing from both the 5' and 3' directions is performed. Seventh and lastly, interacting protein fragments are identified by comparing bait and prey DNA sequence fragments with annotated mRNA sequences from public sequence databases using the BLASTN software program.

Protein Interactome Data Collections. We collected the human protein interactome data from a high-throughput interactome mapping project using the above described SY2H system [5]. There were two major data collection milestones for this project in 2002-3. In the first major milestone, 13,656 unique protein interaction pairs were collected from approximately 50,000 SY2H searches against a prey cDNA library from mRNAs in homogenized human brain. This data set represented proteins from approximately 4,473 human gene loci, or approximately 5,000 unique proteins. In the second and a recent milestone in September 2003, approximately 70,000 unique protein interaction pairs were collected from more than 200,000 searches against a variety of human cDNA libraries. This interaction data set represented approximately 7,500 unique human proteins. We took a series of data snapshots between the milestones to perform our data analysis. The fact that we used slightly different data snapshots for each analysis is not a concern, since all these snapshots represented nearly random "samples" of the same protein interactome—a unique characteristic of the SY2H method (refer to Results).

Bioinformatics Data Analysis. We performed large-scale bioinformatics data analysis tasks to prepare and manage all the protein interaction data using Oracle9i server and genomic data modeling methods described in [12]. We integrated hundreds of gigabytes of biological data from more than 20 different sources. In particular, we integrated all the protein interaction pairs with public REFSEQ, LocusLink, and Gene Ontology annotations [13, 14]. In our data analysis, we used a combination of software tools, including the R statistical package and the Sportfire DecisionSite Browser for statistical data analysis and data visualizations. For this work, we developed several protein interaction data analysis methods, which we would describe along with the discussion of results next.

3.RESULTS

3.1 Comprehensive Protein Coverage and Bias

Due to the unique characteristics of the SY2H method and a homogenized human brain tissue library source, we expect to observe a wide spectrum of expressed proteins (a random sample of the entire "proteome") and interactions between them in the data. In principle, the data should represent a comprehensive survey of the entire human proteome with little sampling bias. In Figure 1, we confirmed this expectation by showing a relative frequency distribution for a snapshot of 5,619 proteins, binned by their chromosomal locations. While the relative distribution of all human REFSEQ proteins varies among different chromosomal and mitochondrial locations, the interacting proteins follow the varying distribution details quite well. Therefore, we can draw two inferences from this analysis. One is that the SY2H method indeed does a good job of randomly sampling the entire human proteome with no bias in coverage. The other is that all human proteins from different chromosomes and mitochondrion (perhaps except for chromosomes 6, 12, X, and Y) seem to share the same tendency to interact with each other.

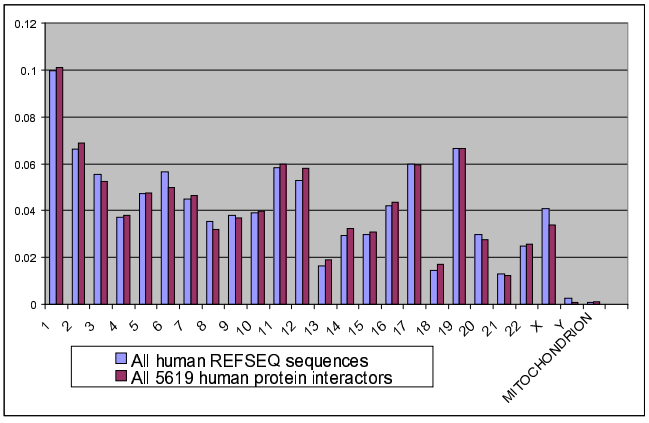


Figure 1. A comparison of the relative frequency distributions between all human REFSEQ sequences and all interacting proteins from the SY2H system, binned by their chromosomal and organelle locations.

Does a comprehensive coverage or a random sampling of the proteome suggest that there should be no bias in whichever proteins may become recruited in interactions? Not at all. If so, proteins of all 3-dimensional shapes would have interacted with each other equally. In Table 2, we showed an example of observed biases based on protein functional categories. Here, we listed eight protein functional categories. In each category, we listed a count of all human proteins from the LocusLink database, a count of interacting proteins identified with the SY2H method, and a percentage of coverage of identified protein for the category. In the last row of the table, we also showed several sums. This data shows that there were 18% of all 33,673 human proteins—a snapshot of 6,213 proteins derived from the SY2H system. However, "protein phosphatases" and "protein kinases" (CLASS I proteins) are highly enriched, at 29% and 26% respectively; "receptor" and "receptor : GPCR" proteins (CLASS II proteins), however, are scarce, at 6% and 5% respectively. We attribute this finding to a possible high **functional bias** towards proteins playing essential functional roles. For example, compared with other proteins, catalytic activities of enzymes (CLASS I proteins) are more frequently modulated by regulatory proteins through protein interactions; therefore, we observed an enrichment of CLASS I proteins. CLASS II proteins are poorly represented perhaps for a different reason--Y2H methods usually cannot capture protein interactions among membrane proteins (most CLASS II proteins).

Table 2. A breakdown of protein counts according to their functional categories.

	All Human Proteins from LocusLink	*Interacting Proteins Identified by SY2H*	*Percentage of Coverage*
Protein phosphatase	240	70	29%
Protein kinase	400	102	26%
Polymerase	161	39	24%
Transcription factor	372	77	21%
Channel protein	339	65	19%
Protease	233	33	14%
Receptor	3,294	203	6%
Receptor: GPCR	705	38	5%
Total	33,673	6,213	18%

3.2 Data Reproducibility

We assessed the reproducibility of interaction data derived from the SY2H system, and found it to be surprisingly good. For **reproducibility**, we refer to the capability of a high-throughput interaction discovery system to identify true interactions consistently. In Figure 2, we show that interaction reproducibility, calculated as the percentage of all interactions that can be replicated across different SY2H searches, is 78.5%. Comparing the our SY2H system with a standard Y2H system, we think it is possible that the reproducibility rate estimated in previous publications (from 10% to 50%) [11] were based collections of high-throughput data generated from different academic labs without the setup of robotic machineries for consistent controls. There is also a lack of report on replicated protein interaction pair data from public sources.

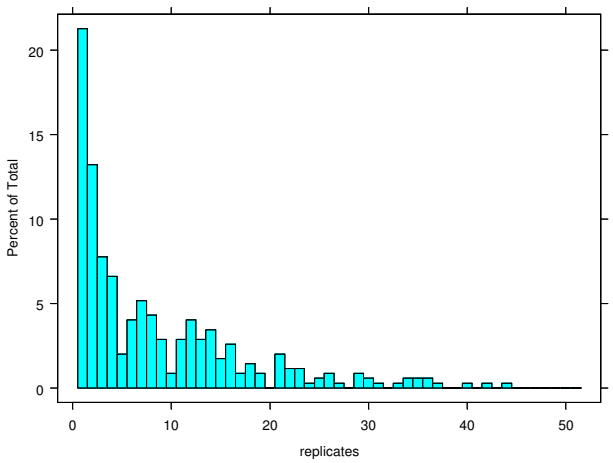


Figure 1. A relative frequency distribution of protein interaction "replicates". A "replicate" bin at x=1 indicates the percentage of interactions (21.5%) that are identified only once. All other "replicate" bins with x > 1 refer to the percentage of true replicated interaction (1-21.5%=78.5%) with an interaction replication count of x. The protein interaction data in this graph come from a random sample of 513 bait proteins, each of which is identified in at least two separate SY2H searches.

This 78.5% may still be an under-estimate of the true data reproducibly level for an SY2H system. This is because identifying protein interactions from searches is also a sampling process, in which the robots often pick a dozen top "positive colonies" for DNA amplification and sequencing. Therefore, one may not have exhaustively identified all replicated protein interactions from replicate searches. In other words, we expect the relative percentage for the "replicates"=1 bin becomes smaller than the 21.5% when the size of data increases.

3.3 Statistical Significance of Interactions

To identify statistically significant protein interactors (as preys) and protein interaction pairs (as bait-prey pairs), we describe a statistical data testing framework. First, we present a **null hypothesis**, in which we presume that the interactions happen randomly among all interactors. Therefore, the rate of interaction discovery, p, can be estimated by the following:

$$p = \frac{I}{N * M}, \qquad (1)$$

Here, I is the total number of observed unique interaction pairs, N is the total number of searches performed, and M is the total number of observed unique preys. To calculate p, for example, using a data snapshot taken from the first milestone (refer to Methods), we have I = 13,660, N = 50,000, M = 5,000, and therefore p = 13,660/50,000/5,000 = 5.5e-5. Similarly, using a recent data snapshot taken from the second milestone, we have I = 70,000, N = 200,000, M = 7,000, and therefore p =

70,000/200,000/7,000 = 5.0*e*-5. The two estimates are very close to each other.

We interpret *p* as an upper-bound estimate of the false positive rate for protein interactions observed in an SY2H system. There are two reasons for us to believe that *p* may be conservatively estimated. First, many of these observed *I* interactions may not be discovered totally by chance (recall functional bias); therefore, *p* should be smaller. We choose to treat interactions as "random" events, also because we do not have sufficient "negative" control interaction data set, i.e., a set of known non-interacting protein pairs. Second, the estimated prey number, *M*, may be higher than we currently used, because a high-throughput SY2H system sometimes may fail to amplify and identify a DNA sequence. The conservative estimate of a *p* at 5.5*e*-5 is good, because we can be confident later that the calculation of *p*-values, which we base on *p*, will be reliable indicators of statistical significances for replicated interactions.

Note, however, much higher false positive rates have been estimated for several standard high-throughput Y2H systems. For example, Gavin *et al* [15] reported a *p=1.07e-3* in their recent study, and Ho et al also reported a *p=1.37e-3* [16]. We attribute the much smaller false positive rate for the SY2H system to the high data reproducibility described in an earlier section.

Next, we describe two **hypothesis test methods**. In both methods, we calculate the *p*-values, one for observing multiple preys interacting with the same bait, and the other for observing the same bait-prey interactions multiple times, given that the null hypothesis is true, i.e. interactions to happen randomly. Our methods are different from a Bayesian method recently developed by Gilchrist *et al* [17], in which only the bait-prey protein interaction hypothesis was discussed. Instead, our hypothesis test methods belong to a "frequentist method", which have the advantage of not requiring a prior protein interaction distribution for an alternative hypothesis.

In the first test method, we are concerned with whether a particular bait tends to interact with many different preys. We want to distinguish whether a bait protein "indiscriminately" chooses an interaction partner by chance or by an un-characterized statistically significant process. In this model, we use r $(r \geq 1)$ to indicate the number of times that a search is replicated, i.e., the number of times the same bait has been observed in different searches. We use l to indicate the number of preys from all the replicated searches sharing the same bait. We use p and M according to the previously described definition. Under the null hypothesis, we assume that every prey-bait interaction is an independent Bernoulli trial with a success rate of p. There are $r \times M$ trials among r replicated searches. Therefore, the probability to obtain l or more preys by chance among r searches (p-value) can be calculated through a binomial distribution,

$$pvalue_{IWTE} = \Pr(L \geq l) = 1 - \sum_{i=0,1,\ldots,l-1} \binom{r \times M}{i} p^i (1-p)^i \qquad (2)$$

Where $pvalue_{\mathrm{IWTE}}$ is the individual-wise type I error for a bait. However, as a total of *N* searches were performed, and each bait is assumed to have *r* replicated searches, the family-wise type I error can be controlled in (3) (Westfall),

$$pvalue_{FWTE} = \Pr(L \geq l) = 1 - \{1 - pvalue_{IWTE}\}^{N/r} \qquad (3)$$

In Table 3, we tabulated the *p-values* under all the scenarios, ranging in l=1 to 10 and r=1 to 4, where N=200,000, M=7,000, and p=5.0e-5. Each cell in the table contains a "family-wise" p-value, which measures the significance level for discovering l preys in r replicated searches. For example, when only six interactions or less are discovered in a non-replicated search (r=1 and l=6), this observation is not significant since p-value=0.314. However, when l increases from 6 to 7, 8, and ≥9 for a fixed r=1, the p-value decreases to 1.86e-02, 8.15e-04, and ≤3.16e-05 respectively, suggesting the data being increasingly significant. This table also confirms that for a fixed l number of preys, the less search replications r it takes to observe all of them, the more significant the observation becomes.

Table 2. A list of P-value that measures the significance of observing *l* number of preys in *r* different searches. The scenarios that are significant at a *p-value* threshold of ≤ 0.05 are highlighted by shade and a bold font.

	r=1	*r =2*	*r=3*	*r=4*
l =6	3.14e-01	1.00e-00	1.00e-00	1.00e-00
l =7	**1.86e-02**	5.88e-01	1.00e-00	1.00e-00
l =8	**8.15e-04**	7.39e-02	6.19e-01	9.95e-01
l =9	**3.16e-05**	**5.90e-03**	1.05e-01	5.57e-01
l =10	**1.10e-06**	**4.11e-04**	**1.15e-02**	1.06e-01
l =11	**3.49e-08**	**2.60e-05**	**1.09e-03**	**1.40e-02**
l =12	**1.02e-09**	**1.51e-06**	**9.48e-05**	**1.63e-03**

If the statistical significant level is set as 0.05 the family-wise p-value, there are many significant conclusions that we can derive from this test result. For example, we can conclude that if we observe at least 7 preys interacting with a single bait in any search, the event is statistically significant (p-value=0.0186). For another example, if the same bait has appeared 3 times in different searches, we have to observe an additional 3 preys for these 10 (=7+3) preys to be taken as statistically significant (p-value=0.0115). In a final example, if a bait interacts with hundreds of other preys in a few different SY2H searches, according to the above result, we say the bait must be selected with a significant bias. This result supports many earlier findings of "sticky proteins" and highly interacting proteins serving as "interaction network hubs" [5, 18].

In the second test, we are concerned with the significance of identifying a protein interaction pair from experimental results. We want to know whether or not a protein interaction can be "trusted" for use in subsequent knowledge discovery tasks. We use t to indicate the number of times that a prey is discovered, and use r, p, and M as described early in this section. Using a binomial model, we can calculate the individual-wise type I error, $pvalue_{IWTE}$, for seeing the same prey appearing t times by chance in r replicated searches as the following:

$$pvalue_{IWTE} = \Pr(T \geq t) = 1 - \sum_{i=0,1,\ldots,t-1} \binom{r}{i} p^i (1-p)^i \qquad (4)$$

However, as all M preys are available, the family-wise type I error can be controlled in (5) (Westfall),

$$pvalue_{FWTE} = \Pr(L \geq l) = 1 - \{1 - pvalue_{IWTE}\}^{M} \quad (5)$$

In Table 4, we tabulated the *p*-values under scenarios for t=1 to 4 and for r=1, 2, 3, 4, where N=200,000, M=7,000, and p=5.0e-5. Each cell in the table contains a family-wise type I error *p*-value, which measures the significance level for discovering the same bait-prey interaction for t times in r replicated searches. For example, when an interaction is discovered only once in a single non-replicated search (r=1 and t=1), this observation is not significant since *p-value*=0.295. However, any replicated interaction identified from at least two different SY2H searches (r≥2 and t=2 to 4) is going to be significant, because the calculated p-value in all these scenarios are less than 0.05.

Table 3. A list of P-value that measures the significance of observing the same interaction pair *t* times in *r* different searches. The scenarios that are significant at a *p-value* threshold of ≤ 0.05 are highlighted by shade and a bold font.

	r=1	*r=2*	*r=3*	*r=4*
t =1	2.95e-01	5.03e-01	6.50e-01	7.53e-01
t =2	**--**	**1.74e-05**	**5.25e-05**	**1.05e-05**
t =3	**--**	**--**	**8.75e-10**	**3.50e-09**
t =4	**--**	**--**	**--**	**0.00e-00**

3.4 Biological Significance of Interactions

Following the discovery of statistically significant patterns in the "raw" data set , the next question arises, "How does one identify biologically significant protein interactions?" Not all statically significant interactions discovered in the previous section are biologically sensible, because a falsely identified human interacting protein can appear in many searches simply because this protein interacts with the yeast transcription factor in the SY2H system. To address this issue eventually, significant research efforts beyond this work is necessary, including efforts to perform complementary or validation experimental studies, conduct manual knowledge curations, and incorporate different types of biological data into the current computational analysis. In [19], we summarized the challenges and opportunities of integrating biological data such as gene expression information, functional annotations, homology information, and interaction network modules.

In this section, we describe an example of such integrative data analysis based on the annotations of protein's interaction partners. The null hypothesis is that proteins do not have specific functional or localization preferences when choosing their interaction partners. To collect statistics under the null hypothesis, we used a numerical re-sampling method, in which we randomly rewired the interaction network. Our randomization procedure, however, preserved each node's degree of connectivity and thus the overall network node degree distribution. For each randomly rewired network, we retrieved the interaction partners $v(n)$ of each protein n and calculated the frequency of occurrences of each annotation term among all the annotations, $A[v(n)]$, available for the proteins in $v(n)$. From this data, we computed the distribution functions for the fractions of each annotation term among all the terms assigned to protein's interaction partners. Since we were interested in statistically significant *co-occurrences* of the annotation terms, we actually calculated a conditional probability, $p[|N(t)|=k|t]$, to observe k occurrences of term t among $A[v(n)]$, given $t \in A[v(n)]$. The continuous approximation (calculating fractions instead of counts) is helpful for analyzing very small number of proteins with very large number of interaction partners, which would otherwise require expensive full network re-sampling to analyze.

Table 4. A summary report showing that highly interacting proteins, binned by their node degree range, have significantly high shares of interaction partners in diverse annotation categories (sampled).

Node Degree Range	20-30	31-40	41-80	>80
Development	24	26	22	26
Chaperone activity	15	7	15	25
Catalytic activity	48	16	4	2
Transporter activity	37	21	19	13
Motor activity	12	9	23	27
Signal transducer	42	16	14	15
Translation regulator activity	9	10	15	28
Extra-cellular	44	30	25	30
Enzyme regulator activity	14	15	6	9
Transcription regulator activity	24	32	29	27
Structural molecule activity	24	19	44	118
Defense/immune activity*	5	2	2	0
Cell adhesion molecule activity*	24	23	21	12
Apoptosis regulator activity*	7	2	5	1
Significant / Total	239/596 =40%	177/269 = 66%	188/398 = 47%	196/281 =70%

***** This term is obsolete in the current version of GO. Our analysis is still consistent since we used the concurrent versions of GO and the protein annotation mapping.

For annotations we used a vocabulary derived from the Gene Ontology (GO) database. Since the GO has directed acyclic graph quasi-hierarchical structure, for each annotation term we perform a "roll-up" similar to [5] by tracing the term back to all its "ancestors" at the GO level l=2 (l=0 is for the *root, l=1* is for "molecular function", "biological process", and "cellular component" labels). With this operation we generally avoid the problem of many terms being too narrow and not sufficiently represented to enable building robust statistics. It is also plausible biologically to expect multiple interactions with related proteins, belonging to the same general group (*e.g.* 'structural' or

'transcription factor'), rather than with a number of same very specific functional modules.

In Table 5, we present a summary of our results. Here, we are primarily interested in characterizing potentially self-activating and "sticky" false p ositive proteins—**highly interacting proteins** that we define here as those having >20 interaction partners. We selected four node degree ranges (20-30, 31-40, 41-80, and >81) and calculated significance levels for each annotation term according to the method outlined above. For instance, in the group of proteins with 20 to 30 interaction partners (first row) that consists of 596 proteins (see the "Significant/Total" column), there are 24 proteins that interact with significantly high numbers of proteins annotated with the "Development" GO term (or its more specific descendants), 15 interact with the significantly high numbers of proteins involved in "Chaperon Activities" *etc*. Total of 239 (40% of 596) proteins in this group have at least one annotation term overrepresented among their interaction partners (note that many proteins have more than one significant term in *A(v(n))*). From our result, we can conclude that 70% of the **promiscuously interacting proteins** (node degree >80) have statistically significant interaction patterns and thus can be biologically significant and active "functional hubs". Combined with the evidence from previous work and previous results in this work, we believe that they should not be recklessly dismissed, but rather thoroughly analyzed with all the biological evidence available. It should be noted that some proteins are involved in various activities under different conditions, so that when the whole set of interaction partners is analyzed regardless of the source tissue, developmental stage, *etc.*, no particular functional category may seem to be overrepresented. Thus, our estimates can be conservative.

4.DISCUSSION

In this study, we performed a comprehensive assessment of the human protein interactome data, which were derived from the SY2H method. We showed that this data set comprehensively surveyed the human proteome without an apparent bias in source chromosomal locations. We also showed that the data had a good reproducibility above 78% and a low false positive rate at approximately 5.5e-5. We developed several statistical data mining techniques to assess both the statistical and biological significance of interactions, especially for those data with replications and annotated GO term labels. We showed evidence were not random noises; instead, they could be "network hubs" of the cell signaling network. We also attributed the low noise in our data to the adoption of standard control in the experimental data generation process.

Several factors may have prevented similar insights into the protein interactome data "noise" issue from being developed until this work. First, protein interactome data were scarce until very recently. Not many researchers have access to this type of data; even fewer have first-hand experience trying to extract useful information from it. Second, there is a genuine lack of biological understanding in how Y2H method works. This may have presented the development of new concepts such as protein "network hubs". Third, many public data are produced in labs without the modern robot equipments or proper enforcement of standard operation protocols. Therefore, experimental variations are prevalent. Fourth, even today, the public protein interactome data set does not contain essential information such as protein interaction regions, interaction strengths, or replicated interactions under similar experimental conditions. With an ongoing surge of protein interactome data in the next few years, we hope our early results will restore some assurance to forthcoming data miners of this information.

5. REFERENCES

[1] Rain, JC, et al., *The protein-protein interaction map of Helicobacter pylori.* Nature, 2001. **409**(6817): p. 211-5.

[2] Ito, T, et al., *A comprehensive two-hybrid analysis to explore the yeast protein interactome.* Proc Natl Acad Sci U S A, 2001. **98**(8): p. 4569-74.

[3] Uetz, P, et al., *A comprehensive analysis of protein-protein interactions in Saccharomyces cerevisiae.* Nature, 2000. **403**(6770): p. 623-7.

[4] Giot, L, et al., *A protein interaction map of Drosophila melanogaster.* Science, 2003. **302**(5651): p. 1727-36.

[5] Chen, JY, et al. *Initial Large-scale Exploration of Protein-protein Interactions in Human Brain*. in *IEEE CSB Bioinformatics 2003*. 2003. Stanford, California.

[6] Li, S, et al., *A map of the interactome network of the metazoan C. elegans.* Science, 2004. **303**(5657): p. 540-3.

[7] Bartel, P and S Fields, eds. *The Yeast Two-Hybrid System*. Advances in Molecular Biology. 1997, Oxford University Press.

[8] Auerbach, D, et al., *The post-genomic era of interactive proteomics: facts and perspectives.* Proteomics, 2002. **2**(6): p. 611-23.

[9] Mrowka, R, A Patzak, and H Herzel, *Is there a bias in proteome research?* Genome Res, 2001. **11**(12): p. 1971-3.

[10] Bader, GD and CW Hogue, *Analyzing yeast protein-protein interaction data obtained from different sources.* Nat Biotechnol, 2002. **20**(10): p. 991-7.

[11] Legrain, P, J Wojcik, and JM Gauthier, *Protein--protein interaction maps: a lead towards cellular functions.* Trends Genet, 2001. **17**(6): p. 346-52.

[12] Chen, JY and JV Carlis, *Genomic Data Modeling.* Information Systems, 2003. **28**(4): p. 287-310.

[13] Pruitt, KD and DR Maglott, *RefSeq and LocusLink: NCBI gene-centered resources.* Nucleic Acids Res, 2001. **29**(1): p. 137-40.

[14] Ashburner, M, et al., *Gene ontology: tool for the unification of biology. The Gene Ontology Consortium.* Nat Genet, 2000. **25**(1): p. 25-9.

[15] Gavin, AC, et al., *Functional organization of the yeast proteome by systematic analysis of protein complexes.* Nature, 2002. **415**(6868): p. 141-7.

[16] Ho, Y, et al., *Systematic identification of protein complexes in Saccharomyces cerevisiae by mass spectrometry.* Nature, 2002. **415**(6868): p. 180-3.

[17] Gilchrist, MA, LA Salter, and A Wagner, *A statistical framework for combining and interpreting proteomic datasets.* Bioinformatics, 2004. **20**(5): p. 689-700.

[18] Hoffmann, R and A Valencia, *Protein interaction: same network, different hubs.* Trends Genet, 2003. **19**(12): p. 681-3.

[19] Chen, JY and AY Sivachenko, *Data Mining Challenges for Protein Interactomics Studies (accepted).* IEEE Magazine in Biology and Medicine, 2004.

Assessment of discretization techniques for relevant pattern discovery from gene expression data

Ruggero G. Pensa[1], Claire Leschi[1], Jérémy Besson[1,2] and Jean-François Boulicaut[1]
1: INSA Lyon, LIRIS CNRS FRE 2672, F-69621 Villeurbanne cedex, France
2: UMR INRA/INSERM 1235, F-69372 Lyon cedex 08, France
{ruggero.pensa,claire.leschi,jeremy.besson,jean-francois.boulicaut}@insa-lyon.fr

ABSTRACT

In the domain of gene expression data analysis, various researchers have recently emphasized the promising application of pattern discovery techniques like association rule mining or formal concept extraction from boolean matrices that encode gene properties. To take the most from these approaches, a needed step concerns gene property encoding (e.g., over-expression) and its need for the discretization of raw gene expression data. The impact of this preprocessing step on both the quantity and the relevancy of the extracted patterns is crucial. In this paper, we study the impact of discretization parameters by a sound comparison between the dendrograms, i.e., trees that are generated by a hierarchical clustering algorithm, computed from raw expression data and from the various derived boolean matrices. Thanks to a new similarity measure and practical validation over several gene expression data sets, we propose a method that supports the choice of a discretization technique and its parameters for each specific data set.

1. INTRODUCTION

Thanks to a huge research effort and technological breakthroughs, one of the challenges for molecular biologists is to discover knowledge from data generated at very high throughput. For instance, different techniques (including microarray [13] and SAGE [24]) enable to study the simultaneous expression of (tens of) thousands of genes in various biological situations. The data generated by those experiments can be seen as expression matrices in which the expression level of genes (rows) are recorded in various biological situations (columns). A toy example of some microarray data is the matrix in Tab. 1a.

Exploratory data mining techniques are needed that can, roughly speaking, be considered as the search for interesting bi-sets, i.e., sets of biological situations and sets of genes that are associated in some way. Indeed, it is interesting to look for groups of co-regulated genes, also known as *synexpression groups* [19], which, based on the guilt by association approach, are assumed to participate in a common function, or module, within the cell. A set of co-regulated genes and the set of biological situations that gives rise to this co-regulation is called a *transcription module*. Discovering transcription modules is one of the main goals in functional genomics.

Various techniques can be used to identify a priori interesting bi-sets. Biologists often use clustering techniques to identify sets of genes that have similar expression profiles (see, e.g., [14]). Statistical methods can be used as well (see, e.g., [16; 4]). It is also possible to look for these putative synexpression groups by computing the so-called frequent itemsets from boolean contexts that encode gene expression properties [1]. Deriving association rules from frequently co-regulated genes has been studied as well [3; 10]. Furthermore, putative transcription modules can be provided by computing the so-called formal concepts (see, e.g., [25]) in this kind of boolean data [21; 22].

A key issue for using these pattern discovery techniques from boolean data concerns gene expression property encoding. Let $\mathcal{O}$ denotes a set of biological situations and

	1	2	3	4	5
a	-1	6	0	12	9
b	3	-2	3	-3	1
c	0	5	-1	6	6
d	4	-1	2	-2	-1
e	-3	9	1	10	6
f	5	-3	3	-6	0
g	4	-4	3	-7	0
h	-2	2	-2	8	5

(a)

	1	2	3	4	5
a	0	1	0	1	1
b	1	0	1	0	1
c	0	1	0	1	1
d	1	0	1	0	0
e	0	1	0	1	1
f	1	0	1	0	1
g	1	0	1	0	1
h	0	0	0	1	1

(b)

	1	2	3	4	5
a	0	0	0	1	0
b	1	0	1	0	0
c	0	0	0	1	1
d	1	0	0	0	0
e	0	0	0	1	0
f	1	0	0	0	0
g	1	0	0	0	0
h	0	0	0	1	0

(c)

Table 1: An example of gene expression matrix (a) with two derived boolean matrices (b and c)

$\mathcal{P}$ denotes a set of genes. The expression properties can be encoded into $\mathbf{r} \subseteq \mathcal{O} \times \mathcal{P}$. $(o_i, g_j) \in \mathbf{r}$ denotes that gene j has the encoded expression property in situation i. Different expression properties might be considered like, e.g., over-expression, up or down regulation, strong variation. Generally, encoding is performed according to some discretization operators that, given user-defined parameters, transform each numerical value from raw gene expression data into one boolean value per gene property. Many operators can be used that typically compute thresholds from which it is possible to decide wether the true or the false value must be assigned. For instance, in Tab. 1b, an over-expression property has been encoded and genes a, c, and e are over-expressed together in situations 2, 4 and 5.

We consider that mining boolean gene expression data sets is extremely useful thanks to the patterns that can be extracted now with efficient algorithms (e.g., frequent closed set [7; 20; 26] or concept extractors [5]). In this context, the critical step of gene expression data discretization has not been studied enough while its impact on both the quantity and the relevancy of the extracted patterns is crucial. For instance, the density of the discretized data depends on the discretization parameters and the cardinalities of the resulting sets (collections of itemsets, association rules or formal concepts) can be very different.

In this paper, we propose a method that supports both the choice for a discretization technique and an informed decision about its parameters. We cooperate with molecular biologists that are used to collect important information about putative synexpression groups and transcription modules by using the hierarchical clustering technique that has been popularized by the Eisen software [14]. We decided to study the impact of discretization parameters by a sound comparison between the dendrograms that are generated by the same hierarchical clustering algorithm applied to both the raw expression data and various derived boolean matrices. Comparing trees by means of ad-hoc similarity measures has been studied a lot, including in the bioinformatics domain for the analysis of phylogenies (see, e.g., [18; 23; 15]). Other measures evaluate the quality of partitions w.r.t. a reference partition of the data set. The difficulty is then to identify on dendrograms the cut levels at which we can compare the partition on the real data set with the one on boolean data set.

The contribution of this paper is twofold. First, we propose a new similarity measure for binary trees (such as dendrograms generated by any hierarchical clustering algorithm), that is level independent, and depends for each node on its subtree structure. Next, we have studied the behavior of our measure on several gene expression data sets in order to support the choice a discretization technique and the discretization parameters that have to be used when encoding boolean gene expression properties in order to perform efficient pattern discovery techniques like association rule mining or formal concept discovery.

In Section 2, we define our similarity measure between two binary trees. In Section 3, we study the behavior of this measure for different gene expression data sets. Finally we consider in Section 4 the impact of our technique on a KDD process which finds biologically relevant information in a well-studied gene expression data set. Section 5 concludes.

2. COMPARING BINARY TREES

The problem of finding the best comparison method for trees is still open even though it has been considered in various application domains. Considering the analysis of phylogenies, distance measures between both rooted and unrooted trees have been designed to compare different phylogenetic trees concerning the same set of individuals (e.g., different species of animals having a common ancestor). Various distance metrics between trees have been proposed. The `nni` (nearest neighbor interchange) and the `mast` (maximum agreement subtree) are two of the most used metrics. `nni` has been introduced independently in [18] and [23] and its NP-completeness has been recently proved [11; 12]. `mast` has been proposed in [15], and [9] describes an efficient algorithm for computing this metrics on binary trees. These two approaches are tailored for the problem of comparing phylogenies where the goal is to measure some degree of isomorphism between two dendrograms representing the same species of biological organisms.

In our data mining problem, we have sets of objects (vectors of expression values for genes in various biological situations), that we want to process with a hierarchical clustering algorithm. Depending on the different discretization operations on raw expression data, a same clustering algorithm working on encoded boolean gene expression data can return (very) different results. We are looking for a method that supports the comparison of these various gene and/or situation dendrograms obtained on boolean data w.r.t. the common reference dendrogram that has been computed from the raw data. We need to measure both the degree of similarity of their structures and the similarity between the contents of their associated collections of clusters. We designed a simple measure which is also easy to compute. Intuitively, it depends on the number of matching nodes between the two trees we have to compare.

2.1 Definition of similarity scores

Let $\mathcal{O} = \{o_1, \ldots, o_n\}$ denote a set of n objects. Let T denote a binary tree built on $\mathcal{O}$. Let $\mathcal{L} = \{l_1, \ldots, l_n\}$ denote the set of n leaves of T associated to $\mathcal{O}$ for which, $\forall i \in [1 \ldots n], l_i \equiv o_i$. Let $\mathcal{B} = \{b_1 \ldots b_{n-1}\}$ denote the set of $n-1$ nodes of T generated by a hierarchical clustering algorithm starting from $\mathcal{L}$. By construction, we consider $b_{n-1} = r$, where r denotes the root of T. We define the two sets:

$$\delta(b_i) = \{b_j \in \mathcal{B} \mid b_j \textit{ is a descendent of } b_i\},$$

$$\tau(b_i) = \{l_j \in \mathcal{L} \mid l_j \textit{ is a descendent of } b_i\}.$$

An example of a tree for a set containing 8 objects (i.e., the genes from Tab. 1a) is given in Fig. 1. In this example, $\tau(b_3) = \{b, d, f, g\}$ and $\delta(b_3) = \{b_1, b_2\}$.

We want to measure the similarity between a tree T and a reference tree T_{ref} built on the same set of objects $\mathcal{O}$. For each node b_i of T, we define the following score (denoted

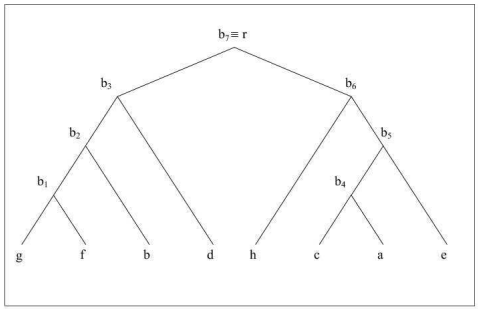


Figure 1: An example of binary tree

S_B and called **BScore**):

$$S_B(b_i, T_{ref}) = \sum_{b_j \in \delta(b_i)} a_j$$

$$a_j = \begin{cases} \frac{1}{|\tau(b_j)|}, & if \\ & \exists b_k \in T_{ref} \mid \tau(b_j) = \tau(b_k) \\ 0, & otherwise \end{cases} \quad (1)$$

In other terms, for a node b in T, its score depends both on the number of its matching nodes in T_{ref} ($b_k \in T_{ref}$ is a matching node for b if $\tau(b) = \tau(b_k)$) and $|\tau(b)|$. To obtain the similarity score of T w.r.t. T_{ref} (denoted S_T and called **TScore**), we consider the **BScore** value on the root, i.e.:

$$S_T(T, T_{ref}) = S_B(r, T_{ref}) \quad (2)$$

As usually, it is interesting to normalize the measure to get a score between 0 (for a tree which is totally different from the reference) and 1 (for a tree which is equal to the reference). For the **TScore** measure, since its max value depends on the tree morphology, we can normalize by $S_T(T_{ref}, T_{ref})$:

$$\overline{S_T}(T, T_{ref}) = \frac{S_T(T, T_{ref})}{S_T(T_{ref}, T_{ref})} \quad (3)$$

$\overline{S_T}(T, T_{ref}) = 0$ means that T is totally different from T_{ref}, i.e., there are no matching nodes between T and T_{ref}. Indeed, $\overline{S_T}(T, T_{ref}) = 1$ means that T is totally similar to T_{ref}, i.e., every node in T matches with a node in T_{ref}. Given two trees T_1 and T_2 and a reference T_{ref}, if $\overline{S_T}(T_1, T_{ref}) < \overline{S_T}(T_2, T_{ref})$, then T_2 is said to be more similar to T_{ref} than T_1 according to **TScore**.

Let us now provide a constructive definition to compute the **BScore**s for every node of the tree, and retrieve its value for the whole tree. Assume that functions $c_l(b_i)$ and $c_r(b_i)$ respectively return the left and the right child of b_i. In Fig. 1 $c_l(b_7) = b_3$ et $c_r(b_7) = b_6$. The **BScore** measure can be redefined as follows:

$$S_B(b_i, T_{ref}) = \sigma(c_l(b_i), T_{ref}) + \sigma(c_r(b_i), T_{ref}) \quad (4)$$

where

$$\sigma(b_k, T_{ref}) = \begin{cases} \frac{1}{|\tau(b_k)|} + S_B(b_k, T_{ref}), & if \\ \quad \exists b_j \in T_{ref} \mid \tau(b_k) = \tau(b_j) \\ S_B(b_k, T_{ref}), & otherwise \end{cases}$$

$$\sigma(l_k, T_{ref}) = 0, \ \forall l_k \in \mathcal{L}$$

This definition emphasizes that the **BScore** for each node depends on the **BScore** values of its children. The fact that each node "inherits" the similarity information of its children is useful when comparing two trees that result from a hierarchical clustering algorithm.

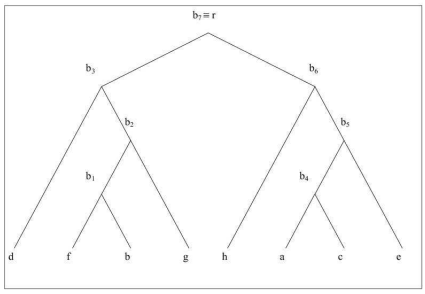
2.2 Comparison between gene dendrograms

Tab. 1a is a toy example of a gene expression matrix. Each row represents a gene vector, and each column represents a biological sample vector. Each cell contains an expression value for a given gene and a given sample. In this example, we have $\mathcal{O} = \{a, b, c, d, e, f, g, h\}$. A hierarchical clustering using the Pearson's correlation coefficient and the average linkage method (see, e.g., [14]) on the data from Tab. 1a leads to the dendrogram in Fig. 1.

Assume now that we discretize the expression matrix by applying two different methods used for over-expression encoding [3]. The first one considers the mean between the max and min values for each gene vector. Values that are greater than the average value are set to 1, 0 otherwise (Tab. 1b). A second method takes into account the max value for each gene vector. Values that are greater than 90% of the max value are set to 1, O otherwise (Tab. 1c).

Assume now that we use the same clustering algorithm on the two derived boolean data sets. The resulting dendrograms are shown in Fig. 2. Fig. 2a (resp. Fig. 2b) represents the gene dendrogram obtained by clustering the boolean matrix in Tab. 1b (resp. Tab. 1c).

a)



b)

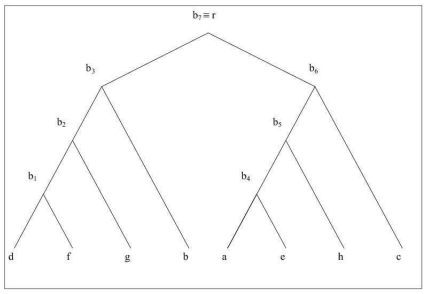


Figure 2: Gene trees built on two differently discretized matrices

We can now use the similarity score and decide which discretization is better for this gene expression data set, i.e., the one for which $\overline{S_T}(T, T_{ref})$ has the largest value. The common reference (T_{ref}) is the tree in Fig. 1. Let T_a and T_b denote the trees in Fig. 2a and 2b respectively. Using Equation 4, we obtain the results in Tab. 2.

To normalize the similarity scores, we just need to divide the **BScore**s of the root of the first two dendrograms,

	T_a			
Node	*Match*	τ	σ	S_B
b_1	-	$\{b, f\}$	0	0
b_2	b_2	$\{b, f, g\}$	0.33	0
b_3	b_3	$\{b, d, f, g\}$	0.58	0.33
b_4	b_4	$\{a, c\}$	0.5	0
b_5	b_5	$\{a, c, e\}$	0.83	0.5
b_6	b_6	$\{a, c, e, h\}$	1.08	0.83
b_7	b_7	$\mathcal{O}$	-	1.67

	T_b			
Node	*Match*	τ	σ	S_B
b_1	-	$\{d, f\}$	0	0
b_2	-	$\{d, f, g\}$	0	0
b_3	b_3	$\{b, d, f, g\}$	0.25	0
b_4	-	$\{a, e\}$	0	0
b_5	-	$\{a, e, h\}$	0	0
b_6	b_6	$\{a, c, e, h\}$	0.25	0
b_7	b_7	$\mathcal{O}$	-	0.5

	T_{ref}			
Node	*Match*	τ	σ	S_B
b_1	b_1	$\{f, g\}$	0.5	0
b_2	b_2	$\{b, f, g\}$	0.83	0.5
b_3	b_3	$\{b, d, f, g\}$	1.08	0.83
b_4	b_4	$\{a, c\}$	0.5	0
b_5	b_5	$\{a, c, e\}$	0.83	0.5
b_6	b_6	$\{a, c, e, h\}$	1.08	0.83
b_7	b_7	$\mathcal{O}$	-	2.17

Table 2: **BScore** values. Nodes matching in T_{ref} are in the *Match* columns.

by the **BScore** of the root of the reference dendrogram (Equation 3):

$$\overline{S_T}\,(T_a, T_{ref}) = \frac{S_T\,(T_a, T_{ref})}{S_T\,(T_{ref}, T_{ref})} = \frac{1.67}{2.17} = 0.77$$

$$\overline{S_T}\,(T_b, T_{ref}) = \frac{S_T\,(T_b, T_{ref})}{S_T\,(T_{ref}, T_{ref})} = \frac{0.5}{2.17} = 0.23$$

Since $\overline{S_T}\,(T_a, T_{ref}) > \overline{S_T}\,(T_b, T_{ref})$, the first discretization method is considered better for this data set w.r.t. the performed hierarchical clustering. In fact, in T_a, only node b_1 does not match (i.e., it does not share the same set of leaves) with any node in T_{ref}, while in T_b, there are only two nodes (b_3 and b_6) that match with some nodes in T_{ref}.
The same process can be applied to situation dendrograms by considering now that the objects are the situations. In practice, we perform both processes to support the choice of a discretization technique as illustrated in the next section.

3. COMPARING DIFFERENT DISCRETIZATION TECHNIQUES

Many discretization techniques can be used to encode gene expression properties from expression values that are either integer values (case for SAGE data [24]) or real values (case for microarray data [13]). In this paper, we consider for our experimental study only three techniques that have been used for encoding the over-expression of genes in [3]:

- "Mid-Ranged". The highest and lowest expression values are identified for each gene and the mid-range value is defined. For a given gene, all expression values that are strictly above the mid-range value give rise to value 1, 0 otherwise.
- "Max - X% Max". The cut off is fixed w.r.t. the maximal expression value observed for each gene. From this value, we remove a percentage X of this value. All expression values that are greater than the $(100 - X)\%$ of the Max value give rise to value 1, 0 otherwise.
- "X% Max". For each gene, we consider the situations in which its level of expression is in X% of the highest values. These genes are assigned to value 1, 0 otherwise.

We want to evaluate the relevancy of a discretization algorithm and its parameters according to the preserved properties w.r.t. a hierarchical clustering of the raw data. So, we have to compare the dendrograms obtained from the three different boolean matrices with the reference dendrogram.
We have considered three gene expression data sets: two microarray data sets and a SAGE data set. The first data set (CAMDA [8]) concerns the transcriptome of the intraerythrocytic developmental cycle of the plasmodium falciparum, a parasite that is responsible for a very frequent form of malaria. We have the expression values for 3 719 genes in 46 different time points. The second data set (Drosophila [2]) concerns the gene expression of drosophila melanogaster during its life cycle. We have the expression values for 3 030 genes and 81 biological samples, including both male and female adult individuals. The third one (human SAGE data from NCBI, see also [17; 22]) contains the expression values for 5 327 human genes in 90 different cancerous and not cancerous cellular samples belonging to different human organs.
In Tab. 3, we report the densities (i.e., the ratio of true values) of the boolean matrices produced with the "Mid-Ranged" method. In Fig. 3, we provide the density curves for the three data sets and depending on different thresholds for the "Max - X% Max" method (densities for the "X% Max" method are quite similar).

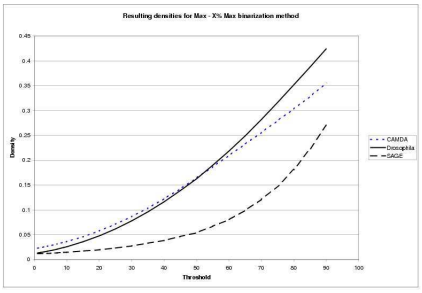


Figure 3: Density values for different "Max - X% Max" thresholds

We processed all the computed boolean matrices with a hierarchical clustering algorithm based on the centered Pearson's correlation coefficient and the average linkage method. The same algorithm with the same options has been applied to the three original matrices. Finally, for each data set, we have compared all the genes and situations trees derived from the boolean matrices with the reference trees. The results in terms of **TScore** (Equation 4) for the "Mid-Ranged" method, are summarized in Tab. 3. For the "Max - X% Max" and "X% Max" meth-

ods we summarize the results depending on the variation of the threshold X for the gene dendrograms in Fig. 4a and Fig. 4c, for the situation dendrograms in Fig. 4b and Fig. 4d. It is important to observe that, for each data set, we obtained the highest values of similarity scores for both the genes and the situations for almost the same discretization thresholds.

		Similarity score	
Data set	**Density**	*Genes*	*Situations*
CAMDA	0.313665	0.034155	0.746437
Drosophila	0.441146	0.059570	0.591343
SAGE	0.053958	0.110131	0.776736

Table 3: Similarity scores for clustering trees on Mid-Ranged discretized matrices

We have also applied the same clustering algorithm on various randomly generated boolean matrices based on the same sets of objects. Then, we have compared the resulting dendrograms with the reference. In the first two data sets (CAMDA and Drosophila), the similarity scores of the randomly generated boolean matrices are always very low or equal to 0. In the SAGE data set, given a density value, the gene scores resulting from randomly generated matrices are always lower than the ones obtained by any discretization method (while the situation scores are always negligible). One possible reason is that the discretized matrices are here very sparse compared to the first two data sets (see Fig. 3). Using a low threshold to discretize such a matrix does not make sense: obtained scores are similar to the scores which are computed on random boolean matrices. Moreover, using a high threshold value X for the "X% Max" discretization method leads to similarity scores that are close to those obtained for randomly generated matrices, though still higher. We can observe the behavior of this particular SAGE data set in Fig. 5.

To conclude this section, comparing dendrograms resulting from the clustering of different types of derived boolean matrices enables to choose the "best" discretization method and parameters for a given data set. If we analyze the graphics of similarity scores w.r.t. the thresholds used in the "Max - X% Max" and "X% Max" methods (see Fig. 4), we observe the presence of either a max or an asymptotic behavior. It means that the best choice for the discretization threshold could be a trade-off between the value for which we get the best similarity score, and the value for which the data mining task remains tractable.

4. AN APPLICATION TO A REAL PROBLEM

We have applied the proposed preprocessing technique to a real gene expression data set to validate our approach throughout a complete KDD process. We have decided to mine the data described in [2]. It concerns the gene expression of the Drosophila melanogaster during its life cycle. The expression levels of 4 028 genes are evaluated for 66 sequential time periods from the embryonic state till the adulthood. The total number of samples is 81 since the gene expression during the adult state is measured for male and female individuals. For our experiment we have used only a set of 20 time periods for adult individuals. This set is composed of 8 male adult samples, 8 female adult samples, 2 male and 2 female tudor samples. The set of genes we have used is derived from the original set from which we removed those genes that are under-expressed in all the 20 situations and over-expressed in at least 11 biological situations. We have obtained a 3 433 $\times$ 20 matrix which has been processed according to our methodology. The raw expression matrix has been discretized using the "Mid-Ranged" and "Max - X% Max" methods. The resulting boolean matrices and the original matrix have been processed with the same ascendant hierarchical clustering algorithm using Pearson's correlation coefficient and average linkage. Then, using our tree comparison technique, we have compared the gene and situation dendrograms. The similarity scores are presented in Fig. 6.

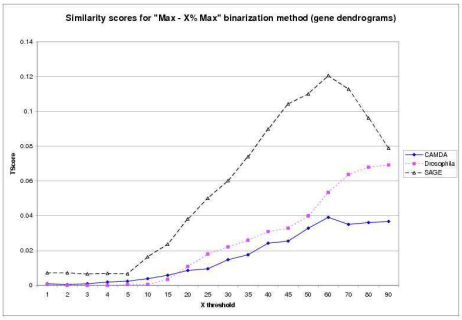
Our goal was to identify a particular class of genes, the so-called "male somatic genes", that characterizes the male adult individuals (see Table S30 in [2]). 31 of these 37 genes are present in our data set and we tried to search them by mining formal concepts in the various derived boolean matrices. Intuitively, formal concepts are maximal rectangles of true values in boolean matrices. For instance, in the boolean context from Tab. 1b, ({a,c,e},{2,4,5} is a formal concept, i.e., a strong association between two closed sets. We used the D-Miner algorithm [5; 6]) which extracts all the concepts satisfying some user-defined monotonic constraints. We extracted all the concepts with at least 3 situations and at least 20 genes. Then we have post-processed the extracted collection to keep those which concern only male individuals. Finally, we measured the number of male somatic genes which appear in the different sets of the post-processed concepts. To better evaluate the results, we also built two other sets of concepts: the collection of concepts which concern only female individuals, and the collection of concepts which involve at least one female individual. We summarize the results in Fig. 7.

The discretization threshold that gives the best similarity score and that we identify in both graphs from Fig. 6 (X = 54% for the "Max - X% Max" method), enables to retrieve 25 of the 31 male somatic genes from the concepts that concern only male individuals. Moreover, even though higher thresholds enable to retrieve more somatic genes, the slope of the curve, after the optimum, begins to decrease, while the slope of the curves of male somatic genes identified in concepts concerning female individuals starts to increase. Choosing the discretization threshold enables to control the trade-off between extraction completeness and noise impact.

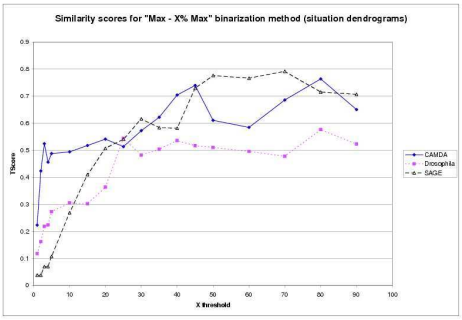
5. CONCLUSION

We defined a new pre-processing technique that supports the evaluation and assessment of different discretization techniques for a given gene expression data set. The evaluation is based on the comparison of dendrograms obtained by clustering various derived boolean matrices with the one obtained on the raw matrix while using the same clustering algorithm. The defined metrics is simple and we have validated its relevancy on different real data sets and on a biological problem. This is a step towards a better understanding of a crucial pre-processing step when we want to apply the extremely promising pattern discovery techniques based on set patterns.

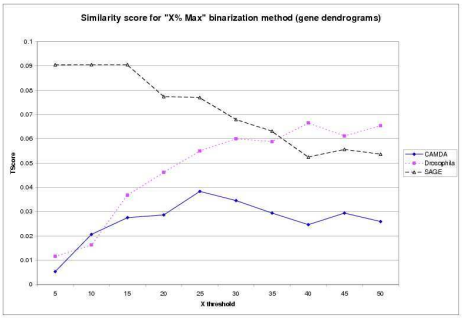
a)



b)



c)



d)

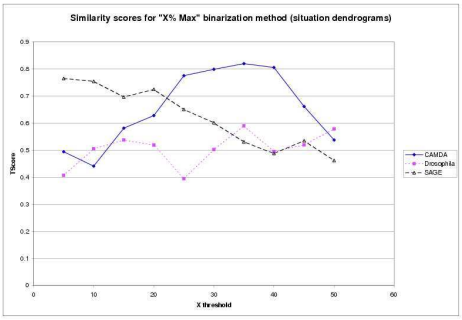


Figure 4: Similarity scores w.r.t. different thresholds for "Max - X%Max" (a and b) and "X%Max" (c and d) discretization methods

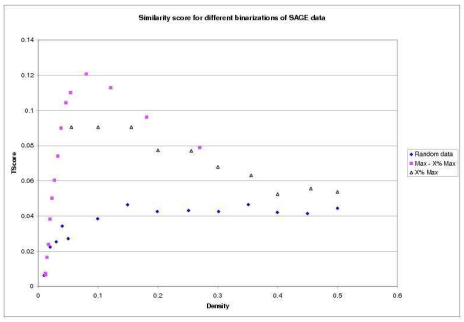
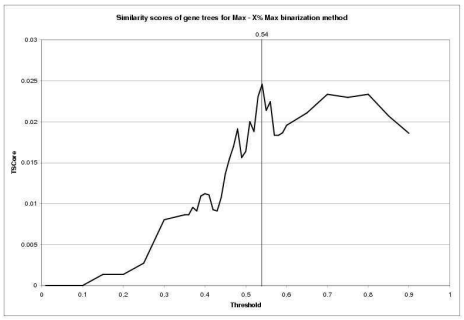


Figure 5: Similarity scores depending on density for "Max - X%Max", "X%Max" and random discretization methods applied to SAGE data



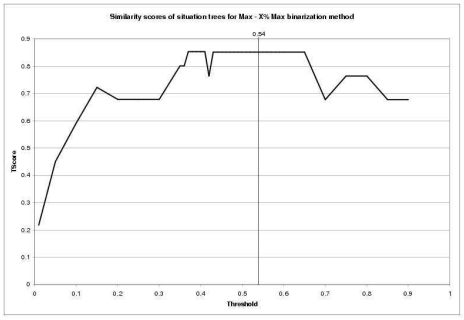


Figure 6: Similarity scores depending on various thresholds for "Max - X%Max" discretization method

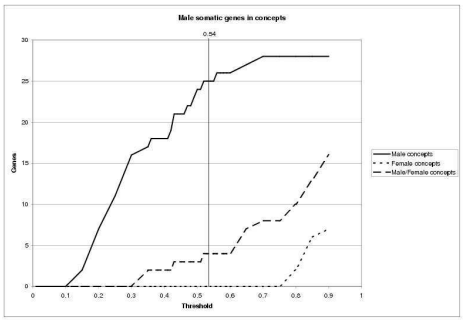


Figure 7: Number of identified male somatic genes w.r.t. discretization thresholds for different sets of concepts

6. ACKNOWLEDGEMENTS

The authors want to thank Céline Robardet, Sylvain Blachon and Olivier Gandrillon for the pre-processing of the SAGE data set, and Sophie Rome for stimulating discussions and her participation to the Drosophila application.

7. REFERENCES

[1] R. Agrawal, H. Mannila, R. Srikant, H. Toivonen, and A.I. Verkamo. Fast discovery of association rules. In *Advances in Knowledge Discovery and Data Mining*, pages 307–328. AAAI Press, 1996.

[2] M.N. Arbeitman, E.E. Furlong, F. Imam, E. Johnson, B.H. Null, B.S. Baker, M.A. Krasnow, M.P. Scott, R.W. Davis, and K.P. White. Gene expression during the life cycle of drosophila melanogaster. *Science*, 297:2270–2275, september 2002.

[3] C. Becquet, S. Blachon, B. Jeudy, J-F. Boulicaut, and O. Gandrillon. Strong-association-rule mining for large-scale gene-expression data analysis: a case study on human SAGE data. *Genome Biology*, 12, November 2002.

[4] S. Bergmann, J. Ihmels, and N. Barkai. Iterative signature algorithm for the analysis of large-scale gene expression data. *Physical Review*, 67, March 2003.

[5] J. Besson, C. Robardet, and J-F. Boulicaut. Constraint-based mining of formal concepts in transactional data. In *Proceedings PaKDD'04*, volume 3056 of *LNAI*, pages 615–624, Sydney, Australia, May 2004. Springer-Verlag.

[6] J. Besson, C. Robardet, J-F. Boulicaut, and S. Rome. Constraint-based concept mining and its application to microarray data analysis. *Intelligent Data Analysis journal*, 9(1), 2004. To appear.

[7] J-F. Boulicaut and A. Bykowski. Frequent closures as a concise representation for binary data mining. In *Proceedings PAKDD'00*, volume 1805 of *LNAI*, pages 62–73, Kyoto, Japan, April 2000. Springer-Verlag.

[8] Z. Bozdech, M. Llinás, B. Lee Pulliam, E.D. Wong, J. Zhu, and J.L. DeRisi. The transcriptome of the intraerythrocytic developmental cycle of plasmodium falciparum. *PLoS Biology*, 1(1):1–16, October 2003.

[9] R. Cole and R. Hariharan. An o(n log n) algorithm for the maximum agreement subtree problem for binary trees. In *Proceedings ACM-SIAM Symposium SODA'96*, pages 323–332, Atlanta, USA, January 1996.

[10] C. Creighton and S. Hanash. Mining gene expression databases for association rules. *Bioinformatics*, 19(1):79 – 86, 2003.

[11] B. DasGupta, X. He, T. Jiang, M. Li, J. Tromp, and L. Zhang. On distances between phylogenetic trees. In *Proceedings ACM-SIAM Symposium SODA'97*, volume 55, pages 427–436. 1997.

[12] B. DasGupta, X. He, T. Jiang, M. Li, J. Tromp, and L. Zhang. On computing the nearest neighbor interchange distance. In *Discrete mathematical problems with medical applications*, pages 125–143, Providence, USA, 2000. Amer. Math. Soc.

[13] J.L. DeRisi, V.R. Iyer, and P.O. Brown. Exploring the metabolic and genetic control of gene expression on a genomic scale. *Science*, 278:680–686, 1997.

[14] M.B. Eisen, P.T. Spellman, P.O. Brown, and D. Botstein. Cluster analysis and display of genome-wide expression patterns. *Proc. Natl. Acad. Sci. USA*, 95:14863–14868, December 1998.

[15] C.R. Finden and A.D. Gordon. Obtaining common pruned trees. *Journal of Classification*, 2:255–276, 1985.

[16] J. Ihmels, G. Friedlander, S. Bergmann, O. Sarig, Y. Ziv, and N. Barkai. Revealing modular organization in the yeast transcriptional network. *Nature Genetics*, 31:370–377, august 2002.

[17] A.E. Lash, C.M. Tolstoshev, L. Wagner, G.D. Schuler, R.L. Strausberg, G.J. Riggins, and S.F. Altschul. SAGEmap: A public gene expression resource. *Genome Research*, 10(7):1051–1060, July 2000.

[18] G. W. Moore, M. Goodman, and J. Barnabas. An iterative approach from the standpoint of the additive hypothesis to the dendrogram problem posed by molecular data sets. *Journal of Theoretical Biology*, 38:423–457, 1973.

[19] C. Niehrs and N. Pollet. Synexpression groups in eukaryotes. *Nature*, 402:483–487, 1999.

[20] J. Pei, J. Han, and R. Mao. CLOSET an efficient algorithm for mining frequent closed itemsets. In *Proceedings ACM SIGMOD Workshop DMKD'00*, pages 21–30, Dallas, USA, May 2000.

[21] F. Rioult, J-F. Boulicaut, B. Crémilleux, and J. Besson. Using transposition for pattern discovery from microarray data. In *Proceedings ACM SIGMOD Workshop DMKD'03*, pages 73–79, San Diego, USA, June 2003.

[22] F. Rioult, C. Robardet, S. Blachon, B. Crémilleux, O. Gandrillon, and J-F. Boulicaut. Mining concepts from large sage gene expression matrices. In *Proceedings KDID'03 co-located with ECML-PKDD 2003*, pages 107–118, Catvat-Dubrovnik, Croatia, September 2003.

[23] D. F. Robinsons. Comparison of labeled trees with valency three. *Journal of Combinatorial Theory, Series B*, 11:105–119, 1971.

[24] V.E. Velculescu, L. Zhang, B. Vogelstein, and K.W. Kinzler. Serial analysis of gene expression. *Science*, 270:484–487, 1995.

[25] R. Wille. Restructuring lattice theory: an approach based on hierarchies of concepts. In I. Rival, editor, *Ordered sets*, pages 445–470. Reidel, 1982.

[26] M. J. Zaki and C. J. Hsiao. Charm: An efficient algorithm for closed itemset mining. In *Proceedings SDM'02*, Arlington, USA, Avril 2002.

Meta-classification of Multi-type Cancer Gene Expression Data

Benny Y.M. Fung
Department of Computing,
The Hong Kong Polytechnic University
Hung Hom, Kowloon, Hong Kong
csymfung@comp.polyu.edu.hk

Vincent T.Y. Ng
Department of Computing,
The Hong Kong Polytechnic University
Hung Hom, Kowloon, Hong Kong
cstyng@comp.polyu.edu.hk

ABSTRACT

Massive publicly available gene expression data consisting of different experimental conditions and microarray platforms introduce new challenges in data mining when integrating multiple gene expression data. In this work, we proposed a meta-classification algorithm, which is called MIF algorithm, to perform multi-type cancer gene expression data classification. It uses regular histograms for gene expression levels of certain significant genes to represent sample profiles. Differences between profiles are then used to obtain dissimilarity measures and indicators of predictive classes. In order to demonstrate the robustness of the algorithm, 10 different data sets, which are individually published in 8 publications, are experimented. The results show that the MIF algorithm outperforms the simple majority-voting meta-classification algorithm and has a good meta-classification performance. In addition, we also compare our results with other researchers' works, and the comparisons are impressive. Finally, we have confirmed our findings with cancer/testis (CT) immunogenic gene families of heterogeneous samples.

Keywords

Gene expression, meta-classification, heterogeneous, multi-type

1. INTRODUCTION

Although DNA microarray techniques bring breakthroughs to cancer study, massive publicly available gene expression data, which are conducted by different laboratories with various experimental conditions and microarray platforms, introduce new challenges to conduct data mining with an integration of multiple and heterogeneous gene expression data. For gene expression data in cancer study, the advance of data mining leads to the discovery of global cancer profiling, patient classification, tumor classification, tumor-specific molecular marker identification and pathway exploration [15]. Different mining algorithms have been proposed, and significant findings are exploited corresponding to different algorithms. For most cases, validations of findings are done by a series of biological experiments or laboratorial works. However, in terms of efficiency and effectiveness of mining algorithms with respect to clinical applicability and robustness, the validations are mainly restricted by cross-validation or sub-sampling within a single data set [4], [11]. This validation scheme is not sufficiently to draw conclusions because of the problems of over-fitting and homogeneity within a single data set. To avoid these problems, there are two potential solutions: (1) it is required to validate mining algorithms with heterogeneous data sets consisting of different microarray platforms and experimental conditions, and (2) meta-analysis is performed with a number of heterogeneous data sets so that it can make meta-decisions with an integration of these data sets, rather than with individual data sets [5], [19].

To perform classification of heterogeneous data consisting of multi-type cancer, some common features (i.e. significant genes) must be founded in various cancer types. Subsets of genes, which are called cancer/testis (CT) immunogenic gene families, are recently proposed to have associations with one or more than one cancer type. Van der Bruggen et al. [23] suggested an approach to identify the molecular definition of tumor antigens recognized by T cells, and this approach leads to the discovery of various human tumor antigens, such as MEGEA1 and BAGE. Discovered tumor antigens are recently grouped into distinct subsets, and the subsets are named as cancer/testis (CT) immunogenic gene families. Currently, researchers have discovered 44 CT immunogenic genes families consisting of 89 individual genes in total [20].

In our previous works, we proposed a measure called "impact factors (IFs)" to improve the classification performance of heterogeneous gene expression data [7], [8]. In this paper, we extend the works and propose a meta-classification algorithm, which is called Majority-voting with Impact Factors (MIF) algorithm, to classify multi-type cancer gene expression data consisting of both different cancer types and microarray platforms. In order to validate the reliability and robustness of the MIF algorithm, 10 gene expression data sets, which are published in 8 different publications, are experimented, and the classification performance of the MIF algorithm is not only compared with the simple majority-voting meta-classification algorithm, but also with results of other researchers in [2].

2. RELATED WORKS

Recent progress in mining gene expression data is to discover knowledge from multiple and heterogeneous gene expression data. Some works are concerning theoretical flexibility to integrate gene expression data with various microarray platforms and technologies. Lee et al. [10] and Kuo et al. [9], respectively, described different approaches based on simultaneous mutual validation of large numbers of genes using two different microarray platforms. They used the NCI-60 data sets consisting of spotted cDNA arrays and Affymetrix oligonucleotide chips. Choi et al. [5] proposed a systematic integration of gene expression data based on normalizing data with an estimated means of other data sets.

For application level, classification is one of the common areas in data mining of gene expression data. Ng et al. [13] proposed a method to perform subtype classification with six different gene expression studies on Saccharomyces cervisiae. Recently, Bloom et al. [2] conducted a study of multi-platform, multi-type and multi-site classification on cancer gene expression data. In the study, 15 cancer types, published in 4 different publications, are experimented.

Meta-classification approaches are mainly divided into three categories [21]. The first category is to average individual

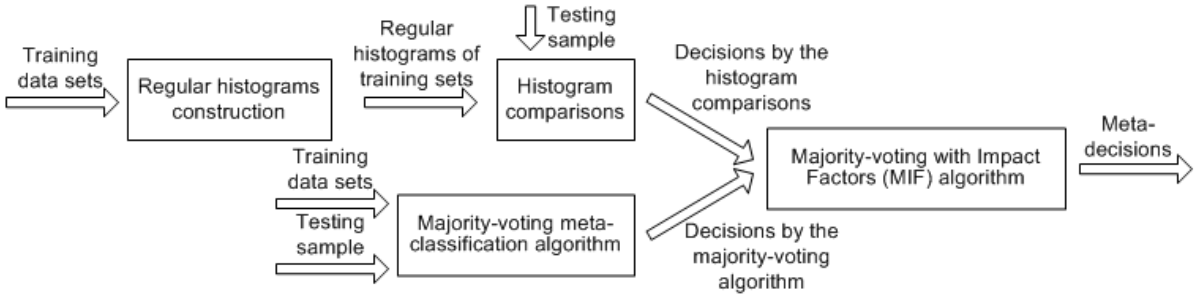


Figure 1. Process overview of the MIF algorithm.

decisions of different element classifiers without altering the original learning algorithms of the element classifiers. The second category is to predict the right learning algorithm or classifier for a particular problem from a set of element classifiers based on analyzing the fitness of the characteristics of testing data sets. The last category is to take a sub-sample of the entire data set and try each algorithm on this sub-sample. Among these three categories, the category of model averaging draws more attention in the literatures. For gene expression data, most works also belong to the category of model averaging. Some works include majority-voting [3], Bayesian combination [4], weighted-voting [4] and neural network ensembles [26].

3. MIF ALGORITHM

In this work, we proposed a meta-classification algorithm, called Majority-voting with Impact Factors (MIF) algorithm, to perform multi-type cancer gene expression data classification. It uses regular histograms for gene expression levels of certain significant genes to represent the profiles of samples. Differences between profiles are then used to obtain dissimilarity measures and indicators of predictive classes. The regular histograms are constructed by the uniform partitioning technique with maximum and minimum expression levels of the significant genes as upper and lower bounds. It aims at estimating densities of expression levels of significant genes in terms of relative positioning with respect to the upper and lower bounds. For a new sample, it compares its histograms with the histograms of individual classes in training sets. The classes with smaller dissimilarity measures are set as predictive classes for the new sample. As the same time, the majority-voting meta-classification algorithm is performed with the new sample too. If the decisions derived from the regular histogram comparisons and the majority-voting algorithm are the same, weighted scores corresponding to individual classes, which are based on the impact factors (IFs), are accumulatively adjusted the dissimilarity measures of the corresponding classes. On the other hands, if their decisions are different, there are no such weighted scores, and the dissimilarity scores are increased according to the results of the majority-voting algorithm. Figure 1 shows the process overview.

Here, we describe the MIF algorithm in details. First of all, individual regular histograms of every sample in each class in training sets are constructed [12]. Suppose that there are m training sets represented by the vector $X=(X_1, X_2, \ldots, X_m)$, and $X_i=(x_{i,1}, x_{i,2}, \ldots, x_{i,l}, x_{i,l+1}, \ldots, x_{i,n})$ be the training set i with l normal samples and $(n-l)$ cancer samples. The expression levels of gene g in X_i be represented by a vector $g=(e_{i,1}, e_{i,2}, \ldots, e_{i,n})$, where $e_{i,j}$ represents the expression level of g in sample j of set i (i.e. X_i), and $c=\{Normal,Cancer\}$ be the class vector such that $x_{i,j}.c$ representing the classes of sample j in set i. The algorithm for the regular histogram construction for training samples is shown in figure 2.

Inputs: aligned training samples sets X, number of bins n_b, number of significant genes n_g

Outputs: pairs of regular histograms for all training samples sets H_{Normal} and H_{Cancer}, sets of significant genes for all training sets G

1. **variables**:
2. $temp_{Normal}$ and $temp_{Cancer}$ be the temporary sets of regular histograms for each candidate of X_i, $temp_{Sig}$ be the temp set of significant for X_i, α be the percentage of bin candidates to be trimmed
3. *for i = 1 to size(X)*
4. $temp_{Normal} = \phi$;
5. $temp_{Cancer} = \phi$;
6. $temp_{Sig} = find\_sig\_genes\ (X_i)$;
7. $G = G + temp_{Sig}$;
8. *for j = 1 to size(X_i)*
9. *if* ($x_{i,j}.c = Normal$)
10. $temp_{Normal} = temp_{Normal} + hist\_proc(x_{i,j}, n_b, temp_{Sig})$;
11. *else*
12. $temp_{Cancer} = temp_{Cancer} + hist\_proc(x_{i,j}, n_b, temp_{Sig})$;
13. *end if*
14. *end for*
15. $H_{Normal} = H_{Normal} + normalize\ (temp_{Normal}, \alpha)$;
16. $H_{Cancer} = H_{Cancer} + normalize\ (temp_{Cancer}, \alpha)$;
17. *end for*

Figure 2. Algorithm for calculating regular histograms for training samples sets.

In figure 2, for each training set X_i, where $X_i \in \{X\}$, significance of genes in X_i is calculated and ranked accordingly in the function "find_sig_genes" at code line 6. The common and widely used statistical method t-test is used to rank significance of the genes [6]. In the t-test, its sign is determined by the numerator. Therefore, the t-values are positive if the mean of normal class is larger than that of cancer class and negative if the mean of normal class is smaller than that of cancer class. Hence, taking genes from both tails from the sorted list, including positive and negative t-values, can assume that the same proportions of genes from both classes are considered. Extracted significant genes sets, $G=\{G_1, G_2, \ldots, G_m\}$, where G_i is the significant gene set in training X_i, are later used to construct and compare the histograms of testing samples.

At code lines 10 and 12 in figure 2, the function "hist_proc" is invoked to construct the regular histograms. The maximum and minimum expression levels among those extracted significant genes are set as the upper and lower bounds of the histograms.

Samples belong to the same classes of the same training sets may have different values for upper and lower bounds. However, we are only interested in the densities of expression levels with respect to sample-based maximum and minimum expression levels, which is in relative positioning. Therefore, if the absolute differences of a sample between two bounds are smaller than other samples, their global differences among significant genes will be smaller in a similar ratio as the bounds also. As a result, the effects of the absolute differences can be eliminated.

The uniform partitioning technique is used to evenly divide the distance between the upper and lower bounds into a required number of bins n_b. Each bin width is defined by *(upper-lower)/*n_b. Each data set should have *l* and *(n–l)* different regular histograms for normal and cancer samples, and all histograms should have n_b bins because of the uniform partitioning. For example, figure 3 shows an example. Assume that there are 100 significant genes, n_b is 10 bins, and the upper and lower bounds are 4917 and -652. By applying the uniform partitioning technique, each bin width is [4917-(-652)]/10=557 to nearest integer. Expression levels of identified significant genes are then mapped to different bins with respect to their expression levels, and the results are shown in figure 3. At the end, the regular histogram of the illustrated sample is represented by the vector of (0.11, 0.76, 0.07, 0.02, 0.01, 0, 0, 0.01, 0, 0.02).

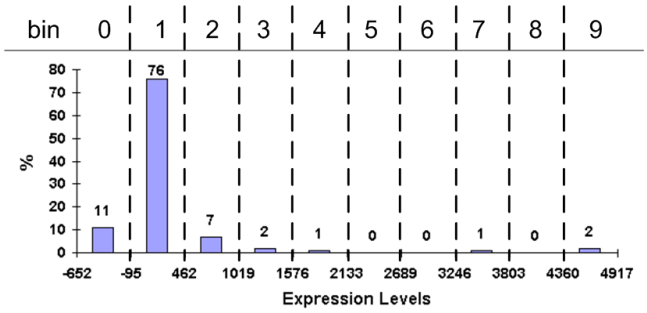


Figure 3. Example of regular histogram's construction for expression levels of significant genes.

After all the histograms corresponding to the same class of the same training sets (i.e. the *for* loop at code line 8) have been computed, α% candidate bins with highest and lowest bin values are trimmed to eliminate the effects of outliners. Remaining bins are then accumulated to form a representative histogram of individual classes in the data sets. Since some entries are trimmed, the value of the sum of all bin values at the representative histograms can be unbounded. It causes inconsistent scaling when comparing with other histograms. In order to have consistent comparisons, normalization is done so that the sum of all bin values in a single representative histogram to have the sum equals to 1. Finally, all representative histograms for individual training sets are added to H_{Normal} and H_{Cancer}. To use the same example in figure 3, the resultant vector becomes (0.76, 0.07, 0.02, 0.01, 0, 0, 0.01, 0) after 5% of candidate bins with highest and lowest bin values are trimmed. In addition, the normalized vector becomes approximately (0.87, 0.09, 0.02, 0, 0, 0.02, 0) in order to have sum equals to 1.

With the computed H_{Normal} and H_{Cancer}, comparisons of the histograms between training and testing samples can be performed. Figure 4 shows the algorithm of the comparisons.

Inputs: pairs of regular histograms for all training sample sets H_{Normal} and H_{Cancer}, sets of significant genes for all training sets *G*, testing sample *s*, number of bins n_b

Outputs: predictive classes by the regular histogram comparisons C_{Hist}

1. **variables**:
2. H_s be the temporary variable of the regular histogram of the testing sample
3. *for i = 1 to size*(H_{Normal})
4. H_s = *hist_proc*(*s*, n_b, G_i);
5. *if* (*dis*(H_s, $H_{Normal,\ i}$) < *dis*(H_s, $H_{Cancer,\ i}$))
6. C_{Hist} = C_{Hist} + {*Normal*};
7. *else*
8. C_{Hist} = C_{Hist} + {*Cancer*};
9. *end if*
10. *end for*

Figure 4. Algorithm for the comparisons of regular histograms between testing and training samples.

First of all, regular histogram of the testing sample *s* with respect to the significant genes set *G* of the training sets is computed. Then, dissimilarity measures between the testing sample and individual classes of training sets are computed, respectively. Assume that $H_s(b)$ be the regular histograms of the testing sample with bin *b*, and $H_c(b)$ is the regular histograms of the classes in the training sets with bin *b*, where *c={Normal,Cancer}*. Now, the dissimilarity measures, *dis*, between two histograms are calculated as:

$$dis(H_s, H_c | c \in \{Normal, Cancer\}) = \sum_b \frac{|H_s(b) - H_c(b)|}{|H_s(b) + H_c(b)|} \qquad (1)$$

The second step is to compare the histogram of the testing sample to pairs of the histograms in each training set and determine predictive classes of the new sample with respective to individual training sets in the code segment from line 5 to line 9 in figure 4. For each training set, there are two histograms corresponding to it, one for each class. The dissimilarity measures of normal and cancer classes are compared, and the classes with smaller values of the measures are set as the predictive classes of the testing sample, and assigned as a new element in set C_{Hist}. Since there is a single prediction for each training set, so there are m elements in C_{Hist} for m different training sets.

At the same time, the majority-voting meta-classification algorithm is performed. In [8], we proposed an empirically-driven model averaging method to integrate individual classification decisions to form meta-decisions. Suppose that there is a data set *D*, and the data are arisen from *k* possible models (i.e. combinations of classifiers and data sets), $M=(M_1, \ldots, M_k)$. If Δ is the quantity of interest (i.e. classification performance), then its posterior distribution of Δ in data set *D* is:

$$pr(\Delta | D) = pr(\Delta | \beta_1, \ldots, \beta_k, D) = \sum_{i=1}^{K} (\beta_i \times pr(\Delta | M_k, D)) \qquad (2)$$

, where β_i is the quantity of pre-knowledge for model M_i, and it is defined as:

$$\beta_i = \frac{acc(D,M_i)\ x\ S_p(D,M_i)x\ S_n(D,M_i)}{\sum_{l=1}^{K} acc(D,M_l)\ x\ S_p(D,M_l)x\ S_n(D,M_l)} \quad (3)$$

, where $acc(D,M_i)$, $S_p(D,M_i)$ and $S_n(D,M_i)$ are the classification accuracy, specificity and sensitivity of model M_i with data set D.

To perform the majority-voting algorithm, K is set to 1 in equation 2. Therefore, we only consider a single model each time, and finally there are k individual decisions for k different models. Hence, the equation is rewritten as:

$$pr(\Delta \mid D) = pr(\Delta \mid \beta_i, D) = \sum_{i \in K} (\beta_i \times pr(\Delta \mid M_i, D)) \quad (4)$$

If there are m and k different training sets and classifiers, there will be $(m \times k)$ individual decisions for the testing sample (i.e. each model produce a decision). For each decision, it is determined by a pair of Δ. Since we are interested in predictive classes of testing sample s, represented as $s.c$, one way to make the prediction is to compare the values of $pr(s.c=Normal|D)$ and $pr(s.c=Cancer|D)$, where $c \in \{Normal, Cancer\}$. If $pr(s.c=Normal|D)$ is larger than $pr(s.c=Cancer|D)$, assigned predictive classes are normal. Otherwise, it is assigned as cancer. In order form meta-decisions among individual decisions, the majority-voting algorithm in equation 5 assigns predictive classes, C_{Vote}, which are the most often predictive classes of individual decisions $s.c_i$.

$$(s.c \mid s.c \in \{C_{Vote}\}) = \underset{c \in \{Normal, Cancer\}}{arg\,max} \sum_{i:\{s.c_i = c\}} 1 \quad (5)$$

Inputs: testing sample s, sets of significant genes for all training sets G, number of bins n_b, predictive classes by the regular histogram comparisons C_{Hist}, predictive classes by the majority-voting algorithm C_{Vote}, impact factors for normal and cancer classes IF_{Normal} and IF_{Cancer}, pairs of regular histograms for all training sample sets H_{Normal} and H_{Cancer}, pre-knowledge measures corresponding to training sets β.

Outputs: meta-decisions C_{Pred}

1. **variables**:
2. d_{Normal} and d_{Cancer} be the dissimilarity values to normal and cancer classes, d_{Acc_normal} and d_{Acc_cancer} be the accumulative dissimilarity values to normal and cancer classes
3. *for i = 1 to size(C_{Hist})*
4. *if* ($C_{Hist,\,i} = C_{Vote,\,i}$)
5. *if* ($C_{Hist,\,i}$ = Normal)
6. $d_{Normal} = \beta_i \times IF_{Normal,\,i} / IF_{Cancer,\,i} \times dis(hist\_proc(s, n_b, G_i), H_{Normal,\,i}) / dis(hist\_proc(s, n_b, G_i), H_{Cancer,\,i})$;
7. *else*
8. $d_{Cancer} = \beta_i \times IF_{Cancer,\,i} / IF_{Normal,\,i} \times dis(hist\_proc(s, n_b, G_i), H_{Cancer,\,i}) / dis(hist\_proc(s, n_b, G_i), H_{Normal,\,i})$;
9. *end if*
10. *else*
11. *if* ($C_{Hist,\,i}$ = Normal)
12. $d_{Normal} = \beta_i \times IF_{Cancer,\,i} / IF_{Normal,\,i}$;
13. *else*
14. $d_{Cancer} = \beta_i \times IF_{Normal,\,i} / IF_{Cancer,\,i}$;
15. *end if*
16. *end if*
17. $d_{Acc\_normal} = d_{Acc\_normal} + log2\ (d_{Normal})$;
18. $d_{Acc\_cancer} = d_{Acc\_cancer} + log2\ (d_{Cancer})$;
19. *end for*
20. *if* ($d_{Acc\_normal} < d_{Acc\_cancer}$)
21. $C_{Pred} = C_{Pred} + \{Normal\}$;
22. *else*
23. $C_{Pred} = C_{Pred} + \{Cancer\}$;
24. *end if*

Figure 5. MIF algorithm.

Figure 5 shows the MIF (Majority-voting with Impact Factors) algorithm. It is an adoption of the decisions of the regular histogram comparisons, impact factors and majority-voting algorithm. In the figure, the combined meta-decisions are C_{Pred}. In the regular histogram comparisons, there are m individual decisions since there is a single decision corresponding to each training set. In contrast, there are $(m \times k)$ individual decisions from the majority-voting algorithm since there is a single decision corresponding to each training set together with a type of classifiers. Therefore, the decisions of the regular histogram comparisons are compared k times with that of the majority-voting algorithm of the same training set. IF_{Normal} and IF_{Cancer} are measures proposed in [7]. They define inter-experimental variations of a heterogeneous testing sample to normal and cancer classes of training samples, and they are expressed as IF_{Normal} and IF_{Cancer}.

Individual decisions of the regular histogram comparisons and the majority-voting algorithm are compared in the code segment from line 4 to line 16 in figure 5. If they are in the same decisions, equation 6 and 7 are applied for decisions of normal and cancer.

$$d_{Normal} = \beta_i \times IF_{Normal,i} / IF_{Cancer,i} \times dis(\alpha, H_{Normal,i}) / dis(\alpha, H_{Cancer,i}) \quad (6)$$

, where $\alpha = dis\ (hist\_proc(s, n_b, G_i)$

$$d_{Cancer} = \beta_i \times IF_{Cancer,i} / IF_{Normal,i} \times dis(\alpha, H_{Cancer,i}) / dis(\alpha, H_{Normal,i}) \quad (7)$$

, where $\alpha = dis\ (hist\_proc(s, n_b, G_i)$

For both equations, β_i is the magnitude of pre-knowledge for model M_i, which is calculated by equation 3. The factors of $(IF_{c1,i}/IF_{c2,i})$, given that $c1,c2 \in \{Normal, Cancer\}$ and $c1 \neq c2$, are linear scaling factors which minimize variations between two classes among different training sets. In fact, d_c's, where $c \in \{Normal, Cancer\}$, are measures with respect to overall gene expression levels in various training sets, but the ratio of gene expression levels between two classes in individual training sets are varied. Hence, d_c's should be rescaled accordingly in order to reduce the impacts of differential ratios between the two classes among various data sets. As a result, individual decisions are insensitive to bias of either class and variations of gene expression levels among training sets.

For the ratio of two different *dis*'s, it weights the results of the majority-voting algorithm by taking the similarity of shapes between two histograms. Remind that candidate i in the set $C_{Hist,i}$ is defined as:

$$C_{Hist,i} = (c1 \mid dis(c1,s) < dis(c2,s) \wedge c1,c2 \in \{Normal, Cancer\} \wedge c1 \neq c2) \quad (8)$$

Hence, the factor of $dis(c1,s)/dis(c2,s)$ makes β_i become smaller, and thus a higher degree of similarity is contributed to meta-decisions because of similarity of the regular histograms.

In contrast, if the two decisions are different, the factors, representing the similarity of the regular histogram comparisons, are excluded. The factors of $(IF_{c1,i}/IF_{c2,i})$ aim at minimizing variations between classes and bias of either class. Therefore, the factors are also used to adjust the values of β_i. However, the factors of $dis(c1,s)/dis(c2,s)$ are weighted factors which give higher ranks to decisions because of similarity of the regular histograms. For the case of different decisions between two

algorithms, the previous method is not appropriate. In fact, the histograms are constructed by a set of significant genes, which are selected and extracted after the accession numbers alignment. Also, the significant genes are ranked in terms of their differential gene expression levels between two classes, which is independent on variations of gene expression levels among different data sets. Therefore, it is possible that (1) some significant genes are omitted during the accession numbers alignment, and (2) selected and extracted significant genes, based on training sets, may cause misleading results. As a result, we use another method and have the following equations for the case of different decisions:

$$d_{Normal} = \beta_i \times IF_{Cancer,\,i} / IF_{Normal,\,i} \quad (9)$$

$$d_{Cancer} = \beta_i \times IF_{Normal,\,i} / IF_{Cancer,\,i} \quad (10)$$

Finally, calculated d_{Normal} and d_{Cancer} are adjusted on log2 scale, and individual results corresponding to their training sets are added together, expressed as d_{Acc_normal} and d_{Acc_cancer} for normal and cancer classes. Their magnitudes are compared, and the classes with smaller magnitudes become meta-decisions of the testing sample.

Table 1. Information of data sets.

Data set ID	*Cancer type*	*Authors*	*Accession annotation*	*Normal sample size*	*Cancer sample size*	*Training data*	*Testing data*
1	Bladder	Ramaswamy et al. [18]	Hu35K	7	11		√
2	Brain	Pomeroy et al. [16]	Hu35K	4	10		√
3	Colon	Notterman et al. [14]	GenBank	4	4		√
4	Lung	Bhattacharjee et al. [1]	U95A	17	126	√	√
5	Lung	Ramaswamy et al. [18]	Hu35K	7	8	√	√
6	Ovary	Welsh et al. [25]	Hu35K	3	30		√
7	Prostate	Singh et al. [22]	U95A	9	25	√	√
8	Prostate	Welsh et al. [24]	U95A	50	52	√	√
9	Prostate	Ramaswamy et al. [18]	Hu35K	9	10		√
10	Uterus	Ramaswamy et al. [18]	Hu35K	6	10		√

Table 2. Number of common genes between training and testing data sets.

		Testing data set ID									
		1	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*
Training data set ID	*4*	7091	6153	6045	12599	7091	6153	12249	12599	12249	7091
	5	13774	8391	7840	7091	13774	8391	6808	7091	6808	13774
	7	6808	5949	5841	12249	6808	5949	12625	12249	12625	6808
	8	7091	6153	6045	12599	7091	6153	12249	12599	12249	7091

Table 3. Experimental results compared wit the majority-voting meta-classification.

Testing set ID	*Type*	*Approach*	*Accuracy (%)*	*Sensitivity (%)*	*Specificity (%)*	*Cost of learning savings*
1	Bladder	Majority-voting	73.61±9.49	39.29±31.68	95.45±5.25	5±3.92
		MIF algorithm	84.72±2.78	60.71±7.14	100.00±0	8.5±1
2	Brain	Majority-voting	75.00±7.14	25.00±35.36	95.00±5.77	1.5±2.38
		MIF algorithm	83.93±3.57	68.75±12.5	90.00±8.16	4.5±0.58
3	Colon	Majority-voting	87.50±0	75.00±0	100.00±0	6±0
		MIF algorithm	87.50±0	75.00±0	100.00±0	6±0
4	Lung	Majority-voting	96.50±0.81	94.12±0	96.83±0.92	28±1.15
		MIF algorithm	94.76±1.21	97.06±5.88	94.44±1.71	26±1.83
5	Lung	Majority-voting	75.00±3.21	42.86±20.2	95.45±9.09	5.5±1.91
		MIF algorithm	91.67±3.56	85.71±0	95.45±9.09	11.5±1
6	Ovary	Majority-voting	80.30±5.25	0.00±0	88.33±5.77	-3.5±1.73
		MIF algorithm	84.85±2.47	33.33±0	90.00±2.72	-1±0.82
7	Prostate	Majority-voting	100.00±0	100.00±0	100.00±0	18
		MIF algorithm	96.32±2.82	91.67±5.56	98.00±2.31	16±1.41
8	Prostate	Majority-voting	63.16±11.37	33.33±39.54	90.00±14.14	5±5.72
		MIF algorithm	57.11±5.85	15.50±12.58	97.12±1.11	14±12.25
9	Prostate	Majority-voting	75.00±3.21	42.86±20.2	95.45±9.09	5.5±1.91
		MIF algorithm	68.42±11.37	52.78±29.22	82.50±15	7.75±4.57
10	Uterus	Majority-voting	81.25±5.1	66.67±23.57	90.00±11.55	7±2
		MIF algorithm	81.25±0	75.00±9.62	85.00±5.78	7.5±0.58

4. EXPERIMETNS & DISCUSSIONS

To measure the classification performance, four measurements are used as performance indicators. Classification accuracy, sensitivity, specificity and learning cost savings are defined in terms of true positive (*TP*), true negative (*TN*), false positive (*FP*) and false negative (*FN*), and their definitions are [4], [13]:

- Accuracy (acc) – $acc=(TP+TN)/(TP+TN+FP+FN)$
- Sensitivity (S_n) – $S_n = TP/TP+FN)$
- Specificity (S_p) – $S_p=TN/(TN+FP)$
- Learning cost savings (sav) – $sav=[(FN+TP)*2]-(FP+2*FN)$

4.1. Data sets

In order to demonstrate the robustness of the MIF algorithm, 10 different data sets, which are individually published in 8 publications, are experimented. They are heterogeneous since they were conducted by different laboratories with different experimental objectives, microarray platforms and human genome arrays. Table 1 shows their information. Among all of them, two lung cancer (Bhattacharjee and Ramaswamy) and two prostate (Singh and Welsh) cancer data sets are arbitrarily selected as training data sets for extension and continuity of our previous works in [7], and all of them are used for testing.

As stated in table 1, there are three different accession numbers annotations, and therefore a process of standardization is required. We map the Hu35K and GenBank annotations into the U95A annotation according to the mapping table done by Ramaswamy et al. [17]. In fact, the mapping is not simply one-to-one mapping. There may be duplicated accession numbers in the mapped data set. Thus, an extra pre-processing step is performed to combine the expression levels by averaging all expression levels of the same accession numbers. After the standardization, it is required to find out those commonly existed genes for pairs of heterogeneous data sets and align their expression levels. In fact, the numbers of gene among different data sets are varied. Unavoidably, some expression levels are omitted because of missing data in either data set of pairs. Hence, the number of genes in aligned sets is either smaller or equals to the number of genes in the original data sets. Finally, we have table 2, which shows the number of commonly existed genes between training and testing data sets.

4.2. Results

In this section, we first compare the results of the MIF algorithm with that of the majority-voting algorithm, and then the results are compared with the works done by Bloom et al. [2]. Bloom's method is to perform multi-platform and multi-site microarray-based tumor meta-classification, and they used the measurement of classification accuracy as performance indicator. For parameters settings, the numbers of required bins n_b, and significant genes n_g, are set as 25 and 100. In addition, α%, which is the percentage of candidate bins to be trimmed, is set to 10% for achieving the optimal performance after some empirical studies. For classifiers training scheme, 70% of samples in each training data sets are selection for individual training at random, and all samples in testing data sets are used for performance measurements. In order to estimate the standard deviation of the performance, each training set is trained 100 times with different training candidates selected randomly.

In table 3, it shows that the MIF algorithm outperforms the majority-voting algorithm in terms of classification accuracy, sensitivity, specificity and cost of learning savings. Except for the cases of prostate cancer, the MIF algorithm achieves around 85% of accuracy, 65% of sensitivity, 90% of specificity and comparatively higher savings on learning cost.

For the classification accuracy, the data sets of lung cancer have the highest performance, but all cases of prostate cancer have little performance reduction. For lung cancer, the accuracy is higher than 90% for both cases (i.e. Bhattacharjee and Ramaswamy). Although there is 2% reduction for the data set Bhattacharjee, the accuracy for the data set Ramaswamy is increased from 75% to 91%. However, all data sets of prostate cancer have different degrees of performance degradation. There are reductions of 7%, 6% and 7% for the accuracy of the data set Singh, Welsh and Ramaswamy. In addition, it shows that two of them, which are Welsh and Ramaswamy, perform worse than other cases not only with the majority-voting algorithm, but also with the MIF algorithm. They only achieve around 60% for the accuracy, which is 20% lower than the average cases. For other cancer types, including bladder, brain, colon, and uterus, their average accuracy is around 85%. For the standard deviations of the accuracy, the MIF algorithm achieves smaller standard deviations for most cases. For the cancer types of bladder, brain, ovary and uterus cancers, the improvement is more than 50%. For the cancer types of lung and prostate cancers, the significance results are varied.

For the classification sensitivity and specificity, the MIF algorithm can have better balanced recall rates between normal and cancer samples, except for the cases of prostate cancer. Classification algorithms should have similar recall rates for samples in both classes so that the algorithms are unbiased to either class. Euclidean distance of sensitivity, S_n, and specificity, S_p, can be used to show the balance of recall rates between samples in two classes, and the distance is:

$$Euclidean(S_n, S_p) = \sqrt{S_n^2 + S_p^2} \qquad (11)$$

In table 4, it shows that the MIF algorithm outperforms the majority-voting algorithm for 6 cases (i.e. 1, 2, 5, 6, 8 and 10) and maintains the same performance for 2 cases (i.e. 2 and 3). Similar to the measurement of classification accuracy, the data sets of prostate cancer do not have impressive results. Testing set 7 and 9 show performance degradation (i.e. the majority-voting algorithm outperforms the MIF algorithm.).

Table 4. Balanced recall rates between normal and cancer sample.

Testing set ID	*Type*	*Majority-voting*	*MIF algorithm*
1	Bladder	1.03	**1.17**
2	Brain	0.98	**1.13**
3	Colon	**1.25**	**1.25**
4	Lung	**1.35**	**1.35**
5	Lung	1.05	**1.28**
6	Ovary	0.88	**0.96**
7	Prostate	**1.41**	1.34
8	Prostate	0.96	**0.98**
9	Prostate	**1.05**	0.98
10	Uterus	1.12	**1.13**

In addition, we have also compared our results with bloom's results in [2]. In table 5, it shows that the MIF algorithm outperforms Bloom's works for bladder and uterus cancers, and

maintains the same performance for lung cancer. However, there is performance reduction for prostate cancer.

Table 5. Comparison of results with other works.

Testing set ID	*Type*	*Classification accuracy (%)*	
		Bloom's results	*our results*
1	Bladder	77	**84**
5	Lung	**91**	**91**
9	Prostate	**94**	68
10	Uterus	74	**81**

4.3. Cancer/testis (CT) immunogenic gene families

Cancer/testis (CT) immunogenic gene families are subsets of genes, which are commonly existed in various cancer types. Some works show that most CT immunogenic gene families are expressed in more than one cancer types, but with various expression frequencies. In [20], Scanlan et al. have reviewed the expression frequencies of them in numerous cancer types consisting of bladder, brain, breast, colon, gastric, and etc. It shows that lung and melanoma cancers contain a higher percentage of CT genes examined at expression frequencies greater than 20%. In contrast, prostate and brain cancers have a relatively lower percentage of the CT genes examined at the same frequencies.

Table 6. Comparisons of the cancer/testis (CT) immunogenic gene families in various cancer types.

	Cancer type					
	Bla	Bra	Col	Lun	Ova	Pro
No. of included lowly-expressed CT genes with a low expression frequency, <= 20%	*17*	*5*	*12*	*29*	*11*	*11*
No of included CT highly-expressed genes with a high expression frequency, > 20%	*11*	*4*	*3*	*17*	*7*	*6*
Proportions of commonly existed highly-expressed genes to lung cancer	*7/11*	*3/4*	*2/3*	*29/29*	*5/7*	*2/6*
Proportions of commonly existed highly-expressed genes to prostate cancer	*4/11*	*1/4*	*1/3*	*2/29*	*2/7*	*6/6*

 # Abbreviations: Bla, bladder; Bra, brain; Col, colon; Lun, lung; Ova, ovary; Pro, prostate.

In our studies, we have analyzed how the proportions of shared highly-expressed CT genes between training and testing samples play a vital role in meta-classification performance of heterogeneous data. We investigated how the number of included lowly- and highly-expressed CT genes is varied with the classification performance. Table 6 shows the number of included lowly- and highly-expressed CT genes in various cancer types. Lung cancer has the highest proportions of both types of CT genes, and brain cancer has the lowest one. However, in [20], it has mentioned that the studies of brain cancer to the CT genes are insufficient in this moment. Therefore, brain cancer is exceptional and hence prostate and ovary cancers belong to the same family of having small proportions of both types of CT genes.

From our experiments, the data sets of prostate cancer only achieve classification accuracy of 75% in average, but the data set of ovary cancer can achieve 84% instead. Hence, it may be deduced that there is no direct and linear relationship between the number of included lowly- and highly-expressed CT genes and the classification performance.

Further, we have investigated how the number of shared highly-expressed CT genes between training and testing samples is in relation to the classification performance. In table 6, the last two rows show the proportions of the highly-expressed CT genes between the corresponding samples, and both lung and prostate cancers, respectively. If we consider the proportions together with the corresponding classification performance, we will have figure 6. In the figure, the classification performance has the same increasing and decreasing trends as the proportions of the CT genes to lung cancer, but reversed trends for the proportions to prostate cancer, except for the case of brain cancer.

Together with figure 6 and table 6, we can see that the proportions of shared highly-expressed CT genes between training and testing samples has impacts on classification performance, and the data sets of lung cancer have dominated roles at meta-decisions because of higher proportions of shared highly-expressed CT genes between training and testing samples. In the figure, lung cancer always has higher proportions of shared highly-expressed CT genes with other cancer types, except for the prostate cancer. The classification accuracy is higher than 80% in average. However, the classification accuracy for prostate cancer has been dropped significantly. It may be evidence to show that the decrease of the performance for prostate cancer is caused by lack of shared highly-expressed CT genes between training and testing

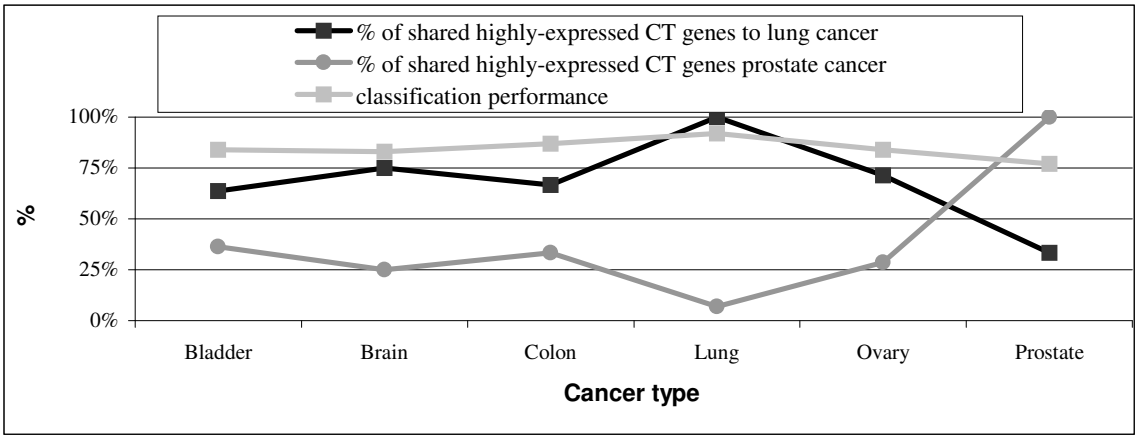


Figure 6. Relationship between shared highly-expressed CT genes and classification performance.

samples. Also, it is observed that the highly-expressed CT genes of prostate cancer in training samples are compromised with the lack of the genes between lung and prostate cancers. The possible explanation is that the number of included lowly- and highly-expressed CT genes in various cancer types. The fact is that the number of both included lowly- and highly-expressed CT genes to lung cancer is almost 3 times higher than that of prostate cancer, causing that data sets of lung cancer may have higher weights at meta-decisions. In addition, in figure 6, the decreasing rate of the performance for prostate cancer is less than that of the proportions of shared highly-expressed CT genes to lung cancer because of the increase of shared CT genes in prostate cancer (i.e. the ordinary type).

5. CONCLUSIONS

With the innovation of DNA microarray technologies, different mining algorithms have been proposed to discover knowledge in cancer gene expression data. Significant findings are recently exploited. However, most works are done with a single data set. In terms of efficiency and effectiveness of mining algorithms with respect to clinical applicability and robustness, it is too weak to draw conclusions because of the problems of over-fitting and homogeneity within a single data set.

In this work, we proposed the MIF algorithm to perform multi-type cancer gene expression data classification, which uses differences of regular histograms for gene expression levels of certain significant genes as parts of dissimilarity measures and indicators of predictive classes. In the experiments, we have intensively used 10 different data sets to show the reliability and robustness of the MIF algorithm. The results are impressive. The classification accuracy is around 85% in average for most cases, except for the data sets of prostate cancer.

To investigate the frustrated performance for prostate cancer, we have looked into the cancer/testis (CT) immunogenic gene families. We have discovered that the numbers of shared highly-expressed (i.e. expression frequencies > 20%) CT genes between training and testing samples have impacts on the classification performance of heterogeneous samples.

6. Acknowledgement

The work of the authors are supported in part by the Central Grant of The Hong Kong Polytechnics University, research project code HZJ89.

7. REFERENCES

[1] Bhattacharjee, A., Richards, W., Staunton, J., Li, C. and Monti, S. (2001). Classification of human lung carcinomas by mRNA expression profiling reveals distinct adenocarcinoma subclasses. Proc. of the Natl. Acad. of Sci. USA, 98(24), pp. 13790-5.

[2] Bloom, G., Yang, I.V., Boulware, D. and Kwong, K.Y. (2004). Multi-platform, multi-site, microarray-based human tumor classification. Am. J. Pathol., 164(1), pp. 9-16.

[3] Cho, S.B. and Ryu, J.W. (2002). Classifying gene expression data of cancer using classifier ensemble with mutually exclusive features. Proc. of the IEEE, 90(11), pp. 1744-1753.

[4] Cho, S.B. and Won, H.H. (2003). Machine learning in DNA microarray analysis for cancer classification. Proc. of the First Asia Pacific Bioinformatics Conference, Adelaide, Australia, 19, pp. 189-198: Australian Computer Society.

[5] Choi, J.K., Yu, U., Kim, S. and Yoo, O.J. (2003). Combining multiple microarray studies and modeling interstudy variation. Bioinformatics, 19 Suppl., 1:i84-90.

[6] Cui, X. and Churchill G.A. (2003). Statistical tests for differential expression in cDNA microarray experiments, Genome Biol., 4(4):210.

[7] Fung, B.Y.M. and Ng, V.T.Y (2003). Classification of Heterogeneous Gene Expression Data. Special Issue on Microarray Data Mining, SIGKDD Explorations, 5(2), pp. 69-78.

[8] Fung, B.Y.M and Ng, V.T.Y. (2004). Selecting the Optimal Classification Results by an Empirically-driven Model Averaging. Proc. of the SIAM Bioinformatics Workshop 2004, Florida, USA, pp. 25-35: SIAM.

[9] Kuo, W.P., Jenssen, T.K., Butte, A.J., Ohno-Machado, L. and Kohane, I.S. (2002). Analysis of matched mRNA measurements from two different microarray technologies. Bioinformatics, 18(3), pp. 405-12.

[10] Lee, J.K., Bussey, K.J., Gwadry, F.G. and Reinhold, W. (2003). Comparing cDNA and oligonucleotide array data: concordance of gene expression across platforms for the NCI-60 cancer cells. Genome Biology, 4:R82.

[11] Lu, Y. and Han, J. (2003). Cancer classification using gene expression data. Information Systems, 28(4), pp. 243-268.

[12] Nadimpally, V. and Zaki, M. (2003). A Novel Approach to Determine Normal Variation in Gene Expression Data. Special Issue on Microarray Data Mining, SIGKDD Explorations, 5(2), pp. 6-15.

[13] Ng, S.K., Tan, S.H. and Sundarajan, V.S. (2003). On Combining Multiple Microarray Studies for Improved Functional Classification by Whole-Dataset Feature Selection. Genome Informatics, 14, pp. 44-53.

[14] Notterman, D.A., Alon, U., Sierk, A.J. and Levine, A.J. (2001). Transcriptional gene expression profiles of colorectal adenoma, adenocarcinoma, and normal tissue examined by oligonucleotide arrays. Cancer Res., 61(7), pp. 3124-30.

[15] Ochs, M.F. and Godwin, A.K. (2003). Microarrays in cancer: research and applications. Biotechniques, 34, Suppl., pp. 4-15.

[16] Pomeroy, S.L., Tamayo, P., Gaasenbeek, M. and Sturla, L.M. (2002). Gene Expression-Based Classification and Outcome Prediction of Central Nervous System Embryonal Tumors. Nature, 415, pp. 436-42.

[17] Ramaswamy, S., Ross, K.N., Lander, E.S. and Golub, T.R. (2003). Evidence for a molecular signature of metastasis in primary solid tumors. Nature Genetics, 33, pp. 49-54.

[18] Ramaswamy, S., Tamayo, P., Rifkin, R. and Mukherjee, S. (2001). Multi-class cancer diagnosis using tumor gene expression signatures. Proc. of the Natl. Acad. of Sci. USA, 98(26), pp. 15149-54.

[19] Rhodes, D.R., Barrette, T.R., Rubin, M.A., Ghosh, D. and Chinnaiyan, A.M. (2002). Meta-analysis of microarrays: interstudy validation of gene expression profiles reveals pathway dysregulation in prostate cancer. Cancer Res., 62(15), pp. 4427-33.

[20] Scanlan, M.J., Simpson, A.J.G. and Old, L.J. (2004). The cancer/testis genes: Review, standardization, and commentary. Cancer Immunity, 4(1), pp. 1-15.

[21] Seewald, A.K. and Frnkranz, J. (2001). An evaluation of grading classifiers. Proc. of the 4th Int. Symposium in Advances in Intelligent Data Analysis (IDA-01), Lisbon, Portugal, pp. 115-124: Springer-Verlag.

[22] Singh, D., Febbo, P.G., Ross, K. and Jackson, D.G. (2002). Gene expression correlates of clinical prostate cancer behavior. Cancer Cell, 1(2), pp. 203-209.

[23] Van der Bruggen, P., Traversari, C., Chomez, P. and Lurquin, C. (1991). A gene encoding an antigen recognized by cytolytic T lymphocytes on a human melanoma. Science, 254(5038), pp. 1643-7.

[24] Welsh, J.B., Sapinoso, L.M., Su, A.I. and Kern, S.G. (2001). Analysis of gene expression identifies candidate markers and pharmacological targets in prostate cancer. Cancer Research, 61, pp. 5974-78.

[25] Welsh, J.B., Zarrinkar, P.P., Sapinoso, L.M. and Kern, S.G. (2001). Analysis of gene expression profiles in normal and neoplastic ovarian tissue samples identifies candidate molecular markers of epithelial ovarian cancer. Proc. Natl. Acad. Sci. USA, 98(3), pp. 1176-81.

[26] Yao, X. and Liu, Y. (1999). Neural networks for breast cancer diagnosis. Proc. of the 1999 Congress on Evolutionary Computation, New York, USA, pp. 1760-1767: IEEE Press.

Bayesian Model-Averaging in Unsupervised Learning From Microarray Data

Mario Medvedovic
University of Cincinnati Med. Ctr
Department of Environmental Health
Cincinnati, OH 45221-0056
Mario.Medvedovic@uc.edu

Junhai Guo
University of Cincinnati Med. Ctr.
Department of Environmental Health
Cincinnati, OH 45221-0056
guojs@ucmail.uc.edu

ABSTRACT

Unsupervised identification of patterns in microarray data has been a productive approach to uncovering relationships between genes and the biological process in which they are involved. Traditional model-based clustering approaches as well as some recently developed model-based mining approaches for integrating genomic and functional genomic data rely on one's ability to determine the correct number of clusters or modules in the data. In this paper we demonstrate that the performance of such methods in general can be significantly improved by accounting for uncertainties inherent to the process of identifying the optimal number of clusters in the data. We demonstrate that the Bayesian averaging approach to clustering via infinite mixture model offers a more robust performance than the traditional finite mixture model in which the optimal number of clusters is determined using the Bayesian Information Criterion. This performance improvement is demonstrated through a simulation study and by the analysis of a relatively large microarray dataset. Finally, we describe the novel heuristic modification of the Gibbs sampler used to fit the infinite mixture mode that effectively deals with issues of slow mixing.

Keywords

Guides, instructions, authors kit, conference publications.

1.INTRODUCTION

Unsupervised identification of patterns in microarray data has been a productive approach to uncovering relationships between genes and the biological process in which they are involved. Conceptually, unsupervised learning from microarray data can be done by identifying genes with similar expression patterns across different experimental conditions, identifying groups of experimental conditions or biological samples with similar expression profiles, or the two dimensional clustering that simultaneously clusters genes and biological samples. In this paper we will be talking mostly about identifying groups of genes with similar expression patterns (profiles) across different biological samples. Groups of such genes are said to be co-expressed and they define patterns of expression. The utility of identifying such groups of co-expressed genes is in the assumption that the co-expression is a reflection of a shared regulatory mechanism driving similarities of expression profiles. Consequently, such groups of genes can be used as a starting point for dissecting expression regulatory mechanisms [23], or functional annotation by assuming that functionally-related genes are most likely to be co-regulated [5].

Clustering methods used for unsupervised identification of co-expressed genes can be loosely grouped into heuristic methods based on various distance measures, and model-based methods which are based on the probabilistic model of the data generation process. Given a distance measure, various heuristic methods proceed to organize gene expression profiles in a hierarchical fashion [3] or by partitioning them into a pre-specified number of clusters of co-expressed genes (e.g. K-means algorithm and Self-organizing maps).

In a model-based approach to clustering, the probability distribution of the observed data is approximated by a probabilistic model. Parameters in such a model define clusters of similar observations and a cluster analysis is performed by estimating these parameters from the data. The Finite Mixture (FM) model is the most common model-based approach to clustering [11]. In the context of microarray data, the FM model was introduced by [24]. In this approach, similar individual profiles are assumed to have been generated by a common underlying "pattern" represented by a multivariate Gaussian random variable. Given the correct number of mixture components (clusters) one can use an EM algorithm to estimate parameters of this model and then use the parameter estimates to assign individual profiles to appropriate clusters. Recently, various generalizations of the Bayesian mixture approach in terms of sophisticated Bayesian probabilistic models have been used to integrate various pieces of additional information in the process of identifying co-expressed genes [21;22], and to identify "modules" of co-regulated genes through the integrated modeling of combinatorial regulation mechanisms and gene expressions via context-specific Bayesian networks [19].

The common denominator of the above-mentioned model-based methods is that they rely on the prior specification of the number of clusters in the data or on one's ability to determine the correct number of clusters from the data. When the correct number of clusters is determined in the data analysis (e.g. by calculating Bayesian Information Criterion – BIC for models with different number of clusters), uncertainties related to its selection are generally not taken into account in the subsequent analysis. Previously we described the Bayesian Infinite Mixture (IM) model for the clustering of gene expression profiles [12] which effectively circumvents the problem of identifying the "correct" number of clusters. In our approach, the clusters are formed based on the posterior distribution of clusterings, which is generated by a Gibbs sampler. The clusterings generated by the Gibbs sampler can vary from one cycle to the next. Consequently, posterior probabilities with various features of the posterior distribution of clusterings are obtained after averaging over models with all possible number of clusters.

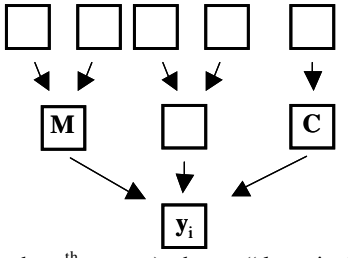
In this paper we describe a new simulated annealing-motivated algorithm for sampling from the posterior distribution of clusterings that effectively solves the severe mixing problem exhibited by Gibbs sampler in high-dimensional situations. More

importantly, we demonstrate dramatic positive effects that Bayesian averaging can have on discovering patterns in microarray data through both a simulation study and the analysis of a relevant real-world microarray dataset. These results are likely to bear on further development of model-based unsupervised learning methods that rely on either the specification of the correct number of clusters or its estimation from the data.

2.FINITE AND INFINITE MIXTURES MODEL BASED CLUSTERING FOR MICROARRAY DATA

Suppose that T gene expression profiles were observed across M experimental conditions. If y_{ij} represents the expression measurement for the i^{th} gene under j^{th} experimental condition then $\mathbf{y}_i=(y_{i1},y_{i2},\ldots,y_{iM})$ represents the expression profile for the i^{th} gene. In a mixture model, each gene expression profile is viewed as being generated by one out of Q different underlying expression patterns. Expression profiles generated by the same pattern form a cluster of similar expression profiles. If c_i is the classification variable indicating the pattern that generates the i^{th} mean expression profile (c_i=q means that the i^{th} expression profile was

Figure 1: Bayesian mixture model



generated by the q^{th} pattern), then a "clustering" is defined by a set of classification variables for all genes $\mathbf{C}=(c_1, c_2,\ldots, c_T)$. Underlying patterns generating clusters of expression profiles are represented by multivariate Gaussian random variables. Profiles clustering together are assumed to be a random sample from the same multivariate Gaussian distribution.

The hierarchical structure of the model is described in terms of a Directed Acyclic Network in **Figure 1**. Nodes (squares) in this diagram represent random variables and directed arcs (arrows) specify conditional dependences between variables in terms of the directed Markov property, which states that a variable is conditionally independent of its non-descendants given its parents in the model. $\mathbf{M}=(\boldsymbol{\mu}_1,\ldots, \boldsymbol{\mu}_Q)$ and $\boldsymbol{\Sigma}=(\sigma_1^2\mathbf{I},\ldots, \sigma_Q^2\mathbf{I})$ denote means and variance-covariance matrices of multivariate Gaussian random variables defining Q underlying patterns respectively ($\mathbf{I}$ denotes the identity matrix). Variables $(\boldsymbol{\lambda}, \tau)$, (β,ϕ), and α are hyper-parameters in prior distributions of model parameters **M, Σ** and **C** respectively. In the case of an FM model, the number of mixture components (Q) is considered fixed, while the IM model represents the limiting case when $Q\to\infty$. Details of the development of IM models and their relationship to mixtures with a Dirichlet process prior [4] are described elsewhere [15;17]. We have previously described Bayesian versions of both finite and infinite mixtures and corresponding Gibbs samplers [12;14]. In this paper, finite mixtures model were treated from a frequentist perspective and estimated using the EM algorithm as implemented in the MCLUST software [6]. The Gibbs sampler for estimating the posterior distribution of clusterings in the IM model is described below. The specification of the prior distribution for classification variables (**C**) determines whether the model represents finite or infinite mixtures.

2.1Gibbs Sampler

Gibbs sampler [7] is a general procedure for sampling observations from a multivariate distribution. A Gibbs sampler proceeds by iteratively drawing observations from complete posterior conditional distributions of all components. As the number of iterations approaches infinity, such a sequence describes observations from the joint multivariate distribution. In our case, we use the Gibbs sampler to estimate the joint posterior distribution of all parameters in our hierarchical model, given the data. We then use the marginal posterior distribution of clusterings to calculate posterior pairwise probabilities of coexpression (PPPC) for all pairs of expression profiles. Suppose that the sequence of clusterings ($\mathbf{C}^B$, $\mathbf{C}^{B+1}$,.., $\mathbf{C}^S$) was generated by the Gibbs sampler after B "burn-in" cycles. The pair-wise probabilities for two genes to be generated by the same pattern are estimated as:

$$P_{ij}=\frac{\#\text{ of samples after "burn-in" for which } c_i=c_j}{S-B}.$$

Using these probabilities as a similarity measure, clusters of similar expression profiles are created using a traditional agglomerative hierarchical clustering with similarities between groups of profiles being defined using the complete linkage. Complete descriptions of the posterior conditional distributions used by the Gibbs sampler can be found in [12], with the slight modification of using an independent, equal variance, covariance structure while in the original model we used the different variance elliptical model.

2.2Convergence of the Gibbs sampler

Two aspects of the Gibbs sampler convergence that generally need to be assessed are the appropriateness of the "burn-in" period, after which a Gibbs sampler has attained its stationary distribution, and the mixing of the sampler, which describes how well a finite sample obtained by Gibbs sampler approximates the target distribution. It has generally been well documented that the simple Gibbs sampler often has very poor mixing properties in both FM and IM models [2;14], probably due to the multi-modality of the posterior distribution. In such a situation, the sampler will be unable to describe the whole posterior distribution in a computationally feasible number of steps. The sampler will get trapped in a sub-optimal mode of the posterior distribution resulting in sub-optimal clustering results; or, because the sampler fails to visit all areas with significant posterior probabilities, confidence estimates in the generated clustering will be biased. Previously, we described a heuristic algorithm for "heating up" the Markov chain described by the Gibbs sampler by using "reverse annealing." The optimal annealing schedule was chosen based on running a significant number of independent chains with different maximum annealing constants. Here we describe a new heuristic algorithm that adjusts the annealing exponent dynamically. Consequently, only a single run is needed to estimate the posterior distribution.

If $\pi(.)$ is the target posterior distribution, "reverse annealing" refers to "flattening" of the posterior distribution using the transformation $\pi^{(\xi)}(x)=\frac{\pi^{\xi}(x)}{K(\xi)}$, $\xi<1$, where $K(\xi)$ is the

normalizing constant. Based on this general idea, if $p(c_i=j|C_{-i}, \Theta)$ is the conditional posterior probability of placing the i^{th} profile into the j^{th} cluster then "flattened probabilities" are defined as

$$p(c_i = j \mid C_{-i}, \Theta)^{(\xi)} = \frac{p(c_i = j \mid C_{-i}, \Theta)^{\xi}}{K(\xi)}, \quad \xi < 1.$$

Since the mixing problem with the Gibbs sampler for the IM model can be particularly pronounced in its inability to generate new clusters, we keep track of the posterior probability of placing a profile in a new cluster. If this probability p_{new} is below the given threshold p_{min}, we decrease ξ by the value ξ_{step}. If p_{new} is above p_{min}, we increase ξ by ξ_{step}. Possible values of ξ are further constrained by the requirement that $0<\xi_{min}<\xi<\xi_{max}\leq 1$. Our modified Gibbs sampler now proceeds by generating n_{cold} samples from the unmodified conditional posteriors (cold cycles). It then generates a single sample using "heated" classification probabilities (heated cycle). The p_{new} from the heated cycle is used to increase or decrease the value of ξ by ξ_{step}. However, only the sample from the last cold cycle (n_{cold} cycles after the heated cycle) is used in the estimation of the posterior distribution of clusterings. In our simulations, we used $n_{cold}=5$, $\xi_{min}=0.1$, $\xi_{max}=1$, $p_{new}=0.01$ and $p_{step}=0.1$. Due to the high computational complexity in the analysis of the cancer data, we used $n_{cold}=3$.

2.3Finite mixture model and EM algorithm

We used the MCLUST package's EMclust procedure to fit finite mixture models to our simulated and real-world data sets. The optimal number of clusters was selected by calculating the Bayesian Information Criterion (BIC) [18] for models for different number of clusters. The only model used in this study was the equal variance, independent, spherical shape (EII) covariance model [6]. The EMclust procedure fits the finite mixture model by first performing an appropriate model-based hierarchical clustering (in the case of the EII model this amounts to the Ward's clustering algorithm). Resulting parameters are used as a starting point for the EM algorithm. Given the maximum likelihood estimates of the model parameters, profiles are assigned to clusters based on the posterior probabilities being generated by different mixtures components and the Maximum A Posteriory (MAP) hypothesis. To compare the performance of the FM model to the IM model, we also calculate FM model based PPPC's as

$$P_{ij} = \sum_{k=1}^{Q} p(c_i = k)p(c_j = k),$$

where $p(c_i=k)$ is the posterior probability of the profile i being generated by component k.

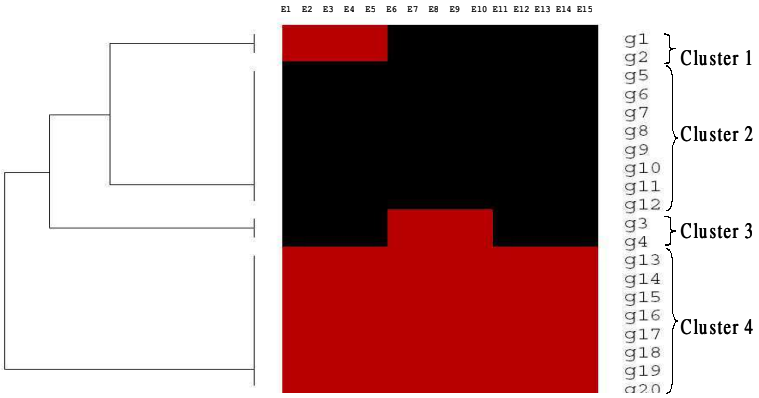
3.SIMULATION STUDY

First, we assessed the importance of Bayesian model-averaging in a simulation study. The study was designed to assess the performance of both FM and IM models in the frequentist sense. That is, we assessed the power of the two clustering methods to separate two different clusters in repeated experiments. We simulated 100 datasets each representing the clustering structure depicted in **Figure 2**. The heat map represents the values of the mean vectors for mixture components generating each profile. Red represents the value of 1 and the black represents the value of 0. For example, in each dataset, profile "g1" was randomly drawn from the 15-dimensional Gaussian random distribution whose mean vector is equal to 1 in first 5 dimensions ($e_1,..,e_5$) and 0 in other 10 dimensions ($e_6,..,e_{15}$). The covariance matrix $\sigma^2\mathbf{I}$ was used so that the data is compatible with our model assumptions. Data was simulated for $\sigma \in c$. This range allowed us to assess the performance of the two approaches in easy and progressively more difficult (i.e. noisier) situations.

3.1Results

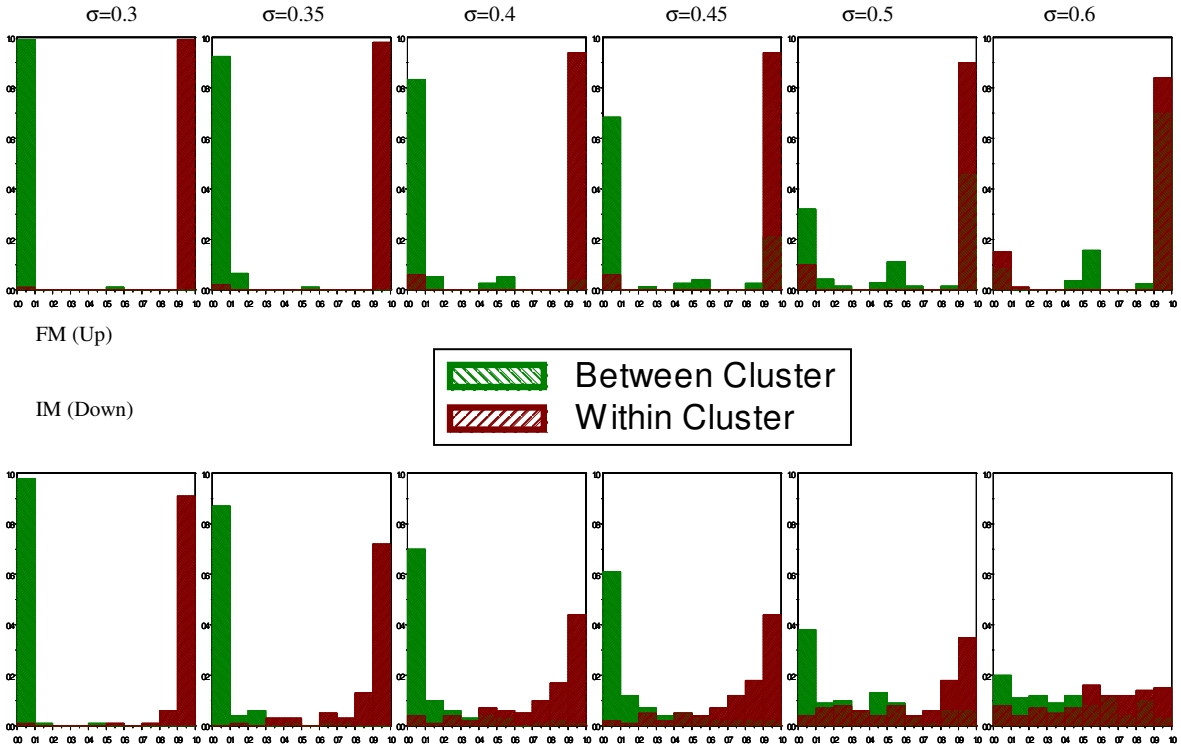
Both methods performed very well in separating two larger and most divergent clusters (Cluster2 and Cluster4) under the conditions of our simulation study. Therefore, we are focusing on the more difficult task of separating clusters 1 and 2. Profiles from these two clusters differ only within first 5 dimensions ("experiments") and Cluster1 is defined by only two profiles. The major question we are asking is how often can we expect the two clusters to be separated. We are assessing this question by observing the distribution of PPPC's for the two profiles in Cluster1 in relation to PPPC's between profiles in Cluster1 and Cluster2. In a sense we are assessing the ability of our clustering methods to correctly conclude that profiles in Cluster1 are different from profiles with Cluster2. However, unlike traditional statistical hypothesis testing procedures, we do not supply the labels for profiles that we are comparing.

Figure 2: Heat map of the clustering structure for the simulated data. Total of 20 15-dimensional profiles belonging to 4 unbalanced clusters are generated in each dataset.



Results of this simulation study support our thesis that FM model-based clustering in which the number of clusters is chosen by the BIC criterion suffers because of its inability to incorporate in the results of the analysis the uncertainty inherent in the process of determining the number of clusters. Histograms in Figure 3 show the "over-confidence" of the FM-based PPPC's which is typical of a statistical analysis that fails to take into account all sources of uncertainty (i.e. variability). The majority of the PPPC's generated by the FM model are clustered around 0 and 1 indicating the high confidence in the separation or non-separation in all situations, even when they are wrong. For example, for the highest noise level, close to 70% of between cluster PPPC's are greater than 0.9 indicating high confidence in the false conclusion that these profiles belong to the same cluster. On the other hand, PPPC's seem to be more reflective of the level of evidence for separating the two clusters present in the data. While the level of confidence in the separation is being reduced as we move from the low-noise to high-noise data, the fraction of PPPC's offering a high confidence in the false conclusion remains low even in the noisiest situation.

Figure 3: Histograms of PPPC's for pairs of profiles belonging to the same cluster (Within Cluster) and pairs of profiles belonging to different clusters (Between Cluster) in 100 simulated datasets for 6 different noise levels.



We can further drive the analogy with traditional statistical hypothesis testing procedures by constructing Receiver Operating Characteristic (ROC) curves that assess the ability of a clustering method to correctly separate profiles from different clusters. We are again focusing of ability to separate profiles in Cluster1 from profiles in Cluster2. For a fixed cut-off point X, we consider that the clustering procedure is correctly concluding that a profile i from Cluster1 does not belong to Cluster2 if $\max\{p(c_i=c_j$ for all profiles j from Cluster2$\}<X$. We consider that the clustering procedure is incorrectly concluding that profile 1 and profile 2 from Cluster1 do not belong in the same cluster if $p(c_i=c_j)<X$. The true positive rate (TPR) is the proportion of times that a correct decision is made and the false positive rate (FPR) is the proportion of times that an incorrect decision is made. As the cut-off X is increased from 0 to 1, both TPR and the FPR will increase. The area under the curve relating the TPR and FPR as X is increased from 0 to 1 describes the efficiency of a statistical procedure with the random decision-making having an area of 0.5 while the ideal statistical procedure would have an area equal to 1. ROC's for the FM and IM models for different noise levels are given in Figure 4. It seems that for each, except the lowest noise level, the IM model significantly outperforms the FM procedure.

4.CANCER DATA ANALYSIS

While the simulation study seems to indicate the importance of model averaging in the model-based cluster analysis, the question remains whether these advantages make any difference in the context of analyzing real-world data. Demonstrating advantages of one clustering method over another in the context of real-world data is complicated by the uncertainties related to the "correct" clustering which is generally not known. To address this question we reproduced the analysis described by [10]. They demonstrated how human cancer databases of microarray data could be used to study a molecular mechanism of cancer induction. In their study, they first identified 21 cyclin D1 target genes in *in-vitro* laboratory experiments. They followed up with an investigation of the relationship between CD1 and these 21 genes in a cancer gene expression database [16]. The statistical significance of that association in the cancer data was established by showing that the distribution of Euclidian distances between

expression profiles of these gene and CD1 were higher then expected by chance (p-value=0.048 using a resampling version of the Kolomogorov-Smirnov test). The conclusion was that the in-vitro signature of the CD1 overexpression is preserved in primary human tumors. We clustered the cancer expression data using the Euclidian distance, IM and FM models with the optimal number of clusters (56), elected by the BIC as described before (Figure 5). Based on results of these cluster analyses, two important points can be made: (1) just by a visual inspection of heat maps, it is apparent that model-based clustering approaches (both FM and IM) created "deaner" groupings of genes with similar expression patterns than the Euclidian distance-based hierarchical clustering procedure. (2) the over-confidence of the FM model, noted in the analysis of the simulated data, is evident in this analysis as well. The consequence of such an over-confidence is that the FM model identifies only 2 of the 21 genes of interest to be significantly associated with CD1. On the other hand, the IM model identifies 7 of them.

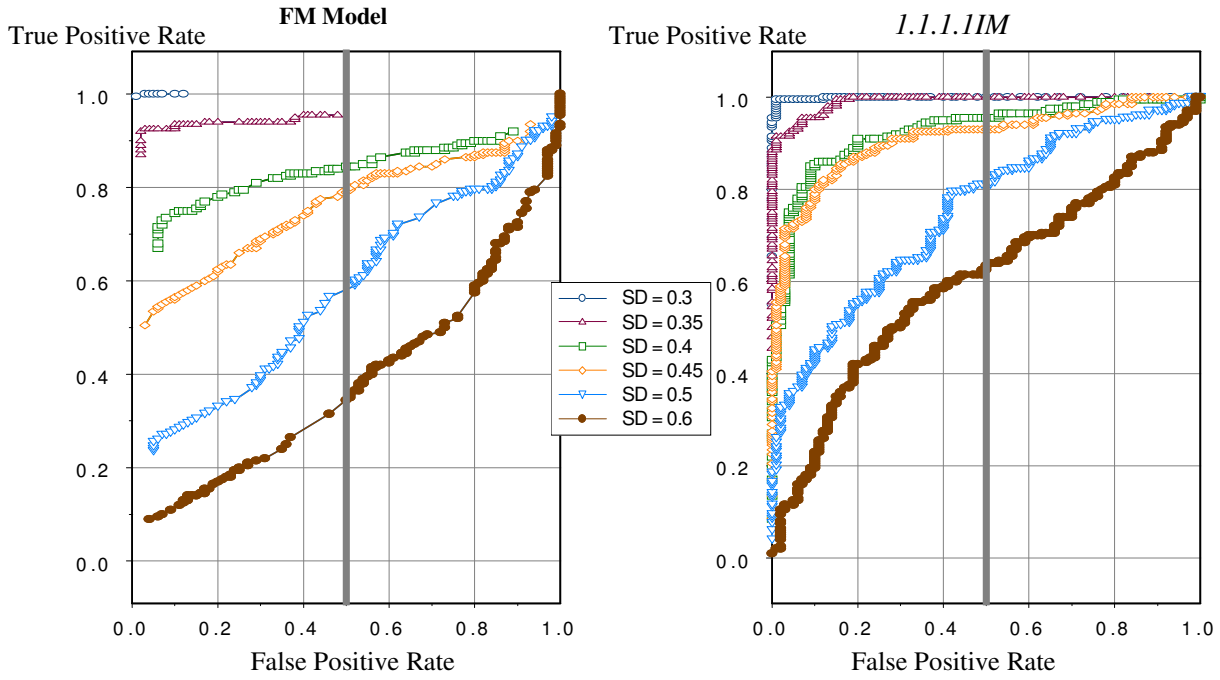
The distribution of the Euclidian distances also seems to suggest that only the associations between CD1 and three of the 21 genes of interest are above the experimental noise. In the context of the Kolmogorov-Smirnov (KS) analysis of Euclidian distances (Figure 3A in Lamb et al. 2003), there are indications that actually 7 to 10 of the 21 genes are contributing to the significance of the association. However, the statistical significance of this observation is impossible to assess within their framework. These results suggest that the IM model is capable of identifying most biologically meaningful relationships in the data by integrating the power of the model-based approach to pull information from the whole dataset while accounting for the uncertainty introduced by not knowing the number of clusters in the data.

5.DISCUSSION

In this paper we demonstrated the utility of Bayesian model averaging in model-based clustering of microarray data in a simulation study and as it is applied to answer a relevant biological question using a relatively large microarray dataset. We demonstrated that the performance of the traditional finite mixture clustering approach in which the optimal number of clusters is chosen using the BIC suffers from over-confidence in false conclusions probably due its inability to account for uncertainties related to the choice of the right number of clusters. The significantly better performance of the equivalent IM model in both the simulation study and the analysis of the real-world data is most likely due to its ability to estimated the posterior distribution of clusterings by effectively averaging over models with all possible number of clusters. The consistency and the precision of the results obtained by the IM approach also suggest that our heuristic modification of the Gibbs sampler effectively alleviates the problem of slow mixing.

We have previously demonstrated the advantages of using model-based clustering approaches over the traditional distance-based heuristic algorithms [14;25]. Model-based methods allow for the precise treatment of the statistical characteristics of the data under investigation, such as replicated observations. Furthermore, when compared to traditional distance measure-based hierarchical clustering algorithms, they are more efficient in using the information from the whole datasets instead of using two vectors of observations at a time. This advantage has been nicely demonstrated in our analysis of cancer data in Figure 5 as well. Additionally, when compared to partitioning heuristic algorithms, such as the K-mean algorithm and the SOM's, they allow for estimation of the number of clusters by assessing the relative fit of models with different numbers of clusters.

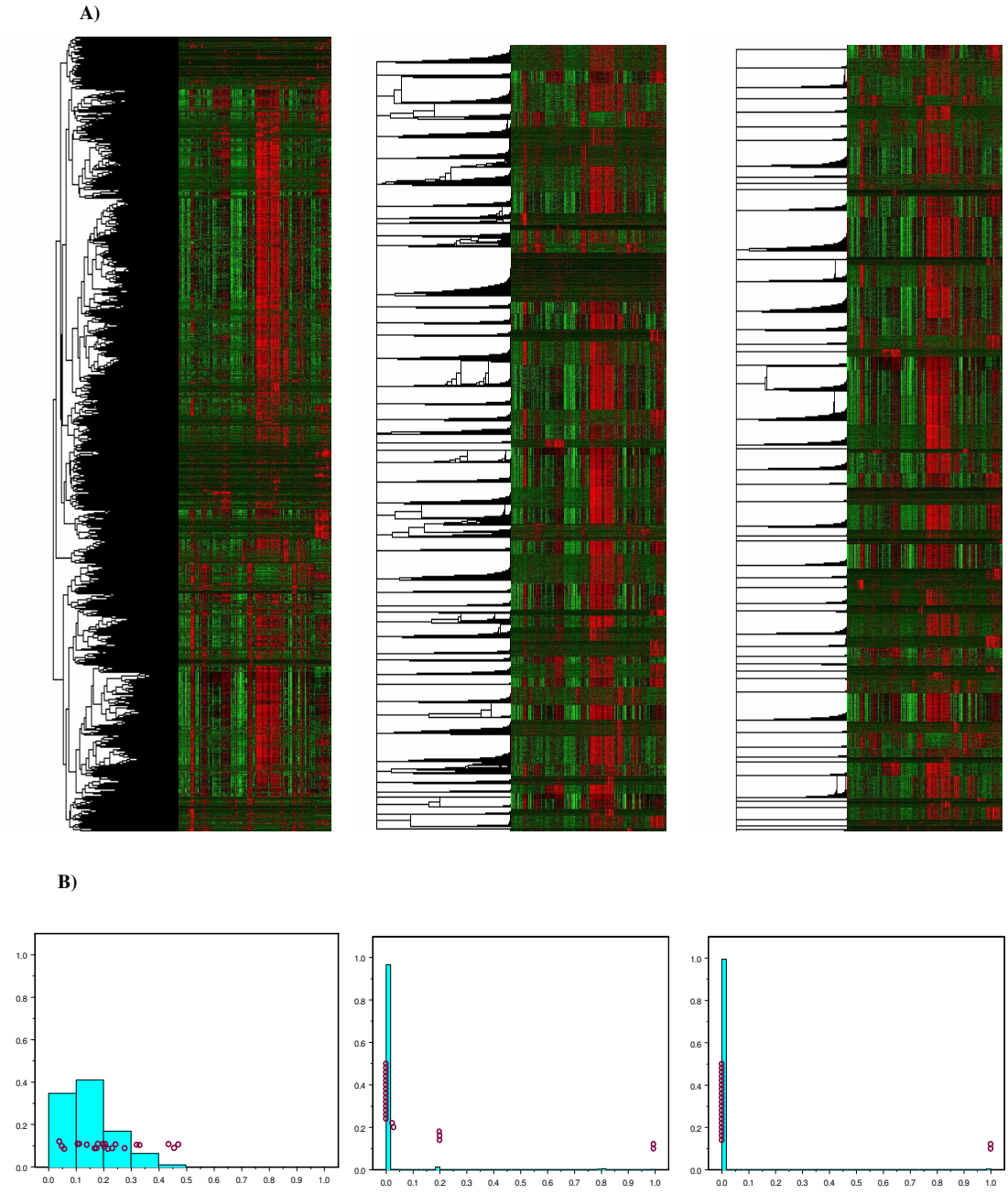
Figure 4: ROC curves describing the ability of the two models to separate the Cluster1 from Cluster2.



Recently introduced generalizations of the traditional mixture models, based on the context-specific Bayesian networks [20], allow for identifying more complex relationships between

solution for this problem could be the adaptation of the IM paradigm for such complex models. Another possible solution could go along the line of averaging results obtained by fitting

Figure 5: A) Cluster analysis based on the Euclidian distances (left), IM model PPPC's (middle), and FM model PPPC's (right). B) Histograms of corresponding similarity measures for all genes with CD1. Circles represent the similarity measures for the 21 genes identified in the laboratory experiments.



different genes as well as incorporating other types of the data in the analysis [21;22]. In this respect, our analysis strongly suggests that the general practice of fixing the number of clusters, components, or modules in terms of [19], before fitting appropriate models might need some modifications. One possible

models with different number of components in a post-hoc analysis.

It is important to notice that the uncertainties in the process of identifying the correct number of clusters are not necessarily the only source of uncertainties that are not taken into account by

the traditional FM approach. In high-dimensional situations, such as the cancer data analyzed in this paper, the log-likelihood maximized by the EM algorithm is almost certainly multi-modal and using any kind of strategy for choosing the "optimal" starting position will not guarantee that the solution will be globally optimal. Since the BIC calculation is based on results of the EM algorithm, these types of computational inadequacies will contribute to the overall uncertainty in the selection of the "optimal" number of clusters. Furthermore, such computa tional problems can result in sub-optimal clustering given the "optimal" number of cluster. Using different variants of the EM algorithm designed to alleviate this problem [13] can sometimes help, but the convergence to the globally optimal solution is still never guaranteed. In this respect, a properly mixing Gibbs sampler can offer another advantage due to its ability to describe the whole posterior distribution instead of searching for the highest mode of the likelihood function. We performed a limited evaluation of the convergence properties of the overall estimation approach (data not shown) and determined that EM convergence issues were probably not a factor in our simulation study due to the relatively simple clustering structure, but they were likely an additional source of uncertainty in the analysis of the cancer data.

The purpose of our analysis was not to disparage the BIC as the criterion for choosing the right number of clusters, but rather to demonstrate the problem of the whole approach in which the right model is chosen based on a preliminary analysis of the data, and where the uncertainties inherent in this process are not propagated into the final estimates of uncertainties about conclusions made based on the whole analysis process. Empirical studies have shown that the criterion works quite well in identifying the correct number of mixture components [1]. On the other hand some recent evaluations showed that an alternative approach of statistical hypothesis testing-based determination of the number of clusters [8] is more robust with respect to the deviation from the assumption of the models for individual mixture components. Unfortunately, these evaluations were made assuming only the simplest possible model for the calculation of the BIC, as implied by the K-means algorithm. It remains unclear if these advantages persist after using the complete FM approach for choosing the right covariance structure as well as the right number of clusters as proposed by the authors of MCLUST [6], or in the situation when the basic covariance structure implied by the K-means algorithm is correct, as was the case in our simulation study. Altogether, the BIC approach remains one of the dominant criteria for choosing models in statistical practice, and it is not clear that any alternative method for choosing the right number of clusters will significantly improve the overall FM performance. On the other hand, we showed that the IM model offers an elegant way around the issue of selecting the right number of clusters in the context of model-based clustering.

Finally, although our heuristic Gibbs sampler modification has been performing very well in all situations we encountered so far, it is not clear how closely does the modified sampler approximate the posterior distribution defined by the IM model. This is problematic since some of the nice conceptual features of the Bayesian IM framework depend on being able to sample from the true posterior distribution defined by the model. For example, the meaning of the posterior pairwise probabilities is not clear unless we can claim that they are derived from the hierarchical statistical model in Figure 1. We can still use them as a high-quality distance measure, but their direct probabilistic interpretation is lost. Some work has been done on developing alternative MCMC methods for fitting conjugate infinite mixture models [9]. However, to the best of our knowledge, alternative MCMC samplers for non-conjugate models, such as the model described here, have not yet been developed.

6.ACKNOWLEDGMENTS

This work has been supported by the NHGRI research grant 1R21HG002849-01.

7.REFERENCES

[1] C. Biernacki and G. Govaert. Choosing models in model-based clustering and discriminant analysis. J. Statis. Comput. Simul., 64 (1999), 49-71.

[2] G. Celeux, M. Hurn, and C. P. Robert. Computational and Inferential Difficulties With Mixture Posterior Distributions. JASA, 95 (2000), 957-970.

[3] M. B. Eisen, P. T. Spellman, P. O. Brown, and D. Botstein. Cluster analysis and display of genome-wide expression patterns. Proc. Natl. Acad. Sci. U. S. A, 95 (Dec.1998), 14863-14868.

[4] T. S. Ferguson. A Bayesian analysis of some nonparametric problems. The Annals of Statistics, 1 (1973), 209-230.

[5] S. Fessele, H. Maier, C. Zischek, P. J. Nelson, and T. Werner. Regulatory context is a crucial part of gene function. Trends Genet., 18 (Feb.2002), 60-63.

[6] C. Fraley and A. E. Raftery. Model-Based Clustering, Discriminant Analysis, and Density Estimation. JASA, 97 (2002), 611-631.

[7] E. A. Gelfand and F. M. A. Smith. Sampling-based approaches to calculating marginal densities. Journal of The American Statistical Association, 85 (1990), 398-409.

[8] G. Hamerly and C. Elkan. Learning the k in k-means. Proceedings of the Seventeenth Annual Conference on Neural Information Processing Systems (NIPS' 03) (2003),-

[9] S. Jain and R. Neal. A Split-Merge Markov Chain Monte Carlo Procedure for the Dirichlet Process Mixture Model. Technical Report No. 2003, Department of Statistics, University of Toronto (2000),-

[10] J. Lamb, S. Ramaswamy, H. L. Ford, B. Contreras, R. V. Martinez, F. S. Kittrell, C. A. Zahnow, N. Patterson, T. R. Golub, and M. E. Ewen. A mechanism of cyclin D1 action encoded in the patterns of gene expression in human cancer. Cell, 114 (Aug.2003), 323-334.

[11] J. G. McLachlan and E. K. Basford, *Finite Mixture Models: Inference and Applications to Clustering*. New York: Marcel Dekker, 1987.

[12] M. Medvedovic and S. Sivaganesan. Bayesian infinite mixture model based clustering of gene expression profiles. Bioinformatics, 18 (Sept.2002), 1194-1206.

[13] M. Medvedovic, P. Succop, R. Shukla, and K. Dixon. Clustering mutational spectra via classification likelihood and Markov Chain Monte Carlo Algorithm. Journal of Agricultural, Biological and Environmental Statistics, 6 (2001), 19-37.

[14] M. Medvedovic, Yeung K.Y., and R. E. Bumgarner. Bayesian mixture model based clustering of replicated microarray data. Bioinformatics (In Press) (2004),-

[15] R. M. Neal. Markov Chain Sampling Methods for Dirichlet Process Mixture Models. Journal of Computational and Graphical Statistics, 9 (2000), 249-265.

[16] S. Ramaswamy, P. Tamayo, R. Rifkin, S. Mukherjee, C. H. Yeang, M. Angelo, C. Ladd, M. Reich, E. Latulippe, J. P. Mesirov, T. Poggio, W. Gerald, M. Loda, E. S. Lander, and T. R. Golub. Multiclass cancer diagnosis using tumor gene expression signatures. Proc. Natl. Acad. Sci. U. S. A, 98 (Dec.2001), 15149-15154.

[17] C. A. Rasmussen. The Infinite Gaussian Mixture Model. Advances in Neural Information Processing Systems, 12 (2000), 554-560.

[18] G. Schwarz. Estimating the dimension of a model. Ann. Stat., 6 (1978), 461-464.

[19] E. Segal, M. Shpira, A. Regev, D. Pe'er, D. Koller, and N. Friedman. Module networks: identifying regulatory modules and their condition-specific regulators from gene expression data. Nat. Genet., 34 (2003), 166-176.

[20] E. Segal, B. Taskar, A. Gasch, N. Friedman, and D. Koller. Rich probabilistic models for gene expression. Bioinformatics, 17 (2001), S243-S252.

[21] E. Segal, H. Wang, and D. Koller. Discovering molecular pathways from protein interaction and gene expression data. Bioinformatics, 19 (2003), I264-I272.

[22] E. Segal, R. Yelensky, and D. Koller. Genome-wide discovery of trancriptional modules from DNA sequence and gene expression. Bioinformatics, 19 (2003), I273-I282.

[23] S. Tavazoie, J. D. Hughes, M. J. Campbell, R. J. Cho, and G. M. Church. Systematic determination of genetic network architecture. Nat. Genet., 22 (July1999), 281-285.

[24] K. Y. Yeung, C. Fraley, A. Murua, A. E. Raftery, and W. L. Ruzzo. Model-based clustering and data transformations for gene expression data. Bioinformatics., 17 (Oct.2001), 977-987.

[25] K. Y. Yeung, M. Medvedovic, and R. E. Bumgarner. Clustering Gene Expression Data with Repeated Measurements. Genome Biology, 4 (2003), R34-

Clustering Labeled Data and Cross-Validation for Classification with Few Positives in Yeast

Miles Trochesset and Anthony Bonner
University of Toronto
Department of Computer Science
10 King's College Rd.
Toronto, ON
M5S-3G4, Canada

{mtroches,bonner}@cs.toronto.edu

ABSTRACT

This paper presents two standard machine learning algorithms, one used in a non-standard way, for predicting the biological functions of essential genes in a systematic and comprehensive manner. We used gene expression and phenotype data from *Saccharomyces cerevisiae*. Determining gene function is simplified to a series of binary classification problems and one of the challenges of this learning task lies in the extremely small number of positives, compared with large amounts of negatives samples. We develop a method based on unsupervised hierarchical clustering used with labeled data to search for regions of high concentrations of positives and make predictions for the unlabeled genes. We also investigate the supervised logistic regression classifier as a baseline for comparing to our technique. Both of these methods are based on different views of the data and we found that depending on the biological processes being predicted, one or the other of these approaches performs better, although our method makes more confident predictions for more biological processes. The outcomes of the research are twofold: first we build a new biological data mining method based on existing machine learning tools that are readily accepted in the biological community. Second we make biological predictions of gene functions, each associated with a level of confidence and all above 50% precision.

1. INTRODUCTION

This paper investigates data mining and machine learning techniques for predicting, in a systematic and comprehensive manner, the possible functions of all putative and known genes (a gene may have several biological functions) in a yeast organism called *Saccharomyces cerevisiae*. We focused more intensely on making predictions for unlabeled genes, and decided to analyze the predictions of labeled genes in the future. Unlabeled genes are genes for which no function has yet been determined, whereas labeled genes are known to have at least one function. Systematic approaches for identifying the biological functions of genes, especially the unlabeled, are needed to ensure rapid progress from genome sequence to directed experimentation and applications (such as drug discovery).

The functions we learned are biological processes. Since relatively few genes are involved in a typical biological process, there are far more negatives than positives (as little as 0.01% of positives in the genome for certain biological processes), although some biological processes involve up to 60% of the genes in the genome. The learning task is made even harder by the fact that the samples we have comprise only about 10% of the genes in the genome (but required tremendous amounts of biological work to obtain nonetheless), 15% of which are unlabeled. So the number of positives available in our samples can be extremely small for some biological processes.

We examined two different methods based on two views of the data. The first view is that the positives and negatives can be separated by a hyperplane, which we fit using logistic regression. In the second view, the data constitutes as a sea of negatives with some small islands of positives of unknown size and number. We identify these concentrations of positives using hierarchical clustering on labeled data, which is not the standard unsupervised way of using this algorithm. We found that for some biological processes, one or the other method performs better, although our hierarchical method produces more confident predictions for more biological processes. Also, the method we develop allows the analysis of biological processes for which we have as little as 5 positive samples, unlike logistic regression which was unable to make predictions when the number of positives was below 20.

In this application, the cost associated with experimentally testing predictions lead us to performing leave-one-out cross-validation, not only to control how well the classifiers are behaving and draw ROC curves, but really to build decision rules for classifying samples. This is a main point in our methodology and we will explain it's details later.

Our analysis uses two types of data, gene expression from cDNA microarrays and growth phenotype data. Whole-genome expression profiling, facilitated by the development of DNA microarrays [12; 21], represents a major advance in genome-wide functional analysis. A single assay can measure the transcriptional response of thousands of genes, and often a full genome, to a change in cellular state such as disease, cell-cycle, cell division, response to stress and chemical compounds, or genetic perturbation and mutations. The scientific community agrees that gene expression alone cannot give a full picture of the cell state, because transcripts such as mRNAs need to be translated into proteins which sometimes need to be activated and each of these steps can be

regulated. Therefore more data types are needed to analyze regulation of the cell at a finer level of granularity. This is another reason why we chose to include sources of phenotype data in this study.
A lot of the classification work using machine learning has been done in cancer classification [1; 2; 9; 14; 15; 17; 19; 24] rather than predicting ontologies. This task is investigated in [5] but only for 6 classes (which were not defined by the Gene Ontology). Our approach is designed for making prediction for any of the classes in the Gene Ontology (on the order of a thousand different classes).

2. OVERVIEW OF THE DATA AND PREPROCESSING

The data used in this paper was gathered at Hughes Lab at the Banting and Best Department of Medical Research in the University of Toronto. In order to investigate the function of essential genes, which are required for survival and therefore cannot be knocked out, Hughes lab constructed a particular type of mutant yeast strains for two thirds of all the essential genes [16]. Construction was suspended because of project deadlines and financial reasons. These 602 mutants allow direct experimentation on the essential genes. There is a one-to-one correspondence between an essential gene and a mutant strain. The following datasets were collected and used for predicting gene function :

- *gene expression* from DNA microarrays measuring the abundance of gene transcripts of the mutant cells relative to the wild-type strain for the entire genome. The 291 samples, corresponding to 218 essential genes with replicates (out of the 602 constructed mutants), are publicly available on NCBI Gene Expression Omnibus (http://www.ncbi.nlm.nih.gov/geo/) through accession number GPL1229. After quantification, hybridized samples were normalized using background subtraction, followed by a LOWESS smoother to correct for dye discrepancies and by a high pass filter to remove any sorts of spatial artifacts (scratches, dust, gradient across the array or red corners ...) After investigating several techniques for imputing missing values, we used BPCAfill [18], which performed the best (using normalized root mean squared error as the measure of goodness of fit) on simulated datasets with the same proportion of missing data (approximately 13%). In the end the dimensionality of the data was reduced from 6307 genes on the arrays to 20 using principal components analysis (PCA) [11; 20] by selecting the eigenvectors associated with the 20 largest eigenvalues of the covariance matrix.

- *size distribution* measures the distribution of cell sizes for 591 of the 602 mutant strains. Normalization procedure: strains were grown by batches and this dataset was normalized so as to make the median of the median distribution of strains grown on the same batch to coincide for all batches. Validation of this normalization was done by verifying that the distribution of control wild-type strains grown in all batches coincided. The distributions were measured at 256 points, and the dimensionality was reduced to 8 by PCA. All growth phenotype datasets are available at http://hugheslab.med.utoronto.ca/Mnaimneh.

- *Drug Response* looks at the sensitivity of the mutant strains to different chemical compounds in 27 experimental conditions. 685 mutant strains, corresponding to 585 mutant strains with replicates, were grown on plates with one drug and the size of the colonies were compared to wild-type grown with the same drug. The value reported in the dataset was the log P-value that a difference existed between the two groups.

- *Morphology* represents the morphological features of the mutant cells which were visually inspected for 17 different characteristics such as elongated, budded or pointed cells. This data is the only type which is categorical. A 1 indicates that the feature was slightly observed for all mutant cells, a 0 indicates it was not. On rare occasion other types appear, 0.5 means the feature was slightly observed but the phenotype was not penetrant, 2 moderately observed for all cells, 2.5 moderately observed but the phenotype was not penetrant, 3 severely observed for all cells.

Each dataset covers a different set of the 602 constructed mutants , although these sets intersect, and the number of positive samples for a particular biological process depends on the dataset being used. A simple solution was to use these datasets independently.
Finally the gene labels we learned, which are organized in a hierarchical manner according to the Gene Ontology (GO) [23], were downloaded from the (SGD) Saccharomyces Genome Database [6; 7]. We used the *biological process* type of the GO database as our labels for gene function because the biologists we work with were interested in these rather than *molecular function* or *cellular component*. Almost 40% of the genes in the genome have no label for all of the the biological processes, we call these genes unlabeled. 15% of the 602 constructed mutants were uncategorized. Some GO biological processes are so broad and general that they involve thousands of genes, such as *protein metabolism [GO:0019538]* or *cell organization and biogenesis [GO:0016043]*. In fact, large top-level (high in the GO hierarchy) categories involving hundreds of genes are often not specific enough to verify experimentally. Therefore we have restricted this study by not showing biological processes that clearly involved too many genes to be interesting.

3. CLASSIFICATION BY HYPERPLANE

In this section we examine the case where the two classes are separable by a hyperplane. This is a strict assumption about the data, but it leads to predictions with high level of confidence for some biological processes nonetheless and represents a baseline for comparing the results obtained with the second view which we describe in the next section. We choose to fit the hyperplane using logistic regression [11] because of it's simplicity and also because it is well understood, and accepted in the biology community [3]. In 3.1 we investigate a method by which we can easily build decision rules customized to a particular biological process for classifying samples, precision being the only user-defined parameter. We apply these decision rules to the unlabeled samples in 3.2.
Each gene can be involved in several biological processes and therefore this is not the classical machine learning approach in which samples can belong to one class only, and of course

several genes can be involved in a biological process. We learned biological processes independently, which simplified the problem to discriminating between two classes for each biological process: either a gene is involved or it is not.

3.1 Cross Validation For Customized Decision Rules

We trained logistic regression classifiers by leave-one-out cross-validation on the labeled samples of each of the biological processes we chose to learn. Each time we computed the posteriors $P(Y = 1|X = x)$ where x was the sample set aside, Y denotes the class label (which takes the value 1 if a gene is involved in the biological process, and 0 otherwise). We had little choice but to use leave-one-out cross validation because, having so few positives in our samples (as little as 5 positives), we could not afford to waste labeled data by separating it into training and test sets.

In order to classify a sample we need to build a decision rule. One very simple rule could be to classify as positive any sample for which the posterior probability is above 0.5. Here we are faced with a decision making problem which needs a little more attention because of the cost associated with making false predictions. In molecular biology, running experiments is very expensive and we want to be very confident about the prediction being true before testing it in wet lab. All the cost of decisions is biased toward false predicted positives in this application and false negatives aren't given as much importance. As a result to increase our confidence on the predicted positives, we computed conservative thresholds for discriminating between classes, each depending on the particular biological process. A sample will be classified as positive if it's posterior is above that threshold $P(Y = 1|X = x) > t$. In the logistic regression setting, the classes are separated by the hyperplane defined by the equation $\theta^T x = 0$. When the input x is on the hyperplane,

$$P(Y = 1|X = x) = P(Y = 0|X = x) = 0.5 \qquad (1)$$

Raising the threshold, corresponds to translating that hyperplane in the direction of θ (or $-\theta$). Our procedure consists of translating the hyperplane toward the positive samples until the ratio of true positives to false positives is sufficiently high. Therefore we use cross validation, not only to control how well the classifiers are performing, but really to build decision rules for classifying the unlabeled samples.

The measure of satisfaction we used for translating the hyperplane is precision, which is the ratio of true positives to predicted positives, i.e.

$$\text{precision} = \frac{\text{true positives}}{\text{predicted positives}} = \frac{\text{TP}}{\text{TP} + \text{FP}} \qquad (2)$$

Predicted negatives cannot be confirmed experimentally (at least at Hughes Lab which is providing us with the data), so knowledge is gained only when predicted positives are confirmed and it is indeed precision biologists are interested in and not overall classification performance.

For a particular biological process, one approach could be to choose the threshold that leads to the maximum precision computed using all labeled samples, but we prefer to take a more conservative approach by setting a user-defined precision. That way predictions will only be made for biological processes for which the classifier reaches that precision at some threshold. For biological processes for which logistic regression performed poorly, no predictions will be made. Because precision is not a monotonic function in t, we chose the lowest threshold leading to the desired precision since this solution maximizes the recall (also known as sensitivity in the signal processing and biological worlds), which is the percentage of positives which are predicted positives:

$$\text{recall} = \frac{\text{true positives}}{\text{all positives}} = \frac{\text{TP}}{\text{TP} + \text{FN}} \qquad (3)$$

We computed five thresholds for each GO biological process, corresponding to precision levels of 100%, 85%, 75%, 60% and 50% based on the labeled data. The precision level used to classify a sample, along with the distance of that sample to the translated hyperplane leads to different of confidence levels.

3.2 Predicting functions for the unlabeled genes

For classifying the unlabeled samples, we trained a logistic regression classifier per biological process using all the available labeled samples and then computed the posterior probabilities $P(Y = 1|X = x)$ where x were the unlabeled samples. Unlabeled samples were classified as positive whenever their posterior was greater than the threshold, and predicted positives were reported.

Predictions were grouped by the precision level used and by biological process and are separated into batches depending on which dataset was used. Each prediction has four fields: a GO biological process, a systematic gene name, the precision level used for computing the threshold and finally the difference between the gene's posterior probability and the threshold which characterizes the distance from the translated hyperplane. All this data was assembled in tab delimited files available as supplementary data.

For increasing the significance of the precisions computed, we forced them to be based on a minimum of 10 predicted positives. We call *confident prediction* one that satisfies that constraint. We only reported confident predictions based on thresholds corresponding to 50% precision and above, this means that we can never make predictions for biological processes involving fewer than 5 genes.

It is worth underlying the fact that precision levels reported are minimums. A sample being predicted positive at a precision level could also have been predicted positive at a higher precision level. Summaries of these predictions are shown in Table 1-3. In these tables we report the number of unlabeled genes predicted grouped by biological process and by precision level. We indicate the number of known genes involved in each biological process as well as the number of positive samples available in the dataset used.

We observed that the procedure of fitting a hyperplane using logistic regression converged only for biological processes having more than 10 positive in our samples. In fact we observed that no confident predictions were made for biological processes involving fewer than 20 positives in our samples. The method we develop in the next section does not have this limitation.

4. HIERARCHICAL CLUSTERING ON LABELED DATA

In this section we investigate a method based on a different view of the data. We consider here that positive samples represent small islands among a sea of negatives, but we don't know how many islands there are nor their size. One

Number of predictions for GO-BP:	known in GOBP	# pos in samples	precision .6	precision .5
transcription [GO:0006350]	534	39		6
transcription, DNA-dependent [GO:0006351]	505	39		6
cell proliferation [GO:0008283]	571	37	5	8
RNA metabolism [GO:0016070]	336	34		10
cell cycle [GO:0007049]	494	33	4	6
RNA processing [GO:0006396]	297	33		4
biosynthesis [GO:0009058]	803	30		5
mitotic cell cycle [GO:0000278]	288	30		5
ribosome biogenesis and assembly [GO:0042254]	186	26		18
ribosome biogenesis [GO:0007046]	151	24		17
macromolecule biosynthesis [GO:0009059]	449	21	1	1
protein biosynthesis [GO:0006412]	442	21	1	1
DNA replication and chromosome cycle [GO:0000067]	219	20		1
transcription from Pol I promoter [GO:0006360]	149	20	7	8

Table 1: Summary of confident predictions made by logistic regression on the gene expression data

Number of predictions for GO-BP:	known in GOBP	# pos in samples	precision .6	precision .5
transcription [GO:0006350]	534	142		4
transcription, DNA-dependent [GO:0006351]	505	141		4
RNA metabolism [GO:0016070]	336	128	4	15
RNA processing [GO:0006396]	297	127		17
cell proliferation [GO:0008283]	571	116		5
ribosome biogenesis and assembly [GO:0042254]	186	85	3	15
ribosome biogenesis [GO:0007046]	151	79	3	15
transcription from Pol I promoter [GO:0006360]	149	76		14
rRNA processing [GO:0006364]	121	65		12
organelle organization and biogenesis [GO:0006996]	550	61		2

Table 2: Summary of confident predictions made by logistic regression on the cell size distributions

possibility would be to use k-nearest neighbors (kNN), but unfortunately we have no idea what to expect for k, and a simple majority vote would not work because of the high number of negatives almost everywhere (including regions of relatively high concentrations of positives). We develop an algorithm based on hierarchical clustering that circumvents these problems.

Clustering has been used extensively in functional genomics to analyze gene expression data [2; 4; 8; 13; 22] and is probably what biologists use and trust most. Biologists often use hierarchical clustering on gene expression data. For example, they usually display the resulting dendrogram immediately beside the gene expression data from which it was derived, and label the leaves of the dendrogram with gene names and/or biological processes. The method we develop here is based on this methodology, but extends it to an automated process. It also has the advantage of using all of the known functions of the genes in the hierarchical tree and not just their main function.

Our method looks for regions in the data space of high concentrations of positives. All that is required is some notion of "distance" between all pairs of elements. In contrast, logistic regression does not work for the morphology dataset because, although the data is technically real valued, it is still too categorical for the fit to converge.

4.1 Details of the Procedure

We first build a hierarchical tree on all available labeled and unlabeled samples using hierarchical agglomerative clustering [10] with average linkage. In constructing the tree, we ignore the labels on the data. In this way, we can include both labeled and unlabeled data in the tree, and more importantly, we can use the same tree for each biological process, thus saving on computing time, since the tree need only be built once. Thus, the construction of the tree can be viewed as a preprocessing step whose cost is amortized over all the biological processes. However, after the tree is constructed, it is not possible to add new unlabeled samples to the data.

We used the correlation coefficient between two samples as a measure of the distance between them rather than Euclidean distance. This is because the actual level of expression of two genes is less important than their profiles being correlated among a set of experiments. For example, the measured expression of a gene might be twice that of another gene in the same pathway because of experimental factors such as oligonucleotide probe quality (folding into a stable secondary structure, melting temperature etc).

Following the construction of the tree, we use it to build a classifier for each biological process. Recall that each such process provides a different set of labels for genes. Since the leaves of our tree represent genes, each leaf is assigned the label of the gene it represents. Leaves for unlabeled genes are labeled as negative, since it is likely that an unlabeled gene is not involved in any particular biological process. (We also flag such leaves, so as to remember that they are unlabeled). We can now look in the tree for regions of high concentrations of positive leaves, after which we assign labels to all the unlabeled genes that fall in such regions. These assignments represent our classifiers predictions.

To make these assignments, the algorithm computes a score σ for each internal node in the tree, reflecting the concentration of positives at the leaves under the node.

$$\sigma = \frac{\#\text{ of positives at leaves}}{\#\text{ of leaves}} \times \left(1 - \alpha e^{-\#\text{ of positives}}\right) \quad (4)$$

The first factor in this formula is the proportion of positive leaves under the node, it reflects the concentration of positives in the region of the data space in which the leaves are. The second factor tends to one when the number of positives raises, and tends to zero as the number of positives

Number of predictions for GO-BP:	known in GOBP	# pos in samples	precision .6	precision .5
RNA metabolism [GO:0016070]	336	140		1
RNA processing [GO:0006396]	297	139		1
cell proliferation [GO:0008283]	571	127	3	5
ribosome biogenesis and assembly [GO:0042254]	186	93		5
ribosome biogenesis [GO:0007046]	151	86		6
rRNA processing [GO:0006364]	121	71		3

Table 3: Summary of confident predictions made by logistic regression on the drugs dataset

```
Build hierarchical tree on all labeled and
unlabeled samples.
For each GO biological process GO-BPi do {
  Label the leaves according to GO-BPi.
  Label unlabeled samples as negatives.
  For each sample Sj do {
    Relabel Sj as negative.
    Compute the score of all internal nodes.
    Compute the score of Sj as maximum score of
    all it's ancestors.
  }
  Find lowest threshold that achieves
  user-specified precision.
  Classify unlabeled samples using this threshold.
  Report predicted positives.
}
```

Figure 1: Algorithm Pseudo-code

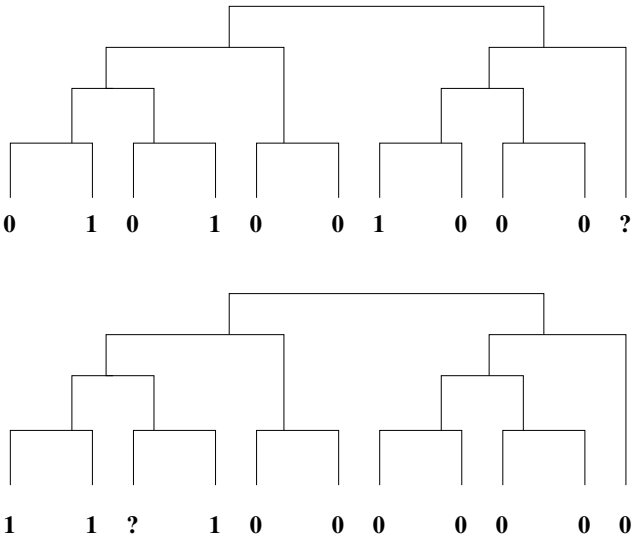


Figure 2: Toy hierarchical tree reused with labels from two biological processes

decreases. It gives more importance (higher score) to nodes with more positive leaves, *i.e.*, to larger regions of positive concentration, since we regard such regions to be more statistically significant. We have used $\alpha = 0.5$ and haven't investigated tweaking this parameter nor using other functions for the second factor of this equation. We then define the score of a leaf to be the maximum score of all it's ancestors (internal nodes). Since unlabeled samples are leaves in the tree, they automatically receive a score, which we use to classify them.

Before building decision rules, we use a technique similar to the cross validation of the previous section. At each iteration, we effectively remove a labeled sample by treating it as unlabeled. The scoring process described above is repeated each time. This provides a score for the labeled sample being treated as unlabeled. Each labeled leaf is scored in this way.

It is now easy to build a decision rule. We simply set a threshold, and a leaf is classified as positive if its score is above the threshold. To evaluate the rule, we apply it to labeled leaves, and compare each leaf's true label to its predicted label. A threshold that achieves a user-specified precision is then chosen. Finally, using this threshold, we use the decision rule to classify all the unlabeled data.

The pseudo-code for this algorithm is given in Figure 1. A toy example of how the tree is reused for each biological process is given in Figure 2. Our method is very fast, the whole process from building the tree to reporting predicted positives in all biological processes took a few seconds for each dataset on a Pentium IV 2GHz. This should be contrasted with the logistic regression methodology which required approximately a half hour for each dataset.

4.2 Results

Predicted positives were reported for all four datasets and assembled in tab delimited files. A prediction has four fields: a GO biological process, the gene systematic name, the difference between the score of the unlabeled leaf and the threshold used, and the precision corresponding to that threshold. The precision and the difference between the score and the threshold represent the confidence we have in the prediction. Summaries of these predictions (except for the morphology dataset) are shown in Table 4-6. We did not show the summaries for the morphology dataset, the number of confident predictions made were approximately the same as in Tables 5 and 6.

Comparing the two methods for identical datasets (Table 1 vs. 4, Table 2 vs. 5 and Table 3 vs. 6), we observe that our method produces many more confident predictions, at precision levels 50% and 60% (even 75% with the drugs dataset), and for more biological processes. In particular, our hierarchical method made prediction for 18 biological processes involving fewer than 20 positive in the samples whereas logistic regression produced none.

In Figure 3 we show the ROC curves of a couple of the classifiers used for making predictions, obtained by the method we developed. We clearly see that our method performs better than guessing the majority class, *i.e.* classify as negative every time, and achieves very high true positive rates at thresholds for which the false positive rates are still very low. For example, the classifier used for predicting genes involved in *glycerophospholipid biosynthesis* reaches a true

Number of predictions for GO-BP:	known in GOBP	# pos in samples	precision .6	precision .5
transcription [GO:0006350]	534	39		18
transcription, DNA-dependent [GO:0006351]	505	39		18
RNA metabolism [GO:0016070]	336	34	9	11
RNA processing [GO:0006396]	297	33	11	11
ribosome biogenesis and assembly [GO:0042254]	186	26	19	21
ribosome biogenesis [GO:0007046]	151	24	12	19
protein modification [GO:0006464]	361	23		3
organelle organization and biogenesis [GO:0006996]	550	22		1
macromolecule biosynthesis [GO:0009059]	449	21		9
protein biosynthesis [GO:0006412]	442	21		9
transcription from Pol I promoter [GO:0006360]	149	20	11	11
rRNA processing [GO:0006364]	121	18	8	8
catabolism [GO:0009056]	276	16		2
cytoskeleton organization and biogenesis [GO:0007010]	255	14		2
mRNA processing [GO:0006397]	124	14		4
macromolecule catabolism [GO:0009057]	176	12		1
lipid metabolism [GO:0006629]	190	11		1
lipid biosynthesis [GO:0008610]	111	11		1
RNA splicing [GO:0008380]	112	10		4
mRNA splicing [GO:0006371]	92	10		4
microtubule-based process [GO:0007017]	94	8		1
microtubule cytoskeleton organization and biogenesis [GO:0000226]	86	8		1
M-phase specific microtubule process [GO:0000072]	62	8		1
membrane lipid metabolism [GO:0006643]	85	6		1
membrane lipid biosynthesis [GO:0046467]	62	6		1
phospholipid metabolism [GO:0006644]	64	5		1
phospholipid biosynthesis [GO:0008654]	48	5		1
glycerophospholipid metabolism [GO:0006650]	34	5		1
glycerophospholipid biosynthesis [GO:0046474]	30	5		1

Table 4: Summary of confident predictions made by our clustering method on the gene expression data

positive rate of 100% for less than 2% false positive rate.

5. CONCLUSION & FUTURE WORK

We developed a method based on hierarchical clustering for labeled data to find regions in the data space of relatively high concentration of positives. This technique allows the analysis of biological processes involving very few genes. With this method, we were able to make confident predictions at precisions of 50% and above for biological processes for which our samples contained as few as 5 positives. The methodology developed here is not restricted to learning essential genes, but could be applied to any set of genes.

We used correlation as a measure of similarity between pairs of elements and average linkage to build the hierarchical tree. It would be interesting to investigate different distance metrics and especially other linkage strategies such as single linkage, which produces clusters that aren't necessarily compact.

We focused on making predictions for unlabeled genes. However, it would be biologically interesting to report cases in which a gene's true label is negative but whose predicted label is a confident positive. This is because negative labels in our dataset are sometimes wrong. A more challenging task would be to use datasets concurrently for the intersecting samples and independently for disjoint sets of samples. Also finding methods for learning biological processes concurrently rather than independently is one of our future goals. We are thinking of using the Gene Ontology hierarchy to propagate up the hierarchy predictions made lower down, because if a gene is involved in a biological process, it is also involved processes above it in the hierarchy. This isn't completely trivial because the hierarchy is not a tree and a process can have several parents. More interestingly, if a prediction is made in a biological process having children, we would like to find methods for making the prediction more specific by propagating it down the hierarchy as far as possible.

Acknowledgments

We wish to thank Hughes Lab for providing us with their data.

6. REFERENCES

[1] A. A. Alizadeh, M. B. Eisen, R. E. Davis, C. Ma, I. S. Lossos, A. Rosenwald, J. C. Boldrick, H. Sabet, T. Tran, X. Yu, J. I. Powell, L. Yang, G. E. Marti, T. Moore, J. J. Hudson, L. Lu, D. B. Lewis, R. Tibshirani, G. Sherlock, W. C. Chan, T. C. Greiner, D. D. Weisenburger, J. O. Armitage, R. Warnke, R. Levy, W. Wilson, M. R. Grever, J. C. Byrd, D. Botstein, P. O. Brown, and L. M. Staudt. Distinct types of diffuse large B-cell lymphoma identified by gene expression profiling. *Nature*, 403(6769):503–511, February 2000.

[2] U. Alon, N. Barkai, D. A. Notterman, K. Gish, S. Ybarra, D. Mack, and A. J. Levine. Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *Proceedings of the National Academy of Science*, 96(12):6745–6750, June 1999.

[3] J. S. Bader, A. Chaudhuri, J. M. Rothberg, and J. Chant. Gaining confidence in high-throughput protein interaction networks. *Nature Biotechnology*, 22(1), January 2004.

[4] A. Ben-Dor, R. Shamir, and Z. Yakhini. Clustering gene expression patterns. *Journal Of Computational Biology*, 6:281–297, 1999.

[5] M. P. S. Brown, W. N. Grundy, D. Lin, N. Cristianini, C. W. Sugnet, T. S. Furey, M. Ares, and D. Haussler. Knowledge-based analysis of microarray gene expression data by using support vector machines. *Proceeding*

Number of predictions for GO-BP:	known in GOBP	# pos in samples	precision .5
transcription [GO:0006350]	534	142	5
transcription, DNA-dependent [GO:0006351]	505	141	5
cell proliferation [GO:0008283]	571	116	1
cell cycle [GO:0007049]	494	99	1
ribosome biogenesis and assembly [GO:0042254]	186	85	5
ribosome biogenesis [GO:0007046]	151	79	2
transcription from Pol I promoter [GO:0006360]	149	76	3

Table 5: Summary of confident predictions made by our clustering method on the cell size distributions

Number of predictions for GO-BP:	known in GOBP	# pos in samples	precision .75	precision .6	precision .5
transcription [GO:0006350]	534	159	2	5	5
transcription, DNA-dependent [GO:0006351]	505	158	2	5	5
RNA metabolism [GO:0016070]	336	140			27
RNA processing [GO:0006396]	297	139			17
DNA metabolism [GO:0006259]	379	68			6
mRNA processing [GO:0006397]	124	60			4
nuclear organization and biogenesis [GO:0006997]	213	42		3	3
chromosome organization and biogenesis (sensu Eukarya) [GO:0007001]	178	35		3	3
establishment and/or maintenance of chromatin architecture [GO:0006325]	155	32		3	3
DNA packaging [GO:0006323]	155	32		3	3

Table 6: Summary of confident predictions made by our clustering method on the drugs dataset

of the National Academy of Science, 97(1):262–7, January 2000.

[6] K. R. Christie, S. Weng, R. Balakrishnan, M. C. Costanzo, K. Dolinski, s S Dwight, S. R. Engel, B. Feierbach, D. G. Fisk, J. E. Hirschman, E. L. Hong, L. Issel-Tarver, R. Nash, A. Sethuraman, B. Starr, C. L. Theesfel, R. Andrada, G. Binkley, Q. Dong, C. Lane, M. Schroeder, D. Botstein, and J. M. Cherry. Saccharomyces genome database (sgd) provides tools to identify and analyze sequences from saccharomyces cerevisiae and related sequences from other organisms. *Nucleic Acids Research*, 32:D311–D314, 2004.

[7] S. S. Dwight, M. A. Harris, K. Dolinski, C. A. Ball, G. Binkley, K. R. Christie, D. G. Fisk, L. Issel-Tarver, M. Schroeder, G. Sherlock, A. Sethuraman, S. Weng, D. Botstein, and J. M. Cherry. Saccharomyces genome database (sgd) provides secondary gene annotation using the gene ontology (go). *Nucleic Acids Research*, 30(1):69–72, 2002.

[8] M. Eisen, P. Spellman, P. Brown, and D. Botstein. Cluster analysis and display of genome-wide expression patterns. *Proceedings of the National Academy of Science*, 95(25):14863–14868, December 1998.

[9] T. S. Furey, N. Cristianini, N. Duffy, D. W. Bednarski, M. Schummer, and D. Haussler. Support vector machine classification and validation of cancer tissue samples using microarray expression data. *Bioinformatics*, 16(10):906–14, October 2000.

[10] D. Hand, H. Mannila, and P. Smyth. *Principles of Datamining*. MIT Press, 2001.

[11] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer, 2001.

[12] T. R. Hughes, M. Mao, A. R. Jones, J. Burchard, M. J. Marton, K. W. Shannon, S. M. Lefkowitz, M. Ziman, J. M. Schelter, M. R. Meyer, S. Kobayashi, C. Davis, H. Dai, Y. D. He, S. B. Stephaniants, G. Cavet, W. L. Walker, A. Westand, E. Coffey, D. D. Shoemaker, R. Stoughton, A. P. Blanchard, S. H. Friend, and P. S. Linsley. Expression profiling using microarrays fabricated by an ink-jet oligonucleotide synthesizer. *Nature Biotechnology*, 19(4):342–347, April 2001.

[13] T. R. Hughes, M. J. Marton, A. R. Jones, C. J. Roberts, R. Stoughton, C. D. Armour, H. A. Bennett, E. Coffey, H. Dai, Y. D. He, M. J. Kidd, A. M. King, M. R. Meyer, D. Slade, P. Y. Lum, S. B. Stepaniants, D. D. Shoemaker, D. Gachotte, K. Chakraburtty, J. Simon, M. Bard, and S. H. Friend. Functional discovery via a compendium of expression profiles. *Cell*, 102:109–126, July 2000.

[14] J. Khan, J. S. Wei, M. Ringner, et. al, and P. S. Meltzer. Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks. *Nature Medicine*, 7(6):673–679, 2001.

[15] Y. Lee and C. K. Lee. Classification of multiple cancer types by multicategory support vector machines using gene expression data. *Bioinformatics*, 19(9):1132–9, June 2003.

[16] S. Mnaimneh, A. P. Davierwala, J. Haynes, J. Moffat, W.-T. Peng, W. Zhang, X. Yang1, J. Pootoolal, G. Chua, A. Lopez, M. Trochesset, D. Morse, N. J. Krogan, S. L. Hiley, Z. Li, Q. Morris, J. Grigul, N. Mitsakakis, C. J. Roberts, J. F. Greenblatt, C. Boone, C. A. Kaiser, B. J. Andrews, and T. R. Hughes. Exploration of essential gene functions via titratable promoter alleles. *Cell*, July 2004.

[17] D. V. Nguyen and D. M. Rocke. Tumor classification by partial least squares using microarray gene expression data. *Bioinformatics*, 18(1):39–50, Jan 2002.

[18] S. Oba, M. Sato, I. Takemasa, M. Monden, K. Matsubra, and S. Ishii. A Bayesian missing value estimation method for gene expression profile data. *Bioinformatics*, 19(16):2088–2096, 2003.

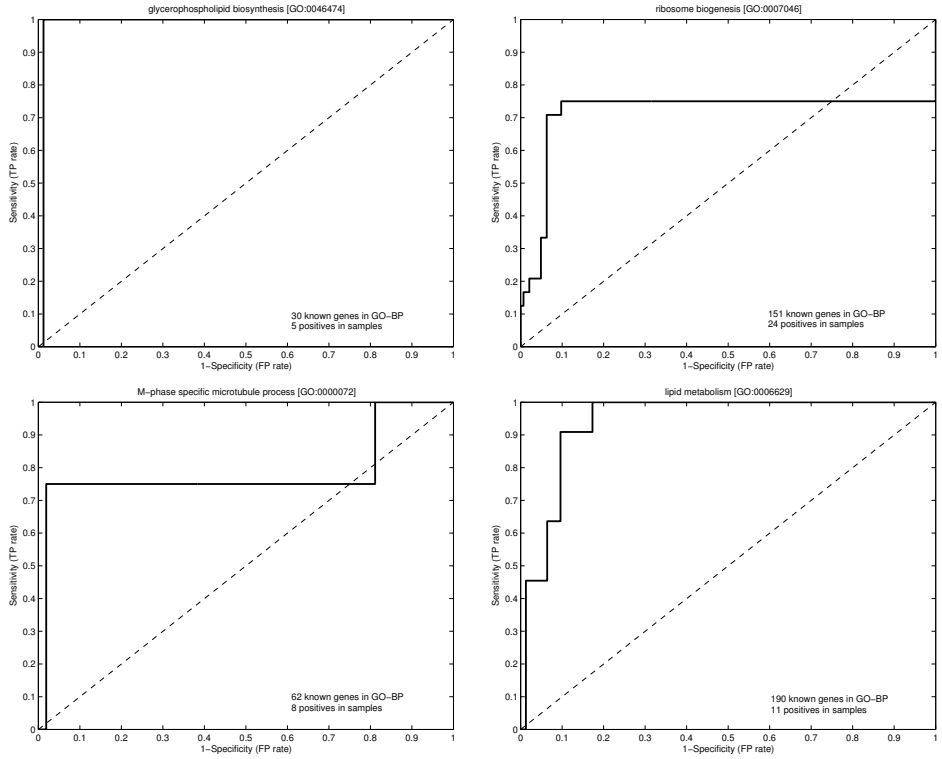


Figure 3: ROC curves for some of the classifiers we used for making predictions

[19] S. Ramaswamy, P. Tamayo, R. Rifkin, S. Mukherjee, C. H. Yeang, M. Angelo, C. Ladd, M. Reich, E. Latulippe, J. P. Mesirov, T. Poggio, W. Gerald, M. Loda, E. S. Lander, and T. R. Golub. Multiclass cancer diagnosis using tumor gene expression signatures. *Proceedings of National Academy of Science*, 98(26):15149–54, December 2001.

[20] S. Raychaudhuri, J. M. Stuart, and R. B. Altman. Principal components analysis to summarize microarray experiments: Application to sporulation time series. *Proceedings of the Pacific Symposium on Biocomputing*, 5:465–466, 2000.

[21] M. Schena, D. Shalon, R. W. Davis, and P. O. Brown. Quantitative monitoring of gene expression patterns with a complementary DNA microarray. *Science*, 270(5235):467–470, October 1995.

[22] R. Sharan and R. Shamir. Click: A clustering algorithm for gene expression analysis. In *ISMB*, 2000.

[23] Gene Ontology Consortium. The gene ontology (GO) database and informatics resource. *Nucleic Acids Research*, 32:D258–D261, 2004.

[24] M. West, C. Blanchette, H. Dressman, E. Huang, S. Ishida, R. Spang, H. Zuzan, J. A. Olson, J. R. Marks, and J. R. Nevins. Predicting the clinical status of human breast cancer by using gene expression profiles. *Proceedings of the National Academy of Science*, 98(20):11462–11467, September 2001.

A Maximum Entropy Approach to Biomedical Named Entity Recognition

Yi-Feng Lin, Tzong-Han Tsai, Wen-Chi Chou, Kuen-Pin Wu, Ting-Yi Sung and Wen-Lian Hsu
Institute of Information Science, Academia Sinica
128, Secton 2, Academy Road, Taipei, Taiwan
{lego, thtsai, jacky957, kpw, tsung, hsu}@iis.sinica.edu.tw

ABSTRACT

Machine learning approaches are frequently used to solve name entity (NE) recognition (NER). In this paper we propose a hybrid method that uses maximum entropy (ME) as the underlying machine learning method incorporated with dictionary-based and rule-based methods for post-processing. Simply using ME for NER, inaccurate boundary detection of NEs and misclassification may occur. Some NEs are partially recognized by ME. In the post-processing stage, we use dictionary-based and rule-based methods to extend boundary of partially recognized NEs and to adjust classification. We use GENIA corpus 3.01 to conduct 10-fold cross-verification experiments. To evaluate the performance, we consider the longest NE annotations. We evaluate our approach using standard precision (P), recall (R), and F-score, where F-score is defined as 2PR/(P+R). The precision, recall and F-score ([P, R, F]) of our ME module for overall 23 categories is [0.512, 0.538, 0.525], and after the post-processing the performance becomes [0.729, 0.711, 0.72] for [P, R, F]. For protein, DNA and RNA classes, our method achieves [P, R, F] of [0.77, 0.80, 0.785], [0.653, 0.748, 0.7], and [0.716, 0.788, 0.752], respectively. The post-processing stage significantly improves the performance of our ME-based NER module.

1. INTRODUCTION

The amount of biomedical literature available on the Web is rapidly increasing. There is a pressing need for biomedical information extraction. To extract useful information from natural language text, we must first recognize biomedical named entities in the text. In fact, named entity (NE) recognition (NER) is a fundamental research topic in natural language processing (NLP), which involves *entity identification* and *classification*.

Unlike NER in the newswire domain, NER in the biomedical domain remains a perplexing challenge. Biomedical NEs in general do not follow any nomenclature, and can be comprised of long compound words or short abbreviations. Some even contain various symbols or spelling variations. In summary, difficulties of NER in the biomedical domain are as follows:

(1) Unknown word identification:
Unknown words can be acronyms, abbreviations, or words containing hyphens, digits, letters, and Greek letters. Examples of NEs with unknown words include: *alpha B1*, *GM-CSF*, *Adenyly cyclase 76E*, and *4'-mycarosyl isovaleryl-CoA transferase*.

(2) Named entity boundary identification:
The boundary of an NE can be a regular English word, unknown word, Roman numeral, or digit. For example, *MHC Class II*, *latent membrane protein 1*, *NF-kappaB consensus site*, *cyclin-like UDG gene product* all have different types of boundaries. Additionally, nested NEs (an NE embedded in another NE, referred to as *cascaded* NEs by Shen et al. [9]) further complicate this problem. Consider the named entity *kappa 3 binding factor*. Its annotation <PROTEIN> <DNA> kappa 3 </DNA> binding factor </PROTEIN> has two right boundaries at *3* and *factor*, which correspond to the embedded NE in the DNA category and the nested NE of the Protein category, respectively.

(3) Named entity classification:
Once an NE is identified, it is then classified into a category such as protein, DNA, RNA, and so on. Ambiguity and inconsistency are often encountered at this stage. NEs with the same orthographical features may fall into different categories. For example, BRIX and SCOP both have the AllCaps feature, but the former is a gene and the latter is a protein. An NE may belong to multiple categories, e.g., *ELK1* is both a DNA and a protein. *p53* is an another example. *p53* is a synonym for the gene *TP53* in HUGO nomenclature; but in the GENIA corpus, *p53* is also tagged as a protein. Such ambiguity is intrinsic. Another complication is that a nested NE of one category may contain an NE of another category. For instance, a protein name may contain the gene coding for this protein. For example, *A27L protein* is a protein name containing *A27L* which is the gene coding for this protein. We need to properly distinguish *A27L* from *A27L protein*.

To tackle these challenges, researchers use NLP techniques such as machine learning, dictionary-based methods and rule-based methods. Tsuruoka et al. [11] and Hanisch et al. [3] present dictionary-based approaches. Since new biomedical NEs keep being generated in literature, the machine learning approach prevails. After the release of GENIA corpus [6], machine learning approaches using GENIA corpus as training corpus are reported [10; 5; 13; 9; 14]. GENIA corpus provides a benchmark for evaluating different methods. The overall F-scores on 23 categories in GENIA corpus reported by these systems were at most 0.67.

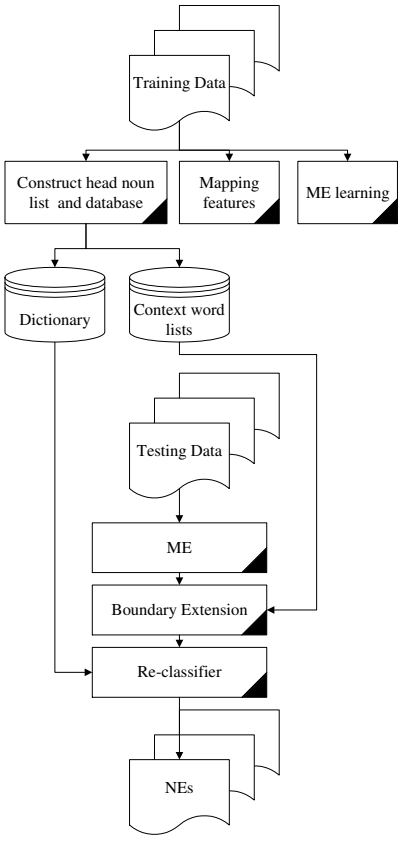


Figure 1: Method overview

The performance of machine learning approaches has big room for improvement. This fact can be attributed to small size of training corpora. Though GENIA corpus is the largest corpus for NER, it is rather small in comparison with the size of biomedical NEs. Various strategies are proposed to enhance the performance. In this paper, we use maximum entropy (ME) as our underlying machine learning method. Unexceptionally, the F-score of pure ME is 0.525 over the 23 categories of GENIA corpus. Our post-processing of ME output aims to resolve boundary detection problems and correct misclassification problems. Dictionary-based and rule-based methods are used, which significantly improves the performance.

2. ME-BASED BIOMEDICAL NER FRAMEWORK

Our recognition method consists of two stages: (1) ME-based recognition, (2) post-processing including boundary extension and reclassification. We first use ME for NER. Then we use a dictionary and rules to correct boundary identification errors by boundary extension. After boundary error correction is performed, the results are reclassified. Our method is depicted in Figure 1.

2.1 Maximum Entropy

We regard each word as a token. Since a named entity can have more than one token, each token is associated with a tag that indicates the category of the NE and the location of the token within the NE, for example, *x_begin*, *x_continue*, *x_end*, *x_unique* where x is a category. The first three tags denote respectively the beginning, the middle and the end of an NE in category x. The fourth tag denotes that a token itself is an NE of category x. In addition, we use the tag *unknown* to indicate that a token is not part of an NE. The NER problem can then be rephrased as the problem of assigning one of $4n + 1$ tags to each token, where n is the number of NE categories. In our ME module, there are 23 named entity categories and 93 tags. For example, one way to tag the phrase *IL-2 gene expression*, *CD28*, and *NF-kappa B* in a paper is "*othername_begin*, *othername_continue*, *othername_end*, *unknown*, *protein_unique*, *unknown*, *unknown*, *protein_begin*, *protein_end*."

ME is a flexible statistical model which assigns an outcome for each token based on its *history* and *features*. Outcome space is comprised of the 93 tags for an ME formulation of NER. ME computes the probability $p(o|h)$ for any o from the space of all possible outcomes O, and for every h from the space of all possible histories H. A *history* is all the conditioning data that enables one to assign probabilities to the space of outcomes. In NER, *history* can be viewed as all information derivable from the training corpus relative to the current token.

The computation of $p(o|h)$ in ME depends on a set of binary-valued *features*, which are helpful in making predictions about the outcome. For instance, one of our features is: when all characters of the current token are capitalized, it is likely to be part of a biomedical NE. Formally, we can represent this feature as follows:

$$f(h,o) = \begin{cases} 1 & \text{if } \textit{Current-Token-AllCaps(h)} = \text{true} \\ & \text{and } o = \textit{protein\_begin}; \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

Here, *Current-Token-AllCaps(h)* is a binary function that returns the value *true* if all characters of the current token in the history h are capitalized. Given a set of features and a training corpus, the ME estimation process produces a model in which every feature f_i has a weight α_i. From [1], we can compute the conditional probability as:

$$p(o|h) = \frac{1}{Z(h)} \prod_i \alpha_i^{f_i(h,o)} \quad (2)$$

The probability is given by multiplying the weights of active features (i.e., those $f_i(h,o) = 1$). The weight α_i is estimated by a procedure called Generalized Iterative Scaling. This method improves estimation of weights at each iteration. The ME estimation technique guarantees that, for every feature f_i, the expected value of α_i equals the empirical expectation of α_i in the training corpus.

As noted in Borthwick [2], ME allows users to focus on finding features that characterizes the problem while leaving feature weight assignment to the ME estimation routine.

2.2 Decoding

After having trained an ME model and assigned the proper weights α_i to each feature f_i, decoding (i.e., *marking up*) a new piece of text becomes simple. First, the ME module tokenizes the text. Then, for each token, we check which features are active and combine α_i of the active features according to Equation 2. Finally, a Viterbi search is run

Table 1: Orthographical features

Orthographical features	Example	Orthographical features	Example
AllCaps	EBNA, NFAT, LMP	AlphaDigit	p50, p65
AlphaDigitAlpha	IL23R, E1A	ATGCSequence	CCGCCC, ATGAT
CapLowAlpha	Src, Ras, Epo	CapMixAlpha	NFkappaB, EpoR
CapsAndDigits	IL2, STAT4, SH2	DigitAlpha	2xNFkappaB, 2A
DigitAlphaDigit	32Dc13, 2D3	DigitCommaDigit	1,25
Digits	1, 2, 3, 1.1	Greek Letter	alpha, beta
Hyphen	-	LowMixAlpha	mRNA, mAb
Roman Numeral	I, II, III	SingleCap	A-Z
Stop word	at, in	Other	",", ".", "(", ")"

Table 2: Head nouns

	Head nouns
Unigram	factor, protein, receptor, alpha, NF-kappaB, IL-2, cytokine, AP-1, kinase, IL-4, transcription, domain, complex, TNF-alpha, IFN-gamma, Nuclear, p50, p65, beta, NFAT, CD28, TNF, PKC, calcineurin, molecules, GM-CSF, GATA-1, IL-12, subunit, cell, STAT3, family, antibody, TCR, CIITA, chain, tumor, gamma, factor-alpha, expression, interleukin, IkappaBalpha
Bigram	NF-kappa B, transcription factor, I kappa, kappa B, nuclear factor, protein kinase, B alpha, kinase C, tumor necrosis, T cell, glucocorticoid receptor, colony-stimulating factor, binding protein, factor alpha, necrosis factor-alpha, adhesion molecule, monoclonal antibody, necrosis factor, T lymphocyte, cytoplasmic domain, gene product, binding domain

Table 3: Morphological features

∼ase	∼blast	∼cin	∼cyte
∼kin	∼lin	∼lipid	∼ma
∼mide	∼peptide	∼phil	∼rin
∼rogen	∼sor	∼tin	∼tor
∼virus	∼vitamin	∼zole	anti∼
cyto∼	dehydr∼	erytho∼	hemo∼

to find the highest probability path through the lattice of conditional probabilities that does not produce any invalid tag sequences. For instance, the sequence [*protein_begin*, *othername_continue*] is invalid because it does not contain an ending token and these two tokens are not in the same category. Further details on the Viterbi search can be found in [12].

2.3 Related Studies of NER Using ME

Raychaudhuri et al. [8] uses ME to assign Gene Ontology tags to genes appearing in biomedical literature. They report that ME outperforms the Naive Bayes method and the nearest-neighbor method. ME is also used for acronym and abbreviation normalization in medical texts. Pakhomov [7] and Kazama et al. [4] report that SVM outperforms ME for biological NER. In Kazama et al. [4], the comparison is made using GENIA corpus version 1.0. The precision, recall and F-score ([P, R, F]) of the SVM-based system was [0.562, 0.528, 0.544] for overall categories and [0.492, 0.664, 0.565] for protein. The ME-based system reports [P, R, F] of [0.534, 0.530, 0.532] for overall performance and [0.491, 0.621, 0.548] for protein. Nevertheless, the authors also state that one advantage of the ME model is that it allows flexible feature selection. When new features, e.g., syntax features are added to ME, users do not need to reformulate the model like in the HMM model and ME estimation routine can automatically calculate new weight assignment. Thus we choose ME as the underlying machine learning model.

3. FEATURES

Feature selection is critical to the success of machine learning approaches. Orthographical features, head noun features, morphological features, and part-of-speech (POS) features are frequently used for token identification. We use POS features annotated in the GENIA corpus and report the remaining features below.

3.1 Orthographical Features

Table 1 lists some orthographical features used in our system. In our experience, AllCaps, CapMixAlpha, LowMixAlpha, SingleCap are more useful than others.

3.2 Head Nnouns

The head noun is usually the major noun or noun phrase of an NE that describes its function or the property, e.g., *transcription factor* is the head noun for the NE *NF-kappa B transcription factor*. Compared with the other words in NE, head noun is a decisive factor for distinguishing the NE class. For instance, the classifications of <Protein> NF-kappa B transcription factor </Protein> and <DNA> IFN-gamma activation sequence </DNA> are determined by the head nouns *transcription factor* and *sequence*. In this work, only unigram and bigram head nouns are considered. We use training corpus to obtain 960 frequently used head nouns, and some are listed in Table 2.

3.3 Morphological Features

We consider morphological features of at least three characters in length. Some are listed in Table 3.

4. POST-PROCESSING AND RECLASSIFICATION FOR ERROR CORRECTION

Using ME, we find some NEs are partially recognized or mistakenly classified. In the post-processing stage, we aim to resolve boundary detection problems of partially recognized NEs by a boundary extension method. Afterwards, we use a re-classifier to resolve NE misclassification. Dictionary-based and rule-based methods are used for post-processing. The dictionary is constructed from the training corpus.

4.1 Boundary Extension

For those partially recognized NEs, we deal with two types of boundary detection problems that arise from (1) nested NEs and (2) brackets for name alias and slash for concatenated names.

Nested NEs may cause boundary detection problems. Consider the example "$[\text{E1A}]_{/\text{protein}}$ gene" → "$[\text{E1A gene}]_{/\text{DNA}}$." A straightforward right(R)-boundary extension rule is to extend the boundary if the NE is followed by NEs and/or head nouns. In the example "$[\text{GATA-1}]_{/\text{protein}}$ activity" → "$[\text{GATA-1 activity}]_{/\text{othername}}$," the word *activity* is not a head noun. How do we determine whether the right boundary should be extended to *activity*? Consider another example: "type $[\text{I receptor}]_{/\text{protein}}$" → "$[\text{type I receptor}]_{/\text{protein}}$." Should the left boundary extend to the word *type*? For the left(L)-boundary extension, we consider extension to include a modifier. What modifiers are allowed?

To resolve the abovementioned problems, we compile two lists of the *leftmost (L)* and the *rightmost (R) context words* of NEs in the training corpus. To construct these lists, we calculate the frequency of each context word candidate and determine a cutoff threshold to include candidates into the lists. The threshold is expected to affect the content of the lists and thus, the performance of post-processing. However, in our experiments, we have tried different threshold values and found that the threshold does not significantly affect the performance. We thus include all the candidates in the lists. Note that these context words may not be head nouns, but unigram head nouns surely belong to the lists.

In the previous example, *activity* is in the R-context word list and thus the right boundary can be extended to *activity*. We use context word lists to examine un-tagged tokens that are adjacent to ME-recognized NEs. If these tokens appear in the L- & R-context word lists, then they are concatenated with ME's output. But simply using context word lists to determine boundary extension may fail in some cases. For example, *binding* is in the R-context word list. But *binding* can be tagged as a verb, an adjective or a noun. If *binding* is tagged as a verb, it is unlikely to be a part of an NE. Only few tokens tagged as a verb are included in NEs of GENIA corpus. We thus consider only adjective and noun as valid POS tags for the token in consideration. To further improve boundary extension accuracy, we examine the validity of the POS tag of the token. If this token appears in a context word list and its POS is valid, we will concatenate this token with the NE.

In summary, our boundary extension algorithm to resolve nested NEs goes as follows:

Step 1. Check R-boundary extension: Extend the boundary of an NE recognized by ME repeatedly if the NE is followed by another NE or a token in R-context word list with valid POS tag.

Step 2. Check L-boundary extension: Repeat similar procedure as in Step 1.

Step 3. Repeat Step 1 and 2 until no extension occurs.

Our algorithm can handle six patterns of nested NE construction presented in Zhou et al. [14].

The second type of boundary detection problem occurs when NEs contain brackets for name alias and slash for concatenated names which are not well handled by maximum entropy. For example, *basic helix-loop-helix (bHLH) motif* is an NE. Our ME module recognizes both *basic helix-loop-helix* and *bHLH* as protein. Since "(" and ")" are not valid context words, the previous algorithm cannot extend the boundary of ME's output. Our solution is to detect whether *motif* is a valid context word. If yes, *basic helix-loop-helix (bHLH) motif* will be concatenated as one named entity.

After performing boundary extension for nested NEs, we use rule-based approach to extend boundary of the second type problem. The rules are given as follows:

1. NE := NE (+ NE) + R-context word;
2. NE := NE + / + NE (+ / + NE) + R-context word.

Inspecting the results generated by ME, we found that some human names were identified as NEs. A special module developed by our laboratory was introduced to filter these errors. This module is originally designed to extract authors, paper titles and journal names from citations.

4.2 Re-classifier

In boundary extension stage, we do not change the classification. Our re-classifier aims to resolve two types of classification errors. The first type is associated with boundary extension, for example, "$[\text{GATA-1}]_{/\text{protein}}$ activity" → "$[\text{GATA-1 activity}]_{/\text{othername}}$." The other type is intrinsic ambiguity caused by abbreviations. Orthographical features of AllCaps and CapsAndDigits are sometimes insufficient to distinguish between abbreviations of protein and DNA. For example, CD28 is a protein, and PS1 a DNA.

The re-classifier performs two steps. The first step is dictionary lookup. If the named entity is in the dictionary, we assign new class according to the dictionary. If the NE is not in the dictionary, we take the second step to adjust the classification according to R context word. We assign the class according to the context word.

5. EXPERIMENTS

5.1 GENIA Corpus

We use GENIA corpus version 3.01 to evaluate our system. The GENIA corpus contains 2,000 abstracts extracted from the Medline database and these abstracts are annotated with Penn Treebank part-of-speech tags. The annotation of the NEs is based on GENIA ontology. In our experiments, we use 23 distinct NE categories of GENIA corpus.

5.2 Experimental Results

We conduct 10-fold cross validation experiments and divide 2000 abstracts into 10 collections. Each collection contains not only abstracts but also paper titles. We evaluate our approach using standard precision (P), recall (R), and F-score, where F-score is defined as $2\text{PR}/(\text{P}+\text{R})$. To evaluate our method, we consider the longest word annotation, since these NEs are useful for relation extraction.

Table 5: NE recognition performance

Config	Boundary Extension			Reclassify		NE Recognition
	BE-1	BE-2	BE-3	RC-1	RC-2	P/R/F
Baseline						0.512/0.538/0.525
Conf4	√	√	√			0.645/0.634/0.639
Conf5	√	√	√	√		0.67/0.658/0.664
Conf6	√	√	√		√	0.707/0.695/0.701
Conf7	√	√	√	√	√	**0.727/0.715/0.721**

Table 7: Partial matching performance

Task	NE Identification			NE Recognition		
Measurement	Precision	Recall	F-Score	Precision	Recall	F-Score
Exact Match	0.776	0.763	0.769	0.727	0.715	0.721
LD=1, ER > CR	0.802	0.788	0.795	0.74	0.728	0.734
LD=2, ER > CR	0.818	0.804	0.811	0.754	0.741	0.747
LD=1, CR > ER	0.804	0.791	0.797	0.744	0.731	0.737
LD=2, CR > ER	0.809	0.795	0.802	0.748	0.735	0.741
RD=1, ER > CR	0.805	0.79	0.797	0.733	0.72	0.726
RD=2, ER > CR	0.813	0.798	0.805	0.737	0.724	0.73
RD=1, CR > ER	0.808	0.791	0.799	0.735	0.721	0.728
RD=2, CR > ER	0.811	0.802	0.806	0.736	0.723	0.729

Table 4: NE identification performance

Config	Boundary Extension			NE Identification
	BE-1	BE-2	BE-3	P/R/F
Baseline				0.56/0.589/0.574
Conf1			√	0.582/0.597/0.594
Conf2		√		0.591/0.6/0.595
Conf3	√			0.757/0.746/0.751
Conf4	√	√	√	**0.776/0.763/0.769**

Table 6: System performance comparison (measured in F-Score)

Category	Overall	Protein	DNA	RNA
Our system	0.721	0.785	0.700	0.752
Zhou et al, 2004	0.666	0.758	0.633	0.612

In Table 4, we report the named entity identification (regardless of classification) performance. We use BE-1 to denote the nested boundary extension algorithm, BE-2 to denote the boundary extension for brackets and slashes, and BE-3 to denote the module to remove human names. From the figures, we can see that each method yields different degree of improvement in NE identification (boundary detection) performance. BE-1, which improves the NE identification performance by 0.177, is the most effective boundary extension method among the three methods.

In Table 5, we report the named entity recognition (including classification) performance. We use RC-1 to denote the re-classifier using dictionary lookup and RC-2 to denote the re-classifier using R context word. In Table 6, we show the performance of our system in overall 23 categories and in protein, DNA and RNA classes, and compare them with those reported in Zhou et al. [14]. We can see that our system has advantage over Zhou's system in each main NE category and in overall performance. In Table 7, we report the partial matching results. We use LD = i (RD = i) to mean that the recognized NE differs from the annotation by only i words at the left (right) boundary. ER and CR denote the length of the recognized NE (the experiment result) and the length of the annotation (the correct result).

6. CONCLUDING REMARKS

In this paper, we propose a hybrid method using maximum entropy and dictionary/rule-based methods. Currently, dictionary is only used in the post-processing stage. In the future, we shall improve our system by also using dictionary in the preprocessing stage. However, we need to overcome the difficulty arising from integration of dictionary preprocessing with ME. In the post-processing stage, we shall explore more extensively on determining rules for boundary extension and entity concatenation. In addition, we shall try to automatically generate good rules to enhance our system.

7. REFERENCES

[1] A. Berger, S. A. D. Pietra, and V. J. D. Pietra. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22:39–71, 1996.

[2] A. Borthwick. *A maximum entropy approach to named entity recognition*. New York University, 1999.

[3] D. Hanisch, J. Fluck, H. Mevissen, and R. Zimmer. Playing biology's name game: identifying protein names in scientific text. In *PSB '03*, 2003.

[4] J. Kazama, T. Makino, Y. Ohta, and J. Tsujii. Tuning support vector machines for biomedical named entity recognition. In *ACL 2002*, 2002.

[5] K.-J. Lee, Y.-S. Hwang, and H.-C. Rim. Two-phase biomedical ner recognition based on svms. In *ACL 2003*, 2003.

[6] T. Ohta, Y. Tateisi, H. Mima, and J. Tsujii. Genia corpus: an annotated research abstract corpus in molecular biology domain. In *Proceedings of the Human Language Technology Conference (HLT)*, 2002.

[7] S. Pakhomov. Semi-supervised maximum entropy based approach to acronym and abbreviation normalization in medical text. In *ACL 2002*, 2002.

[8] S. Raychaudhuri, J. Chang, P. Sutphin, and R. Altman. Associating genes with gene ontology codes using a maximum entropy analysis of biomedical literature. *Genome Research*, 12, 2002.

[9] D. Shen, J. Zhang, G. Zhou, J. Su, and C. Tan. Effective adaptation of a hidden markov model-based named entity recognizer for biomedical domain. In *ACL 2003*, 2003.

[10] K. Takeuchi and N. Collier. Bio-medical entity extraction using support vector machines. In *ACL 2003*, 2003.

[11] Y. Tsuruoka and J. Tsujii. Boosting precision and recall of dictionary-based protein name recognition. In *ACL 2003*, 2003.

[12] A. J. Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, IT:260–269, 1967.

[13] K. Yamamoto, T. Kudo, A. Konagaya, and Y. Matsumoto. Protein name tagging for biomedical annotation in text. In *ACL 2003*, 2003.

[14] G. Zhou, J. Zhang, J. Su, D. Shen, and C. Tan. Recognizing names in biomedical texts: a machine learning approach. *Bioinformatics*, 20:1178–1190, 2004.

Discovering Spatial Relationships Between Approximately Equivalent Patterns in Contact Maps *

Hui Yang, Keith Marsolo, Srinivasan Parthasarathy and Sameep Mehta
Department of Computer Science and Engineering
The Ohio State University
Columbus, Ohio, USA

{yanghu, marsolo, srini, mehtas}@cse.ohio-state.edu

ABSTRACT

We present a method for finding relationships between approximate patterns in contact maps. We examine contact maps generated from protein data in order to discover spatial relationships among the connected patterns contained in those maps. We discuss our criteria for determining whether two patterns are approximately equivalent as well as the motivation behind our work. Finally, we provide results that validate our efforts.

General Terms

Algorithms, Experimentation, Performance

Keywords

Spatial association mining, pattern-set mining, approximation, contact maps

1. INTRODUCTION

Discovering important structures in molecular datasets has been the focus of many recent research efforts in biological and chemical informatics. These efforts have targeted, for example, substructure analysis in small molecules and macromolecules such as proteins and nucleic acids, as well as material defect analysis in molecular dynamics simulations [13; 24; 7; 36; 11; 12; 26; 31]. Most of the work in discovering substructures in molecules has focused on representing the molecule as a 3-D graph and finding frequent subgraphs that are contained within [16; 33; 17; 21; 14; 6; 5; 32]. A problem occurs, however, when trying to determine whether two subgraphs are equal. In general, the problem of subgraph isomorphism is NP-complete, and as such, any efficient solution will require the use of heuristics or similar techniques to keep the running time manageable. Another approach used recently has been to represent a molecule as a contact map. The principle behind a contact map is to only represent the interactions between points, as opposed to an entire three-dimensional structure. Using such a representation reduces the dimensionality of the problem down to a more manageable size. A contact map is essentially an adjacency matrix, where matrix position *A(i,j)* will be set to 1 if residues (or atoms, depending on the resolution used) are "in contact" and 0 otherwise. The definition of "in contact" can change depending on the data being examined, but most applications use the Euclidian distance between two atoms, with a user-specified distance as a cutoff threshold.

We are interested in using contact maps to represent protein molecules. A protein is composed of a series of amino acids. This sequence is commonly referred to as the protein's primary structure. When placed in aqueous solution, a protein will "fold" into a three-dimensional structure, with the structure uniquely determined by the protein's sequence. While the exact steps that a protein undergoes while folding is unknown, it is known that a protein will fold into a series of substructures (α-helices and β-sheets), referred to as secondary structures and these substructures will fold into larger structures, called tertiary structures. Trying to determine the steps, or the pathway that a protein follows while folding remains an open problem in biology. In the protein domain, contact maps are useful in that they provide a visual representation of the secondary structures that make up a protein molecule. For instance, α-helices show up as thick bands on the main diagonal and β-sheets appear as bands either parallel or anti-parallel to the main diagonal, depending on the conformation of the secondary structure. In addition to reducing the dimensionality of the dataset and providing a method of visualization, representing a molecule as a contact map also allows for the efficient use of bit-wise operations during implementation.

In the protein domain, contact maps have been used for a number different applications, including molecular alignment, fold prediction, and the discovery of *non-local structures* (or patterns) [10; 9; 18; 29]. We are also interested in mining contact maps to discover non-local structures, however, we intend to look for *spatial relationships* between the patterns across multiple contact maps, not just within a single map. In a contact map, non-local patterns are indicative of interactions between the tertiary structures of a protein molecule. Thus, if we can find relationships between non-local patterns across several different contact maps, we might be able to shed some insight into the protein folding problem. Finally, we would like to cluster a set of proteins based on the relationships that we generate to determine whether there is any correlation between those relationships and a molecule's function. By incorporating information from a database like the Structural Classification of Proteins (SCOP) database [23], which classifies proteins based on their 3-D structure, we would like to make predictions

*This work was supported by NSF Career Grant IIS-0347662 and NSF Grant CCF-0234273

about a molecule's function based on its contact map. In addition, we would also like to see if the reverse is true: whether we can use the spatial relationships we discover to generate rules that can be used to describe a protein's function.

One problem with protein data is that it is inherently noisy. Therefore, one cannot treat the distances between atoms as absolute. Two different crystallizations of a protein might yield slightly different coordinates for a molecule and lead to different contact maps. Thus, we need to derive a method to discover approximate patterns, and define a notion of approximate equivalence.

In this paper we present an algorithm for generating spatial associations based on approximate patterns within a contact map. We give an overview of the related work in this field in Section 2. Our algorithm is presented in Section 3. We show our experimental results in Section 4 and provide our conclusions and future goals in Section 5.

2. RELATED WORK

A great deal of work has gone into the area of using contact maps in the protein domain. Hu et al. have looked into mining contact maps to generate frequent dense patterns [10]. Additional work has gone into mining non-local patterns in contact maps [9]. A number of researchers have attempted to structurally align two protein molecules by solving the the Maximum Contact Map Overlap problem [18; 4]. Others have shown that it is possible to reconstruct a protein's structure from a contact map even in the presence of a large amount of noise [30]. Zhao and Karypis have developed a technique to predict a molecule's contact map using Support Vector Machines (SVM), which can be beneficial in fold recognition and structure determination [35]. Several groups have looked into using contact maps and the principles of energy minimization to create a system to recognize a protein's folds [28; 19; 3]. Finally, contact maps have been used to create a heuristic solution to the protein fold prediction problem [29].

A number of researchers have been looking into the area of spatial association mining. Koperski et al. [15] have used the technique to find association rules in geographic information system (GIS) databases. The Spatial Mining for Data of Public Interest (SPIN!) project[1] has looked into the mining of GIS databases as well as other areas such as census data. These efforts have primarily looked at defining associations based on a set of spatial predicates. Others have proposed methods to discover metric-based spatial associations [22; 8], though these metrics are defined over points, not over objects.

3. ALGORITHM

In this section, we describe the major steps taken towards generating approximate pattern-associations in contact maps. An overall description of our algorithm is given in Figure 1. We will now describe each step in further detail.

3.1 Contact Map Generation

When generating a contact map, one can examine the distances between individual atoms, between residues, or even secondary structures, depending on the resolution desired.

[1] http://www.ccg.leeds.ac.uk/spin/index.html

1. Generate contact maps for protein molecules.
2. Identify maximally connected patterns for each map and represent each pattern as a *feature vector*.
3. Cluster the *feature vectors* into *approximately* equivalent groups using a *k-means*-based clustering method.
4. Choose the optimal number of clusters based on the clustering entropy.
5. Re-label each pattern in a contact map with its corresponding cluster label.
6. Create an *occurrence vector* for each occurrence of a labeled pattern.
7. Generate *spatial pattern-associations* based on the *occurrence list*

Figure 1: Pattern-Association Mining Algorithm

We chose to look at the distances between the α-carbons (C_α) of each amino acid. Thus, for a protein with N amino acids, we will generate a binary matrix of size $N \times N$. We define each position *C(i,j)* in the contact map in the following manner: Given two amino acids a_i and a_j, if $d(a_i, a_j)$, the Euclidian distance between the C_α atoms of a_i and a_j, is less than a user-specified threshold t, then $C(i, j) = 1$. Otherwise, $C(i, j) = 0$. Since a contact map is symmetric across the diagonal, we only examine half of the matrix when running our experiments. In addition, we ignore the protein backbone (the 1s on the diagonal) in all of our tests.

3.2 Extracting Maximal Connected Patterns

Every non-edge position *(i,j)* in a contact map has eight *neighbor bits* at locations $(i\pm1, j\pm1)$, $(i\pm1, j)$, and $(i, j\pm1)$. For edge positions, we assume the out-of-bound neighbor bits to be 0. We define a *bit pattern* or simply, a *pattern*, to be an arbitrary collection of 1 and 0 bits. A *connected pattern* is a pattern where, for every position that contains a 1, there is a neighboring bit that is also set to 1. The *minimum bounding rectangle (MBR)* is the minimum rectangle that encloses a pattern. We define a *maximally connected pattern* (also referred to as a *feature* in this article) to be a connected pattern p where every neighbor bit not in p is 0. We apply a simple region growth algorithm to identify all *maximally connected patterns* within every protein contact map in a dataset. Connected patterns of size 1 are not considered.

3.3 Generation of Feature Vectors

One of the issues raised when working with contact maps is how to represent a feature. Several different methods have been employed, each with varying success. One simple approach is to represent a feature as a set of positions *(i,j)* where each position in the set corresponds to a 1 in the original pattern [9]. This method works best when the patterns are sparse and spread over a large area. An alternative approach is to represent a pattern as an array of bit strings [10]. Both of these approaches work well when the patterns examined are relatively small. When the number of patterns and the patterns themselves are large, however, both representation methods require an unacceptable amount of storage space.

In this work, we often deal with features that contain thousands of 1s and since we are attempting to identify non-local

features across a large set of contact maps, we must store thousands of unique (and potentially large) patterns. It is clear that representing every 1 in a pattern is not a viable option. Therefore, we must use an approximate representation, one that captures a feature's major characteristics, is storage-efficient and is easily explainable and interpretable. We propose a method using the following fields to represent a pattern:

- *Height*: the number of rows contained in a pattern's MBR.
- *Width*: the number of columns in a pattern's MBR.
- *NumOnes*: the number of 1s in a pattern.
- *Angle*: the general linear distribution trend of all the 1s in the pattern within its MBR.
- *xStdDev*: the standard deviation of all the 1s' x-coordinates (this quantifies how the 1s spread along the x dimension).
- *yStdDev*: the standard deviation of all the 1s' y-coordinates.

Thus, a *feature vector* is a 6-tuple consisting of the above fields. The reason that we require both the height and width of a pattern's MBR instead of simply using the area is that we believe two patterns should be considered "different" when one MBR has a different number of rows or columns than the other, even if both MBRs have the same area. To compute the angle of a connected pattern we use the least-squares method to estimate the slope of a linear regression line. For a pattern containing n 1s, we denote the positions of the 1s as: $(x_1, y_1) \ldots (x_n, y_n)$. The least-squares method then estimates the slope (β_1) as:

$$\beta_1 = \sum_{i=1}^{n}((x_i - \overline{x}) * (y_i - \overline{y}))/\sum_{i=1}^{n}((x_i - \overline{x})^2)$$

Notice that β_1 is a real number in the range $\pm\infty$. This makes the comparison of two patterns' β_1 values difficult. Therefore, we convert the β_1 value of each pattern to its corresponding angle off the x-axis. After this conversion, the values of an angle are in the range of [0, 180). After the feature generation step, we are left with a set of feature vectors. We then normalize those vectors to decrease the impact of attributes with a large range of values.

3.4 Clustering

Our next step is to place the maximally connected patterns into approximately equivalent groups. Two common methods can be used to do this: classification and clustering. Classification is a supervised procedure which requires the user to pre-label a set of connected patterns in order to build up a set of decision rules. Such a requirement is difficult to meet because it requires a domain expert's participation, which is impractical in this case due to the large number and variety of features that are generated. Thus, clustering is used to group the features into *approximately equivalent groups*. Besides being an unsupervised procedure, by using an appropriate similarity metric, a clustering algorithm can place similar elements together while separating dissimilar items. We consider each group generated from the clustering procedure to be an approximately equivalent pattern group. A pattern is assigned to the group to which its feature vector has the highest similarity.

When determining the similarity between two patterns, we believe the most significant parameters of the feature vector to be the dimensions of the MBR. As a result, similar patterns should have similar-sized MBRs. We ensure this property by weighing the *height* and *width* attributes more than the others when clustering the feature vectors.
Once all the vectors have been clustered, we re-label each pattern with its corresponding cluster label. By re-labeling the patterns, we are left with a much smaller set of feature types, as opposed to a large number of individual features. This enables us to study the spatial relationship between patterns in a more effective and efficient manner. We are guaranteed that the information "lost" by our clustering method is minimized by our clustering scheme, discussed next.

Quantitatively Measuring the Clustering Quality

After the completion of any clustering algorithm, one should measure the "goodness" of the clusters. Informally, a "good" cluster is one that has high *intra*-cluster similarity and low *inter*-cluster similarity. If one takes the opposite view and measures the quality of a cluster based on the *dissimilarity* of the features within that cluster, one is left with the quality measure of *entropy*. The higher a cluster's entropy, the greater the degree of dissimilarity among the members of that cluster. Given a set of events e_1,e_2,. . ., e_n, where the probability of an event $e_i's$ occurrence is p_i, then the entropy (H) of the set is defined as:

$$H = -p_i * log_2(p_i)$$

Each feature vector in our dataset is composed of 6 attributes. When computing the entropy of a cluster, we need to compute it in such a way that ensures every attribute contributes to the final entropy value. In addition, once we have computed the entropy for each cluster, we cannot simply sum them to determine the goodness of a clustering run because some clusters are larger than others and thus should not carry the same weight. We propose a goodness measure that weighs each individual cluster's entropy by that cluster's size in relation to the size of the entire dataset. Thus, for a dataset of N records, partitioned into k clusters, c_1,. . ., c_k, where a cluster c_i $(1 \leq i \leq k)$ has an individual entropy H_i and contains N_i elements, then the total entropy of this clustering is given by the following formula:

$$H = \sum_{i=1}^{k} H_i * (N_i/N)$$

Now we look at computing the individual entropy of a cluster. We compute the entropy of a cluster using the non-normalized feature vectors. As stated previously, each feature vector is composed of 6 attributes. The first three attributes, *Height*, *Width* and *NumOnes* are discrete, while the remaining attributes, *Angle*, *xStdDev* and *yStdDev* are continuous. For a discrete attribute, we take every unique value of that attribute in the cluster as a single event. We count the total number of occurrences for that value and then compute the probability of this value by dividing the number of times it occurred by the number of feature vectors in the cluster. For the *Angle* attribute, we assume it has a uniform distribution and compute its entropy as follows:

1. For all the vectors in a cluster, compute the minimum and maximum angle values, denoted $Angle_{min}$

and $Angle_{max}$.
2. Partition the interval [$Angle_{min}$, $Angle_{max}$] into equi-width intervals of length 30.

Each interval is treated as a single event, and we are able to compute the entropy for the *Angle* attribute exactly the same way as we compute it for a discrete attribute. For the other two attributes, *xStdDev* (σ_x) and *yStdDev* (σ_y), we assume they follow a Gaussian distribution and therefore their entropy can be computed by the following formula [27]:

$$H(x) = \log_2(\sqrt{2\pi e} * \sigma_x)$$

Finally, the entropy of a cluster is computed as:

$$H_i = \sum_{i=1}^{6} H(Attribute_i)$$

Choosing the Cluster Size

In order to pick the "optimal" number of clusters for grouping our feature vectors, we run the k-means clustering algorithm [20] on different k values. We then compute the entropy for each run using the method described above and finally, we plot the entropy vs. the number of clusters and choose a value k where the entropy plot begins to show a linear trend.

3.5 Mining Spatial Pattern-Associations

Creation of an Occurrence Dataset

Once the number of clusters has been chosen, we re-label each pattern with its cluster label, i.e. the cluster ID, and for each occurrence of a pattern in a contact map, we create an entry with the following fields:

- p_i: the cluster ID of the pattern.
- m_i: the contact map ID where the pattern is located.
- r_i: the row number of the pattern's MBR's upper left bit within the contact map.
- c_i: the column number of the pattern's MBR's upper left bit within the contact map.
- h_i: the height of the pattern's MBR at location (r_i, c_i) within the contact map.
- w_i: the width of the pattern's MBR at location (r_i, c_i) within the contact map.

The above representation is analogous to the *vertical format* structure used for frequent association mining [34]. The vertical format allows us to efficiently generate spatial pattern associations, as we will see shortly. From this point on, we only deal with the re-labeled patterns.

Computing Pattern Distance

Before we define the problem of spatial pattern-set mining, let us first define how to compute the distance between two connected patterns. The distance between two patterns p_1 and p_2 is defined only if they occur in the same map. Two types of metrics can be used to compute the distance between two patterns, with the first type defined over their feature vectors and the second over their spatial shapes and locations in a map. We do not consider the first type as it does not reflect the spatial distance between two patterns. Several distance metrics are available based on spatial shape and location. They include the Hausdorff distance [1], the distance between the center 1-bits in both patterns, the shortest distance between two 1-bits from each pattern, and the distance between two patterns' MBRs.

In this work, we use the last metric, the distance between two patterns' MBRs. There are several reasons that we use such a distance metric. It gives an approximate spatial distance between two patterns and is easy to explain. Also, it has a better scalability compared to other metrics such as the Hausdorff distance. There are three different cases that can occur when computing the distance between two MBRs:

- *Case 1 (Overlap)*: If two MBRs are overlap, then the distance between them is 0
- *Case 2 (Parallel)*: If two MBRs are parallel to each other, then the distance between them is the Euclidian distance between the two closest edges.
- *Case 3 (Other)*: If two MBRs are neither overlapping nor parallel, the distance between them is the minimum Euclidian distance between any pair of vertices.

Spatial Pattern Creation

Given n spatial patterns $P = \{p_1, p_2, \ldots, p_n\}$, and k 2-D maps $M = \{m_1, m_2, \ldots, m_k\}$, a spatial dataset D can be described as: $D = \{E_i\}$,
where $E_i = <p_i, m_j, r_i, c_i, h_i, w_i>$, and $p_i \in P$ and $m_j \in M$
In the context of contact maps, each E_i corresponds to one occurrence of a maximal connected pattern, with p_i being the pattern's cluster ID. Given a spatial dataset D as described above, we define the problem of spatial association mining as the identification of associations which are not only frequent over the 2-D maps in M, but also meet a user-specified pattern distance criterion.

A *pattern association* or *pattern-set* S of size k is one that consists of k patterns $\{p_0, p_1, \ldots, p_{k-1}\}$, where $p_i \in P$ and $0 \leq i \leq (k-1)$ and $distance(p_0, p_i) \leq maxDist$, where $0 < j \leq (k-1)$ and $maxDist$ is a user-specified distance threshold. Thus, a pattern-set S covers a *circular* area on a 2-D map, with its center located in p_0 and its radius no greater than $maxDist$. p_0 is also called the *center pattern* of S. Unless otherwise noted, the first pattern in S is its center pattern. A pattern-set of size k is denoted as a *k-set*. The *support* of a pattern-set is the percentage of contact maps in the dataset in which it occurs. A *frequent pattern-set* is one whose support is greater than or equal to a user-specified parameter *minSupport*. Note that when we say a pattern association is in a given map, we currently do not consider its specific occurrences in the map, just that it exists. We plan to integrate information regarding in-map occurrences into our future work.

A pattern set S_1 is a *sub-pattern-set* of S_2, if: $\forall p_i \in S_1, p_i \in S_2$ and they have the same center pattern. Accordingly, S_2 is a *super-pattern-set* of S_1. For instance, <A,B,C> is a sub-pattern-set of the set <A,B,C,D>. A *maximal frequent pattern-set* is one that does not have a frequent super-pattern-set.

Pattern-Set Generation: Basic Algorithm

The basic principle of our pattern-set generation algorithm is to generate pattern-sets in an increasing level-wise manner, starting with pattern-sets of size 1. The first step is to identify all the individual patterns that reside in at least *minSupport* contact maps. Given that all pattern occurrences are organized in the vertical format representation,

this step is fairly easy to implement. The second step is to generate all frequent 2-sets, the third step to generate k-sets where $k > 2$, and the last step is to generate only maximal pattern sets, which is optional.

The *anti-monotonicity* property of a frequent spatial pattern-set is used to facilitate the generation of frequent k-sets with $k \geq 2$. The property of anti-monotonicity states that a pattern-set cannot be frequent if one of its sub-pattern-sets is infrequent. Therefore, when generating k-sets, we only need to consider those where all the *(k-1)*-sub-pattern-sets are also frequent. We define a *candidate pattern-set* as one where all its sub-pattern-sets are frequent.

In the basic algorithm, for a candidate 2-set $< p_i, p_j >$, a brute-force method is used to check whether it is frequent by examining all possible location combinations of p_i and p_j in a map. Such a method has very poor performance, given that a pattern can occur at multiple locations within a contact map.

A similar process is used to generate all frequent k-pattern-sets of size $k > 2$ by first generating candidate k-sets based on the frequent $(k-1)$-sets, then computing their support to see if they meet the *minSupport* threshold. Since only circular pattern-sets are considered in this work, we do not need to compute the pattern distance in this step. For example, if we know both $< A, B >$ and $< A, C >$ occur at location (m_i, r, c), where m_i is the ID of a map, and (r,c) is the location of the upper left bit of the MBR, then we know $<A,B,C>$ must also occur at (m_i, r, c), as we are sure that both B and C are within the distance of *maxDist* from A at (m_i, r, c). By the same token, if both $S_1 = \{s_0, s_1, \ldots, s_{k-1}, p\}$ and $S_2 = \{s_0, s_1, \ldots, s_{k-1}, q\}$ occur at position (m_i, r, c), then $S_3 = \{s_0, s_1, \ldots, s_{k-1}, p, q\}$ must also occur at (m_i, r, c).

Pattern-Set Generation: Optimizations

Two optimization techniques are employed to improve performance when generating all 2-pattern-sets. One quickly eliminates the maps that do not contain a certain candidate pattern-set, the other prunes patterns that are sure not to be within *maxDist* of a pattern.

To eliminate maps that do not contain a certain candidate 2-set, we assign each map in the dataset a unique ID that remains fixed throughout the entire algorithm. By doing this, we can record the IDs of the first and last map where a pattern or pattern-set appears, denoted as m_{min} and m_{max}. We can then define an interval $[m_{min}, m_{max}]$ which represents a superset of the maps in which a pattern or pattern-set exists. For a 2-set $< p_i, p_j >$, we intersect p_i's $[m_{min}, m_{max}]$ interval with that of p_j's. Then we decide whether a further step is needed to determine this set's support. If the intersected interval spans fewer than *minSupport* maps, such a set can be immediately discarded; otherwise, a further step is needed to decide whether it is frequent, which can be done much faster than the non-optimized version since we now have a much smaller set of maps to examine.

In order to prune away patterns that are certain to be greater than *maxDist* from a given pattern, the following method is used. For a given pattern p_i, we order its occurrences by (r_i, c_i) values. Once we have all of a single pattern's locations ordered in a map, the following step can be taken to prune far-away patterns. For a pattern p at location (r, c) in a map, where h is the height of p's MBR at this location and (r,c) is the location of the upper left bit of p's MBR, we first divide the map into the following 3 areas:

- A_1-the area above the row with row number $= \lceil$(r-*maxDist*)$\rceil$
- A_2-the area under the row with row number $= \lceil$(r+h+*maxDist*)$\rceil$
- A_3-the remaining area

We determine into which of these three areas another pattern q is located before computing its distance to p. If q lies in either in A_1 or A_2, we are sure that it cannot be close to p. Since a pattern's occurrences in a map are ordered, we can use a binary search to mark the last occurrence of q in A_1 and the first occurrence of q in A_3. Once this marking is complete, we only need to compute the distances from p to q between the two marked occurrences (i.e. if q lies in A_2).

The other optimization technique introduced to improve the performance is the usage of *equivalence classes*. An equivalence class is a collection of frequent pattern-sets, where all sets have the same *prefix*. A set's prefix is composed of all the patterns in a set except the last one. The size of an equivalence class is defined as the size of its corresponding pattern-sets. Obviously, all sets in an equivalence class have the same center pattern. For instance, suppose $P = \{A, B, C\}$. P would have the following size-2 equivalence classes: $\{< A, A >, < A, B >, < A, C >\}$, $\{< B, A >, < B, B >, < B, C >\}$, and $\{< C, A >, < C, B >, < C, C >\}$ (Note: we assume all the pattern-sets are frequent). One potential size-3 equivalence class for P is $\{< A, A, A >, < A, A, B >, < A, A, C >\}$.

The optimized algorithm to generate frequent pattern sets of size greater than 2 is as follows: We first partition all frequent 2-sets into equivalence classes. As demonstrated in [25], all equivalence classes are independent of one another. Therefore, we can work on one equivalence class a time to generate larger frequent sets. This allows us to efficiently mine frequent spatial associations in a large dataset, as we are dealing with equivalence classes individually instead of the whole dataset.

Evaluating Frequent Pattern-Sets

For a frequent pattern-set, one would like to define a measure of "usefulness." This measurement is often subjective and domain-specific. In the protein context, we propose using a pattern-set's entropy to measure its "usefulness." We do this by integrating the SCOP lineage information of a pattern-set's associated proteins. We realize several other public databases also provide a method of structure-based protein classification, and that their classifications for a given protein may disagree, but for the time being, we use the SCOP classification.

For each frequent pattern-set, we identify the list of proteins contained in that pattern-set. We then classify these proteins into different groups based on a protein's SCOP lineage. A protein's SCOP lineage is organized into 6 levels: $L_1 : class$, $L_2 : fold$, $L_3 : super\text{-}family$, $L_4 : family$, $L_5 : protein$, and $L_6 : species$. In our experiments, we look at the first 4 levels.

Once the N proteins contained in a pattern-set S are classified at a certain SCOP level, we compute the entropy to measure how well these proteins are distributed among different SCOP categories. Take *L1: class* as an example. L_1 is divided into 11 sub-classes, denoted $\{c_1, c_2, \ldots, c_{11}\}$. When computing the entropy for S at this level, we first

count the number of proteins in each sub-class, denoted $\{n_1, n_2, \ldots, n_{11}\}$. The entropy is computed as:

$$H(S) = \sum_{i=1}^{11} -(n_i/N) \times log_2(n_i/N)$$

The pattern-set generation algorithm provides a parameter that allows a user to specify a maximum entropy at a given SCOP level. As with other user-specified parameters, the value of this parameter differs from dataset to dataset and is determined empirically.

As we discussed in Section 2, there has been a great deal of work toward the mining of spatial associations. We feel that our work differentiates itself from existing efforts in a number of areas:

- Vertical format data representation: To the best of our knowledge, there has not been any work in spatial data mining that represents a database using a vertical format. Accordingly, we are the first to use "equivalence classes" to expedite the process of mining spatial data.
- Metric-based spatial associations: Unlike the spatial associations proposed in [15], where a spatial association is defined over a set of spatial predicates (such as *close_to()* and *west_of()*, which are pre-defined and can only approximately describe the relationships between spatial objects), in our work, the relationship between spatial objects in a pattern-set is accurately quantified by Euclidian distances.
- 2-D object-oriented spatial associations: Existing metric-based spatial association mining algorithms [22; 8], have defined their distance metrics over *points* instead of *objects*. In this case, however, using points instead of objects can lead to information loss. The distance metric implemented here functions over actual 2-D objects; in this case, MBRs. The metric quantifies the topological relationship between two MBRs when they overlap or are parallel to each other and takes into account the size of the MBRs otherwise.
- Quantitative measurement of "interestingness": An entropy-based measurement is proposed to indicate whether a particular pattern-set is "interesting."

4. EXPERIMENTAL RESULTS

In this section, we present the experimental results carried out on 3 different datasets.

4.1 Datasets

To generate our contact maps, we used proteins taken from the Protein Data Bank (PDB) [2]. We generated three different sets of contact maps using a cut-off distance between amino acids of 4.5Å, 6Å, and 7.5Å. Table 1 shows the datasets generated. Also given in the table are the number of unique feature vectors, the total number of feature occurrences and the average number of features per protein.

4.2 Clustering Results

To cluster the feature vectors, we used a k-means-based clustering algorithm, where the Euclidian distance between two feature vectors is used as the similarity metric. As mentioned previously, in order to choose an *optimal* number of clusters, we ran the clustering algorithm multiple times for each dataset with different values for k. Once we obtained the clustering results, we computed the entropy for each cluster and plotted the weighted sum of the entropy versus the number of clusters (k). Based on the plot, we chose the value of k where entropy became approximately linear. When this criterion leads to multiple solutions, we chose the one that has smaller $H(Height)$ and $H(Width)$, i.e. the one that gives a tighter clustering solution in terms of a connected pattern's height and width.

Figure 2 shows the entropy plots for the three datasets. Based on the entropy values and $H(Height)$ and $H(Width)$, 24 and 32 clusters are the k values selected for all three datasets. We evaluated both and the results using 24 clusters were always worse in terms of the quality of pattern-sets than those obtained using 32 clusters (see Section 4.3). One possible explanation for this result is that placing the features into 24 approximate equivalence groups eliminates the information that can distinguish one pattern from the other when 32 groups are used.

4.3 Evaluation of Circular Spatial Pattern Sets

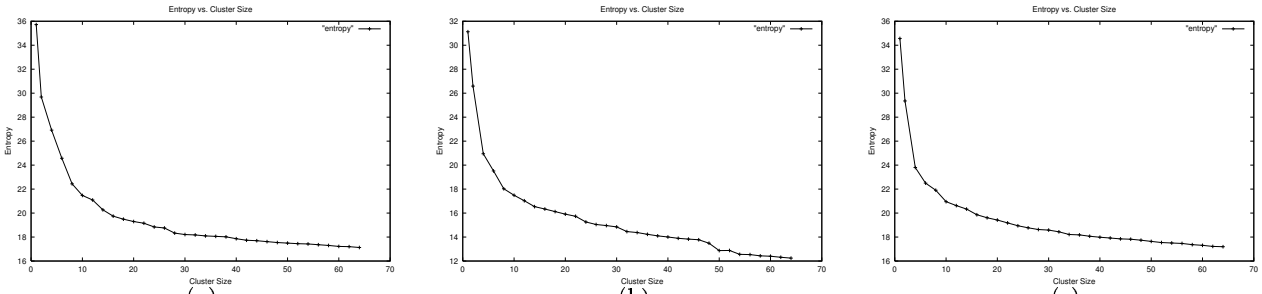
We only consider frequent pattern-sets whose SCOP lineage-based entropy is less than a certain threshold. In addition, when both a pattern-set and its super-pattern-set have an entropy below threshold, only the latter is kept for analysis. We denote the pattern-sets retained after these two steps as *low entropy maximal pattern-sets*, or *quality pattern-sets*. Three parameters are used to generate low entropy maximal pattern-sets, $minSupport$, $maxDist$, and the maximal pattern-set entropy at one or more SCOP levels. For convenience, we refer the last parameter as the *entropy cut-point*. If $maxDist$ is fixed, one can observe that the set of frequent pattern-sets derived at a lower $minSupport$ value must be a superset of the set of frequent pattern-sets derived with a higher $minSupport$. Therefore, in order to get a larger collection of pattern-sets, we set the $minSupport$ to a relatively low value of 0.02. As for the $maxDist$ parameter, various values were applied to each dataset. Based on the experimental results (SCOP-based entropy used as the main leverage), we empirically chose the following values for the 3 datasets: 32 for the 4.5Å dataset, 45 for the 6.0Å dataset and 55 for 7.5Å. Like the other parameters, the maximal SCOP-based pattern-set entropy is selected empirically. In our experiments, we only look at the first 4 SCOP levels. At a given SCOP level, we prefer a pattern-set that has a lower entropy, since a lower entropy usually indicates that a large percentage of the cluster's proteins belong to the same SCOP group. The entropy cut-points we chose corresponding to SCOP levels L_1 and L_2 are 2.0 and 3.2, respectively. For the other two levels, the entropy cut-point is 3.7.

Table 2 presents the number of low entropy maximal pattern-sets generated in each dataset. In order to compare results between the datasets, $minSupport$ is set so that the value of $(Numberofproteins) \times minSupport$ is the same across all datasets. Thus, 0.02 is used for the 6.0Å dataset and 0.01 for the other two.

A closer look at the results shows that the pattern-sets demonstrate different clustering ability. Nearly all the sets from the 4.5Å dataset consist of *Small proteins*, a sub-class at SCOP level L_1. In other words, for most pattern-sets, their associated proteins are classified as small proteins in SCOP. One example is the pattern-set (10 1 22) (Note that each value in a pattern-set corresponds to a clusterID obtained by clustering the individual feature vectors). Such a pattern-set was found in 22 proteins, which had the follow-

Threshold Distance	Number of Proteins	Number of Unique Features	Number of Total Feature Occurrences	Average Number of Features per Protein
4.5Å	2,169	23,148	74,396	34
6.0Å	1,090	36,967	175,525	52
7.5Å	2,122	53,817	410,041	122

Table 1: Contact Map Datasets



(a) (b) (c)

Figure 2: Clustering results: (a)4.5 Å (b)6 Å (c)7.5 Å

dataset	$H(L_1) \leq 2.0$	$H(L_1) \leq 2.0 and H(L_2) \leq 3.2$	#:$H(L_1) \leq 1.5$	#:$H(L_1) \leq 1.5$ *and* $H(L_2) \leq 1.7$)
4.5Å	433	142	19	0
6.0Å	214	93	47	11
7.5Å	1102	468	314	13

Table 2: Low entropy maximal pattern-sets

ing SCOP L_1 distribution: 2 *all-*β, 2 $\alpha + \beta$, 17 small and 1 designed. Unlike the 4.5Å dataset, the pattern-sets from the 7.5Å dataset tend to favor *all-*β proteins.

One other observation to be drawn about the pattern-sets generated for the 4.5Å dataset is that very few of them have $H_{L_1} < 1.5$. Empirically, we found that if a pattern set's entropy at a certain SCOP level was less than 1.5, then nearly all of its associated proteins belonged to the same SCOP sub-group. In addition, we found that the 4.5Å pattern-sets generally occur in very few proteins. One possible explanation for this behavior might be that the 4.5Å contact maps are too sparse to capture most structural information, while the 7.5Å maps are so dense that they introduce too much noise which would confuse the structural distinction for other types of proteins such as *all-*α, $\alpha + \beta$, etc.

The pattern-sets from the 6.0Å dataset have a relatively balanced distribution in terms of the number of protein groups they are able to distinguish. For instance, the pattern set (5 5 10 25 26), was found in 23 proteins with the following SCOP L_1 distribution: 21 *all-*α, 1 *all-*β and 2 α/β. (Please see Figure 3(b) for a visualization of the above pattern-set in the *all-*α protein 1a2f (ID from the PDB)). On the other hand, the pattern set (3 3 7 18) is good at distinguishing *all-*β proteins. Among the 49 proteins where this set occurs, the following SCOP L_1 distribution was found to exist: 1 *all-*α, 39 *all-*β, 2 α/β, 5 $\alpha + \beta$, 1 membrane and cell surface, and 1 designed. (Please see Figure 3(a) for an illustration of the above pattern-set in the *all-*β protein 1a25 (ID from PDB)).

One property illustrated by the pattern-sets from the 6.0Å dataset that does not exist in the the 4.5Å dataset is that there exists a collection of maximal pattern-sets that have low entropy (< 1.5) at the SCOP level L_1 (some are even close to 0) and that there exists a collection of maximal pattern-sets that have low entropy across the first four SCOP levels (see Table 2). An example pattern-set that presents both of these properties is (10 5 10 28). There are 23 proteins that contain this pattern-set, and their distribution at each of the first four SCOP levels is shown in Table 3.

4.4 Performance

In Figure 1, we give an overview of our mining algorithm. In this section, we provide a high-level analysis of the algorithmic running time of each step in that algorithm. Please note that we provide this analysis without proof. Before proceeding, however, we define several variables that will be used to help quantify the running time of each step. Let M be the number of protein molecules in a given dataset, and N_a be the number of amino acid residues contained in the largest protein molecule. N_f is defined to be the maximum number of occurrences of a feature in a contact map, N_u the total number of unique features in the dataset, and N_o the total number of occurrences of all the features in a dataset of M contact maps.

- The generation of contact maps occurs in order $O(M{N_a}^2)$ time.
- The time required to identify all the features in a dataset is $O(M{N_a}^2) + O(N_u)$. Of this, $O(M{N_a}^2)$ is the time required by the region growth algorithm to extract all the maximally-connected patterns in the

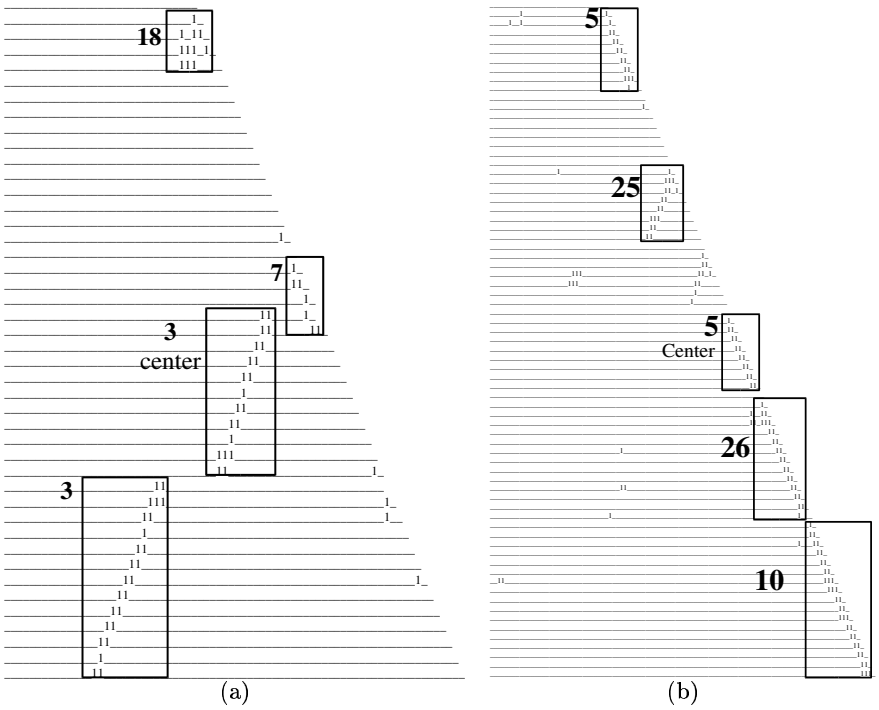


Figure 3: (a) The locations of pattern-set (3 3 7 18) in protein 1a25. (b) The locations of pattern-set (5 5 10 25 26) in protein 1a2f.

L_1:class	L_2:fold	L_3:super family	L_4:family
1 all β	1 SH3-like barrel	1 SH3-like barrel	1 SH3-domain
20 α/β	20 PLP-dependent transferases	20 PLP-dependent transferases	20 AAT-like
1 Peptides	1 Amyloid peptides	1 Amyloid peptides	1 Amyloid peptides
1 designed	1 Alpha-t-alpha	1 Alpha-t-alpha	1 Alpha-t-alpha

Table 3: The distribution of 23 protein containing the maximal pattern-set (10 5 10 28) with a low entropy across the first four SCOP levels.

maps and $O(N_u)$ is the time needed to identify the unique features. For the 7.5Å dataset, which contains over 400,000 feature occurrences, the time to complete this step was less than 20 minutes.

- The time required by one iteration of the *k-means*-based clustering algorithm is $O(kN_u)$ with k being the number of clusters. Of all the steps in the algorithm, this step took the longest, as we must collect a set of cluster solutions before we can decide on an optimal number of clusters. For each value of k, we let the algorithm run for 300 iterations to make sure any solution is approximately optimal. The time for one clustering run ranged from from 1 to 2 hours. The larger the dataset, the longer the clustering would take to complete, but the running time is not affected by the number of clusters that were generated. Once the number of clusters has been selected, it is possible to re-use the clustering solution to label the features for a new set of proteins, provided that the same distance threshold is used when generating the contact maps.
- The fourth step of the algorithm selects an optimal number of clusters based on the clustering entropy of a clustering run. It requires $O(N_u)$ time to compute the entropy for a given run.
- Re-labeling each pattern and creating an occurrence vector requires a running time of $O(N_o)$ for a particular dataset.
- The final step of the algorithm involves the actual generation of frequent spatial pattern-associations. The time required in this step can be decomposed into 3 phases:
  1. Discover all frequent 1-sets. This takes $O(N_o)$ time.
  2. Generate all frequent 2-sets, which can be done in $O(M{N_f}^2)$ time. This is so because in the worst case, one needs to compute the distance between every pair of feature occurrences in a map.
  3. Generate all pattern-sets of size greater than 2. It is hard to quantify the time required to generate a candidate set and all the frequent pattern-sets of a given size, because the time is not only impacted by the two user-specified parameters, *minSupport* and *maxDist*, but is also dataset-specific. As a result, we provide only the time required to confirm whether a candidate pattern-set is frequent, which is $O(MN_f)$. The time is linear to the number of occurrences since there is no need to compute the distance between two feature occurrences in this step.

  Please note that the performance analysis given here assumes the worst-case scenario. In practice, the two threshold parameters, *minSupport* and *maxDist*, can play a significant role in affecting the performance of this step.

5. CONCLUSIONS AND ONGOING WORK

In this paper we present our algorithm for discovering spatial relationships between approximately equivalent patterns in contact maps. While this work is still in the preliminary stages, we were able to find several interesting relationship rules. With further tuning of the algorithm parameters, we hope to find even more biologically-meaningful results.

We are currently extending this work in several aspects: First, we are implementing the other distance metrics described in this paper and intend to run an exhaustive comparison between them. Second, we plan to extend the pattern-set mining algorithm so that it can also generate pattern-sets of other spatial relationships. For example, we are interested in finding pattern-sets that have all pairs of involved patterns within a certain distance, and those that can be spatially arranged as a sequence, with the distance between any two adjacent patterns below a certain threshold. Finally, we plan to take into account a pattern-set's intra-map occurrences, including both the number of occurrences and the locations of those occurrences in a given map.

In addition, we would like to expand this work to other domains. We have access to several datasets containing information about the agricultural yield of farm fields for specific crops over a series of years. By generating contact maps for this dataset and applying our algorithm, it might be possible to determine whether there is any relationship between certain areas and specific crops.

6. REFERENCES

[1] M. J. Atallah. A linear time algorithm for the hausdorff distance between convex polygons. *Information Processing Letters*, 17:207–209, 1983.

[2] H. Berman, J. Westbrook, Z. Feng, G. Gilliland, T. Bhat, H. Weissig, I. Shindyalov, and P. Bourne. The protein data bank, 2000.

[3] Marco Berrera, Henriette Molinari, and Federico Fogolari. Amino acid empirical contact energy definitions for fold recognition in the space of contact maps. *BMC Bioinformatics*, 4(8), Feb. 2003.

[4] B. Carr, W. Hart, N. Krasnogor, J. Hirst, and E. Burke. Alignment of protein structures with a memetic evolutionary algorithm. In *2002*, 2002.

[5] D.J. Cook, L.B. Holder, S. Su, R. Maglothin, and I. Jonyer. Structural mining of molecular biology data. *IEEE Engineering in Medicine and Biology*, 20(4):67–74, 2001.

[6] H.M. Grindley, P.J. Artymiuk, D.W. Rice, and P. Willett. Identification of tertiary resemblence in proteins using a maximal common subgraph isomorphism algorithm. *J. of Mol. Biol.*, 229(3):707–721, 1993.

[7] L. Holm and C. Sander. Protein structure comparison by alignment of distance matrices. *J. Mol. Biol.*, 233:123–138, 1993.

[8] Wynne Hsu, Jing Dai, and Mong Li Lee. Mining viewpoint patterns in image databases, 2003.

[9] J. Hu, X. Shen, Y. Shao, C. Bystroff, and M.J. Zaki. Mining non-local structural motifs in proteins. In *BIOKDD 2002*, Edmonton, Canada, 2002.

[10] Jingjing Hu, Xiaolan Shen, Yu Shao, Chris Bystroff, and Mohammed J. Zaki. Mining protein contact maps.

[11] I. Jonassen, I. Eidhammer, D. Conklin, and W. Taylor. Structure motif discovery and mining the pdb. In *German Conference on Bioinformatics*, 2000.

[12] J. Kim, E. Moriyama, C. Warr, P. Clyne, and J. Carlson. Identification of novel multi-transmembrane proteins from genomic databases using quasi-periodic structural properties. *Bioinformatics*, 2002.

[13] R. King, A. Karwath, A. Clare, and L. Dehaspe. Genome scale prediction of protein functional class from sequence using data mining. In *The Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2000.

[14] I. Koch, T. Lengauer, and E. Wanke. An algorithm for finding maximal common subtopologies in a set of protein structures. *J. of Comp. Biol.*, 3(2):289–306, 1996.

[15] K. Koperski and J. Han. Discovery of spatial association rules in geographic information databases. In *Proc. 4th Int'l Symp. on Large Spatial Databases*, pages 47–66, 1995.

[16] M Kuramochi and G. Karypis. Frequent subgraph discovery. In *IEEE International Conference on Data Mining*, Nov 2001.

[17] M Kuramochi and G. Karypis. Discovering frequent geometric subgraphs. In *IEEE International Conference on Data Mining*, Dec 2002.

[18] Giuseppe Lancia, Robert Carr, Brian Walenz, and Sorin Istrail. 101 optimal pdb structure alignments: a branch-and-cut algorithm for the maximum contact map overlap problem. In *Proceedings of the fifth annual international conference on Computational biology*, pages 193–202. ACM Press, 2001.

[19] R. Najmanovich M. Vendruscolo and E. Domany. Protein folding in contact map space. *Physical Review Letters*, 82(656), 1999.

[20] J MacQueen. Some methods for classification and analysis of multivariate observation. In L.M. Le Cam and J. Neyman, editors, *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 281–297, Berkeley, 1967. University of California Press.

[21] E.M. Mitchell, P.J. Artymiuk, D.W. Rice, and P. Willett. Use of techniques derived from graph theory to compare secondary structure motifs in proteins. *J. Mol. Biol.*, 212:151–166, 1990.

[22] Yasuhiko Morimoto. Mining frequent neighboring class sets in spatial databases. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 353–358. ACM Press, 2001.

[23] A. G. Murzin, S. E. Brenner, T. Hubbard, and C. Chothia. Scop: a structural classification of proteins database for the investigation of sequences and structures. *J. Mol. Biol*, 247:536–540, 1995.

[24] W. Pan, J. Lin, and C. Le. Model-based cluster analysis of microarray gene-expression data. *Genome Biology*, 2002.

[25] Srinivasan Parthasarathy, Mohammed Javeed Zaki, Mitsunori Ogihara, and Sandhya Dwarkadas. Incremental and interactive sequence mining. In *CIKM*, pages 251–258, 1999.

[26] L. De Raedt and S. Kramer. The level-wise version space algorithm and its application to molecular fragment finding. In *Seventeenth International Joint Conference on Artificial Intelligence*, 2001.

[27] C. E. Shannon. A mathematical theory of communication. *SIGMOBILE Mob. Comput. Commun. Rev.*, 5(1):3–55, 2001.

[28] M. Vendruscolo and E. Domany. Protein folding using contact maps. *Vitamins and Hormones*, 58(171), 2000.

[29] Michele Vendruscolo and Eytan Domany. Efficient dynamics in the space of contact maps. *Folding & Design*, 3(5):329–336, 1998.

[30] Michele Vendruscolo, Edo Kussell, and Eytan Domany. Recovery of protein structure from contact maps. *Folding & Design*, 2(5):295–396, 1997.

[31] J. T. L. Wang, B. A. Shapiro, D. Shasha, K. Zhang, and C.-Y. Chang. Automated discovery of active motifs in multiple rna secondary structures. In *International Conference on Knowledge Discovery and Data Mining*, 1996.

[32] X. Wang, J.T.L. Wang, D. Shasha, B.A. Shapiro, I. Rigoutsos, and K. Zhang. Finding patterns in three-dimensional graphs: Algorithms and applications to scientific data mining. *IEEE Transactions on Knowledge and Data Engineering*, 14(4):731–749, jul/aug 2002.

[33] X. Yan and J. Han. gspan: Graph based substructure pattern mining. In *IEEE International Conference on Data Mining*, Dec 2002.

[34] Mohammed J. Zaki and Karam Gouda. Fast vertical mining using diffsets. In *RPI Technical Report 01-1*, 2001.

[35] Ying Zhao and George Karypis. Prediction of contact maps using support vector machines. In *Bioinformatics and Bioengineering*. IEEE, 2003.

[36] X. Zheng and T. Chan. Chemical genomics: A systematic approach in biological research and drug discovery. *Current Issues in Molecular Biology*, 2002.

Differential Association Rule Mining for the Study of Protein-Protein Interaction Networks

Christopher Besemann*
Computer Science Dept
North Dakota State University
Fargo, North Dakota 58105
christopher.besemann

Anne Denton
Computer Science Dept
North Dakota State University
Fargo, North Dakota 58105
anne.denton

Ajay Yekkirala
Biology Dept
North Dakota State University
Fargo, North Dakota 58105
ajay.yekkirala

ABSTRACT

Protein-protein interactions are of great interest to biologists. A variety of high-throughput techniques have been devised, each of which leads to a separate definition of an interaction network. The concept of differential association rule mining is introduced to study the annotations of proteins in the context of one or more interaction networks. Differences among items across edges of a network are explicitly targeted. As a second step we identify differences between networks that are separately defined on the same set of nodes. The technique of differential association rule mining is applied to the comparison of protein annotations within an interaction network and between different interaction networks. In both cases we were able to find rules that explain known properties of protein interaction networks as well as rules that show promise for advanced study.

General Terms

association rule mining, protein interactions, relational data mining, graph-based data mining, redundant rules

1. INTRODUCTION

Association Rule Mining (ARM) is a popular technique for the discovery of frequent patterns within item sets [1; 2; 13]. The technique has been generalized to the relational setting [18; 10; 22] including the study of annotations of proteins within a protein-protein interaction network [22]. In many bioinformatics problems, biologists are interested in comparing different sets of items. Rather than identifying patterns among protein annotations, biologists often want to contrast annotations of interacting proteins [25]. Going one step further, is also a want to contrast different network definitions to understand which experimental technique to use for which purpose.

Several definitions of protein-protein interactions have been introduced. For our study we concentrate on three: Physical interactions are determined through experiments such as the yeast-two-hybrid method [16; 30] and indicate a level of biochemical interaction. Genetic interactions are derived from in-vivo experiments in which the lethality associated with mutation of two genes is tested [26]. Domain-fusion interactions are detected in silico by comparing different species [19; 28]. Two genes in one species are labeled as interacting if they have homologs in another species and those homologs are exons of the same gene. Previous approaches to network comparison have studied each network in isolation and have compared statistics between networks [25; 27]. We use differential association rule mining techniques to identify rules that directly contrast the differences in annotations across interactions, and between different types of interactions.

Can differences be identified from standard ARM output? Assume, for example, that proteins with "transcription" as annotation are found to frequently interact with proteins that are localized in the "nucleus". This rule may be due to two independent rules, one that associates "transcription" and "nucleus" within a single protein, and others that represent a correlation of "transcription" and/or "nucleus" between interacting proteins. In fact, since trascription takes place in the nucleus this would make sense. We would not consider this a sign of a difference between interacting proteins. The same type of rule could, however, indeed stand for a difference. Consider the rule that proteins in the "nucleus" are found to interact with proteins in the "mitochondria". It can be expected that a single protein would not simultaneously be located in the "nucleus" and in the "mitochondria". We can therefore assume that the rule highlights a difference between interacting proteins and may identify an instance of compartmental crosstalk. This rule is significantly more interesting to a biologist than the rule relating "nucleus" and "transcription". It is much more expressive of the properties of the respective interaction network.

So far we have distinguished between the two examples on the basis of our biological background knowledge. Two approaches could be taken to translate the idea into a useful ARM algorithm. We could devise a difference criterion involving correlations between neighboring nodes and/or rules found within individual nodes. Such an approach would not benefit from any of the pruning that has made ARM an efficient and popular technique. Our algorithm takes an approach that makes significant use of pruning: Only those items are considered for the ARM algorithm for which each item in a set is unique to only one of the interacting nodes. The rule associating "transcription" and "nucleus" would thereby only be evaluated on those "transcription" proteins that are not themselves in the "nucleus", and those "nucleus" proteins, that are not themselves involved in "transcription".

There are other reasons why a focus on differences is more

*Authors' email: @ndsu.nodak.edu

Node		Edge	
ORF	Annotations	ORF0	ORF1
YPR184W	$\{< cytoplasm >\}$	YPR184W	YER146W
YER146W	$\{< cytoplasm >\}$	YNL287W	YBL026W
YNL287W	$\{< SensitivityTOaaaod >\}$	YBL026W	YMR207C
YBL026W	$\{< transcription >, < nucleus >\}$		
YMR207C	$\{< nucleus >\}$		

Figure 1: Initial Tables

effective for association rule mining in networks than a standard application of ARM on joined relations. Traditionally association rule mining is performed on sets of items with no known correlations. Interacting proteins are, however, known to often have matching annotations [27]. Using association rule mining on such data, in which items are expected to be correlated may lead to output in which the known correlations dominate all other observations either directly or indirectly. This problem has been observed when relational association rule mining is directly applied to protein networks [22; 4]. Excluding matching items of interacting proteins is therefore commonly advisable in the interest of getting meaningful results alone [4]. Matching annotations can be studied by simple correlation analysis, in which co-occurrence of an annotation in interacting proteins is tested. In the presence of such correlations, association rules are likely to reflect nothing but similarities between interacting proteins.

We use the concept of including only items that are unique to one of a set of interacting nodes to further address the task of comparing different interaction networks. In principle networks can be compared by studying each individually and comparing the results. When applying association rule mining to annotations in protein interaction networks, such an approach faces two difficulties. First, not all biological experiments have been done on all proteins. It is, therefore, safest to base a comparison of two networks only on proteins that show both types of interaction. Second, association rule mining gains its computational efficiency from item set pruning. Any test that is done at a later time removes rules that were produced unnecessarily. If the selection process can be converted to act on item sets themselves, pruning is restored. We demonstrate how the concept of unique items can be used to extract differences between networks.

2. DIFFERENTIAL ASSOCIATION RULES

We assume a relational framework to discuss differences within and between networks. The concept of a network may suggest use of graph-based techniques. Graph-theory typically assumes that nodes and edges have at most one label. Relational algebra on the other hand has the tools for the manipulation of data associated with nodes and edges. A relational representation of a graph with one type of nodes requires one relation for data associated with nodes, which we will call node relation, and a second relation that describes the reflexive relationship between nodes, the edge relation. To compare networks we will use multiple edge relations. Association rule mining is commonly defined and implemented over sets of items. We combine the concept of sets with the relational algebra framework by choosing an extended relational model similar to [13] . Attributes within this model are allowed to be set-valued, thereby violating first normal form. We go one step further by allowing sets of tuples, i.e. relations themselves, as attribute values. Consider a database with node relations $R_N(T, D)$ where T is a tuple identifier and D is a set of descriptors. Tuples in R_N have the form $< t_i, D_i >$ where D_i is a relation of descriptors $< d_j >$ (see figure 1 table Node for representation). Descriptors are tuples with just one attribute of domain $\mathcal{D}$. We call the $< d_j >$ descriptors to distinguish them from items. Items have a second attribute to identify their node of origin, see definition (3). We will call the sets of items that form the basis for association rule mining *basis set*.

Definition 1. A *single-node basis set* is identical to a set of descriptors $D_i \subseteq \mathcal{D}$. This definition is equivalent to the basic definition of an item set used in association rule mining [1].

Our goal is to mine relational basis sets that will be constructed from multiple descriptor sets that belong to the same tuple of a joined relation. An edge relation has two attributes $R_E(T_l, T_r)$, with T_l as well as T_r being foreign keys that refer to identifiers in one or more node relations (see figure 1 table Edge for representation). Edge relations can, in principle, have the alternate form $R_E(T_l, T_r, D^{(E)})$ with $D^{(E)}$ being a set of edge descriptors. We could split such a relation into a separate node relation as well as a standard edge relation as in [7].

Joined-relation basis sets are formed in multiple steps. Edge and node relations are joined through a natural join operation ($*$). Attribute names are changed [11] such that they are unique. We use this step to ensure that information about the origin of different attributes is maintained. Attributes are identified by consecutive integers to which we will refer as origin identifiers $g \in \mathcal{G} = \{0, ..., (n-1)\}$ where n is the number of node relations. This information will be used in a later step to actually modify the descriptors according to their origin before joined-relation basis sets are constructed from multiple descriptor sets.

Definition 2. A *joined-relation basis set* is derived through the following steps. A 2-node joined-relation is created by

$$R_{2N} \leftarrow \rho_{0.T,0.D}(R_N(T,D)) * \rho_{0.T,1.T}(R_E(T_l,T_r)) * \rho_{1.T,1.D}(R_N(T,D)). \quad (1)$$

Generalization to n-node joined-relations is straight forward. Note, however that we can have multiple alternatives. For

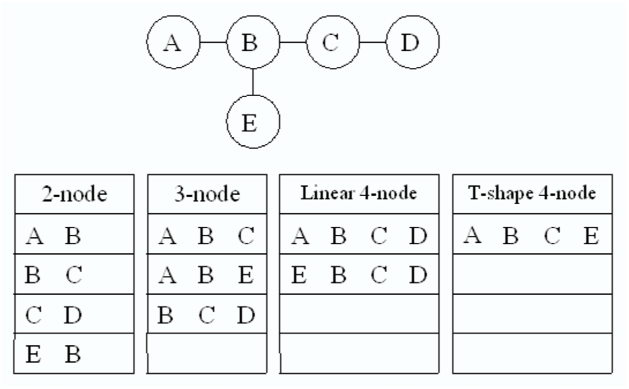


Figure 2: Representation of basis sets.

TID	Join	
1	$\{<0, cytoplasm>\}$	$\{<1, cytoplasm>\}$
2	$\{<0, SensitivityTOaaaod>\}$	$\{<1, transcription>, <1, nucleus>\}$
3	$\{<0, transcription>, <0, nucleus>\}$	$\{<1, nucleus>\}$

TID	Unique	
1	NULL	NULL
2	$\{<0, SensitivityTOaaaod>\}$	$\{<1, transcription>, <1, nucleus>\}$
3	$\{<0, transcription>\}$	NULL

Figure 3: Join and Unique

a 4-node joined-relation we can have

$$\begin{aligned} R_{4Nl} \leftarrow\ & \rho_{0.T,0.D}(R_N(T,D)) * \rho_{0.T,1.T}(R_E(T_l,T_r)) \\ & *\rho_{1.T,1.D}(R_N(T,D)) * \rho_{1.T,2.T}(R_E(T_l,T_r)) \\ & *\rho_{2.T,2.D}(R_N(T,D)) * \rho_{2.T,3.T}(R_E(T_l,T_r)) \\ & *\rho_{3.T,3.D}(R_N(T,D)) \qquad (2) \\ R_{4Ng} \leftarrow\ & \rho_{0.T,0.D}(R_N(T,D)) * \rho_{0.T,1.T}(R_E(T_l,T_r)) \\ & *\rho_{1.T,1.D}(R_N(T,D)) * \rho_{1.T,2.T}(R_E(T_l,T_r)) \\ & *\rho_{2.T,2.D}(R_N(T,D)) * \rho_{1.T,3.T}(R_E(T_l,T_r)) \\ & *\rho_{3.T,3.D}(R_N(T,D)). \qquad (3) \end{aligned}$$

Notice that in equation (2) the joining corresponds to a chain of 0-1-2-3 and in equation (3) there is a branch 1-2 and 1-3. Figure (2) illustrates forming basis sets given a simple network, we can see the alternatives at the 4-node join. Attribute renaming $\rho_{A_0 \ldots A_n}$ is used as defined in [11]. We then apply a Cartesian product of a relation consisting of a single tuple containing the origin identifier $< g >$ with each descriptor set individually. It converts the descriptors d_j into tuples $< g, d_j >$. g is the same origin identifier that is used as prefix in the attribute name

$$\begin{aligned} g.I_i &= < g > \times \{< d_0 >, ..., < d_k >\} \\ &= \{< g, d_0 >, ..., < g, d_k >\}. \qquad (4) \end{aligned}$$

Definition 3. An *item* is defined as a tuple $< g, d_j >$ where g is an integer which is the origin identifier and d_j is the descriptor value of an attribute.

Note that we will use an abbreviated notation for items in the results section ($g.d_j$ instead of $< g, d_j >$). A joined-relation basis set B_i is derived as the union of descriptor sets for each tuple identified by t_i of the joined relation. For a 2-node joined-relation basis set or 2-node basis set we have

$$\forall t_i \quad B_i = 0.I_i \cup 1.I_i. \qquad (5)$$

The set of all basis sets is $C = \{B_0, ..., B_m\}$ where m is the number of tuples in the joined relation an example of the product can be seen in figure (3 table Join) as the result of the operations to the relations in figure (1).

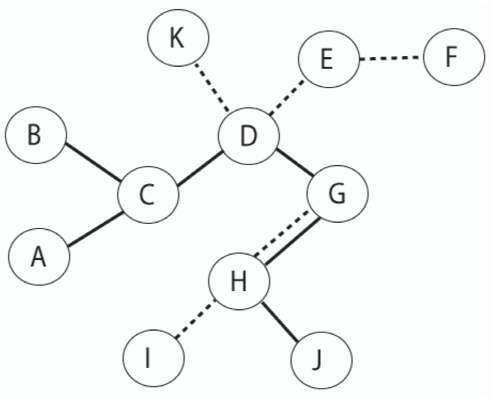
Definition 4. A *uniqueness operator* U is defined as follows. For each set-valued attribute on which it operates the set difference is computed between that attribute and the union of all other attributes of that domain.

$$\begin{aligned} & U(R_{nN}(t_i, \{0.I, ..., (n-1).I\})) : \\ & \forall t_i \ \ \forall_{j=0}^{(n-1)} \ j.I_i^U = j.I_i - \bigcup_{k=0, k \neq j}^{(n-1)} k.I_i \qquad (6) \end{aligned}$$

with $g.I_i$ defined as in equation (4).

Figure (3 table Unique) shows the results of the unique operation on the joined portion. In this paper the uniqueness operator is applied to all set-valued attributes of a joined-relation but other choices are possible, such as requiring uniqueness only across a subset of edges.

Definition 5. A *unique item basis set* is defined through the following steps. An n-node joined-relation is created as described in definition (2). The uniqueness operator is applied to all set-valued attributes. Then the Cartesian product is used to create item tuples, and the process continues as for joined-relation basis sets.



Transactions		
C	D	E
C	D	K
G	D	E
G	D	K
J	H	I

Figure 4: Left: Two graphs defined over the same set of nodes, Right: Network comparison basis set

Definition 6. A *network comparison basis set* differs from a unique node item basis set through the use of different edge relations. In the current paper we limit ourselves to 3-node network comparison basis sets. We only consider those edges that are unique to one of the network definitions. Edges that are represented in both networks are removed since they cannot give us information on differences between networks.

$$\begin{aligned} R_{3NC} \quad \leftarrow \quad & \rho_{0.T,0.D}(R_N(T,D)) * \rho_{0.T,1.T}(R_{E1}(T_l,T_r)) \\ & *\rho_{1.T,1.D}(R_N(T,D)) * \rho_{1.T,2.T}(R_{E2}(T_l,T_r)) \\ & *\rho_{2.T,2.D}(R_N(T,D)) \end{aligned} \tag{7}$$

Compare Figure (4) for a graphical representation of the extraction of a network comparison basis set. The other steps are done as for unique node item basis sets. The uniqueness operator is applied to all nodes. Assume for example a protein with a physical interaction between 0 and 1 and a genetic interaction between 1 and 2. Assume further a standard basis set as {0.A, 0.B, 1.C, 2.A, 2,D}. This would lead to a network comparison basis set of {0.B, 1.C, 2.D}. Examples of reported rules would be $0.B \rightarrow 1.C$ which is specific to the physical interaction and $1.C \rightarrow 2.D$ which is specific to the genetic. We limit the scope of our algorithm to rules that involve only one of the networks as definition (8). Any such rule will automatically represent a property that is in contrast to the other network.

Definition 7. Given the above definitions of basis sets, association rules are defined in their standard way. A rule has the form $X \rightarrow Y$ where X and Y are sets of items (see definition 3). The *support* of a rule is the probability $P(X \cup Y)$ within the set of all basis sets C. The *confidence* of a rule is the conditional probability $P(Y|X)$. The set of all items in the rule is an item set $I = X \cup Y$.

It is important to understand that any relational association rule depends on the context in which it was generated. A rule that involves only two nodes related by one edge can, in principle, be found in a 2-node join-relation and any higher order relation. The support and confidence will however vary depending on that context, and a rule that is strong in one context may not be so in another. We follow [7] in always using the lowest order possible. For network comparison purposes we need three entities to derive 2-node rules. See definition (6). The problems associated with multiple contexts leads us to the following definitions.

Definition 8. An item set J has *network comparison scope* if it represents all nodes that are related through one edge relation and no nodes that are related through a different edge relation. If the item set is furthermore unique, support and confidence based on this item set will reflect network properties that are specific to one type of network and not to any other network involved in the comparison. For instance given we have network A covering origin identifiers 0,1 and network B covering identifiers 1,2 then the itemset {0.nucleus,1.cytoplasm,2.transferase} would not be in network comparison scope but itemsets {0.nucleus,1.cytoplasm} and {1.cytoplasm,2.transferase} would be.

Definition 9. An item set J is *out-of-scope* if one or more nodes are not represented, i.e., if $|\pi_G(J)| < n$ where $||$ indicates the cardinality, π is the relational projection operation, G is the identifier attribute of the item tuples, and n is the number of node relations that were joined. In figure (3 table Unique) item sets for TID 1 and 3 are considered out-of-scope on the transaction level.

Definition 10. An item set J is *repetitious* if at least one descriptor occurs more than once, i.e., if $|\pi_D(I)| < |J|$ where π_D is the projection on the descriptor attribute. Two items are considered *repetitious* if they belong to the same joined-relation basis set, their origin identifier differs, and their descriptors are equal. Figure (3 table Join) item sets for TID 1 and 3 have repetitious items.

3. RELATED WORK

Oyama et al. [22] apply association rule mining to joined-relations of physical protein interactions and their annotations. This work notes the problem of what we term repetitious item sets but does not resolve it. Relational association rule mining has more generally been addressed in the context of inductive logic programming [10; 18; 17]. These approaches are very flexible and leave most choices up to the user. This paper, on the other hand, addresses the question of what specifications allow extracting meaningful rules. It is useful to notice that the major portions of differential rule mining can be imported to different frameworks including ILP.

Some biological publications have touched on the concept of comparing networks. The authors in [27] address aspects such as density of the networks and how well the genetic interactions predict physical interactions. Another work [23]

looks at correlation and interdependency characteristics between the genetic and physical networks. The distribution of annotations on an individual network is discussed in [25]. These approaches fall short of contrasting annotations in different networks. A further related research area is graph-based ARM [15; 21; 31; 6]. Graph-based ARM does not typically consider more than one label on each node or edge. The goal of graph-based ARM is to find frequent substructures in that setting.

Removal of a class of redundant rules is an important part of differential rule mining. Redundant rules have been studied, and closed sets [8; 33] have proven a successful approach to their elimination. Closed sets alone do not, however, address the problem of contrasting different nodes or networks. Since we know what kinds of rules we want to eliminate, it is significantly more efficient to do so at the relational join level. This strategy has the added benefit of correcting support and confidence of all rules to reflect only the contribution that is non-redundant to a combination of repetitious and out-of-scope item sets.

There are other areas of research on ARM in which related transactions are mined in some combined fashion. Sequential pattern or episode mining [2; 32; 24; 34] and inter-transaction mining [29] are two main categories. Some similarities in the formalism can be observed since we are also interested in mining across what can be considered transactions. A tuple in a joined-relation can ultimately be compared with sequences of transactions. Overall the goals of these approaches are too different to be applicable to our setting in any direct way.

4. IMPLEMENTATION

The differential association rule mining algorithm was implemented in a modular fashion. Three major parts are distinguished. Preprocessing (steps 1.-3.) includes application of the uniqueness operator U (see definition 4 in section 2). The actual item set generation (step 4.) is done based on sets of items that appear as regular sets to the ARM program. Results in this paper use the Apriori algorithm from Christian Borgelt [5]. Postprocessing (steps 5.,6.) does additional filtering at the item set and rule level.

Preprocessing includes the following tasks. For undirected graphs only one direction is typically included in data sets. We create both directions to ensure correct representation and then join the relations. Joined relations were created with different methods depending on the comparison type for input.

The uniqueness operator, U, from equation (6) was applied to all basis set relations (step 8.). If the operator U has removed all items related to any one of the entities the basis set is marked as deleted (steps 9.,10.). Such basis sets can never contribute to in-scope item sets or rules. The basis set is therefore not passed to the ARM method. We do, however, calculate support and confidence based on the full set of joined table basis sets by counting all basis sets. Once the basis sets are processed into the unique basis sets, standard Apriori is applied (step 4.).

Frequent item sets or closed item sets are returned as the usual result of Apriori. For undirected graphs symmetric versions of each item set are returned and have to be removed (step 5.). Input from Apriori is sent to the rule generation phase (step 6.). Item sets are tested if all entities

---

Number of nodes in the join relation: n
n-entity joined relation basis set: B_i
Set of basis sets C:$\{B_0,...,B_m\}$

Diff-ARM(n,$minconf$,$minsup$,C)
1. For undirected graphs represent each direction
2. Join relations and eliminate cycles
3. C^U=U_OP(n,C)
4. FreqSets=Apriori:FreqItemset_Gen(C^U,$minsup$)
5. For undirected graphs remove symmetric contributions
6. U_SCOPERULE($FreqSet$,n,$minconf$)

U_OP(n,C) Returns→ C^U
7. foreach transaction, $B_i \in C$
8. $B_i^U = U(B_i(\{0.I_i, ..., (n-1).I_i\}))$
9. foreach $j.I_i^U \in B_i^U$
10. if($j.I_i^U == \emptyset$) → mark tuple as deleted
11. $C^U += B^U$

U_SCOPERULE($FreqSet$, n, $minconf$)
12. foreach $J_i \in FreqSet$
13. if($|\pi_G(J_i)| == n$)
14. Apriori:Rule_Gen(J_i,$minconf$)
15. Apply rule filtering

Figure 5: Differential ARM Algorithm

are represented (step 13.). If not, the item set is removed as being out-of-scope. Rules are then produced as in standard ARM by processing the frequent item sets (step 14.). The algorithm concludes with a set of rules that satisfy the requirements from section 2. Rule results are additionally filtered so that any node does not have items in both the antecedent and the consequent of the rule after the final set (step 15.). The following equation defines this step for a given rule A→C:

$$\pi_G(A) \cap \pi_G(C) == \emptyset \tag{8}$$

4.1 Data sets

Our data consist of one node relation gathered from the Comprehensive Yeast Genome Database at MIPS [20; 9], gene_orf. The gene_orf node relation represents gene annotation data. Annotations are hierarchically structured, with hierarchies for function, localization, protein class, complex, enzyme commission, phenotype and motif. In any category, attributes are multi-valued and we pick the highest level in each hierarchy as descriptors. The relation contains the ORF identifier as key and the set of annotations related to that ORF as attribute (descriptor set).

We used three different definitions for protein-protein interactions which are undirected edges for yeast: physical, genetic and domain fusion. The physical edge relation was built from the ppi table at CYGD [9] where all tuples with type label of "physical" were used. The genetic edge relation was taken from supplemental table S1 of genetic interactions from [27] where both Synthetic Sick and Synthetic Lethal entries are used. Our third edge relation was the domain fusion set built from the unfiltered results posted from [28; 14]. The set was filtered to reflect only ORFs contained in our node relation.

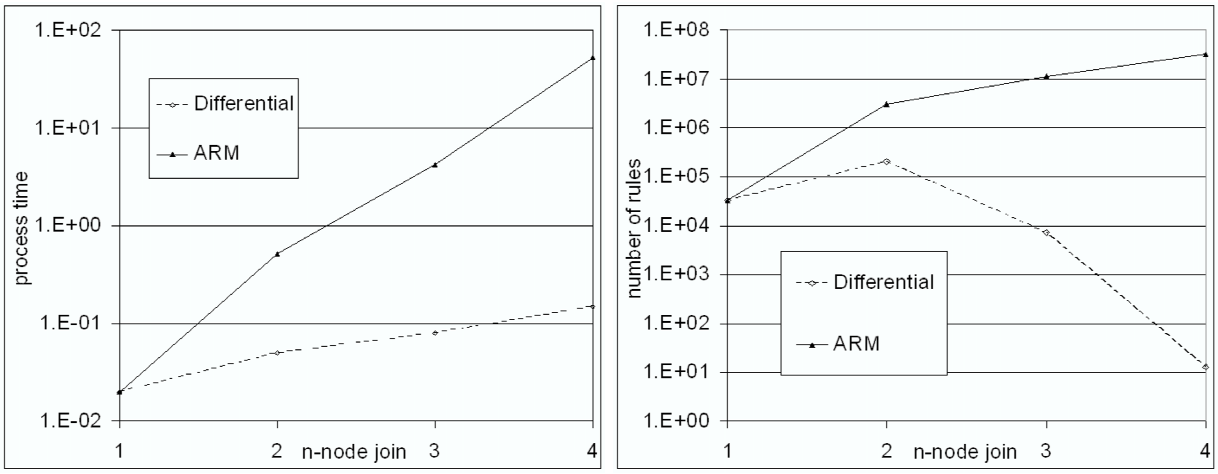


Figure 6: Left: Processing time, Right: Reduction in Number of Rules

4.2 Performance

Three contributions to the complexity have to be distinguished: preprocessing, Apriori and postprocessing. The most important contribution is the Apriori step. Since we did not modify the algorithm itself, changes in performance come from data reduction. The resulting improvement is highly significant. Figure (6) shows the processing time of the Apriori algorithm under a performance trial. Recorded is the time to generate frequent item sets for unique item basis sets of one to 4 nodes. We did not include time to load the database or print the rules. As seen, the differential ARM algorithm outperforms ARM by a factor of 100 in the 4-node setting. The reduction in the number of rules is even more significant. The difference between the number of rules in differential and standard ARM demonstrate how correlations dominate standard ARM output and thereby render it useless.

5. RESULTS

We will first look at an example of a rule that is strong based on the application of a standard ARM algorithm on joined tables but not so if only unique items are considered. A clear example is the rule mentioned in the introduction:

$$\begin{array}{lll} \{0.\text{transcription}\} & \rightarrow & \{1.\text{nucleus}\} \\ \text{support} = 0.29\% & & \text{confidence} = 28.38\% \end{array} \quad (9)$$

This rule is a consequence of a strong single-node rule together with correlations that are documented by a repitious rule

$$\begin{array}{lll} \{0.\text{transcription}\} & \rightarrow & \{0.\text{nucleus}\} \\ \text{support} = 0.70\% & & \text{confidence} = 69.59\% \end{array}$$

$$\begin{array}{lll} \{0.\text{nucleus}\} & \rightarrow & \{1.\text{nucleus}\} \\ \text{support} = 5.74\% & & \text{confidence} = 29.02\% \end{array}$$

Using the uniqueness operator changes the support of rule (9) to 0.02% and a confidence of 2.08%. We expect support and confidence to be lower when the uniqueness operator is applied, since annotations are removed. Strong rules in our data set do, however, in general have a support around 0.2-2% and confidence around 6-20%. Based on these numbers the rule (9) cannot be considered strong and ranks much lower in the new results.

For the remainder of this section we will report differential association rules and no standard ARM results. The following rule was found to be strong in the physical interaction network

$$\begin{array}{lll} \{1.\text{mitochondria}\} & \rightarrow & \{0.\text{cytoplasm}\} \\ \text{support} = 1.2\% & & \text{confidence} = 27.3\% \end{array}$$

This rule clearly corresponds to annotations that would not be expected to hold within a single protein but may hold between interacting ones. A protein located in the mitochondria would not have localization cytoplasm. We do, however expect compartmental crosstalk as studied in a paper by Schwikowski et al.[25] between those two locations. The observation confirms to us that we see rules that are sensible from a biological perspective. Comparison with [25] further helped us confirm some less expected rules such as

$$\begin{array}{lll} \{1.\text{mitochondria}\} & \rightarrow & \{0.\text{nucleus}\} \\ \text{support} = 0.72\% & & \text{confidence} = 16\%. \end{array}$$

We also found rules that have not yet been reported in the literature. The following rule was also observed within the physical interaction network

$$\begin{array}{lll} \{1.\text{ER}\} & \rightarrow & \{0.\text{mitochondria}\} \\ \text{support} = 0.21\% & & \text{confidence} = 6\% \end{array}$$

This rule was of interest particularly due to its comparatively high support. From a biological perspective one would not expect proteins in the endoplasmatic reticulum (ER) to physically interact with proteins in the mitochondria. To analyze the significance of the result we looked at some ORFs that support the rule. One pair was

(0.YLR423C: ER)
(1.YOR232W: mitochondria,
GrpE_protein_signature(PDOC00822),
Molecular_chaperones).

On further investigation it was found that GrpE along with a Molecular_chaperone is involved in protein import into the mitochondria [3]. This information leads to a hypothesis

Table 1: Statistics

Table	int/orf	max int	#>20	#int
physical	3.55	289	73	14672
genetic	7.88	157	93	8336
domain fusion	44.6	231	305	28040

that YLR423C could be aiding the import mechanism or be interacting with the chaperone. This example demonstrates how differential association rules can provide insights into the functioning of the cell and can lead to further studies.

5.1 Differences Between Interaction Types

We will now look at rules that derive from the network comparison formalism of definitions (6) and (8) (inter-network comparison). Given multiple types of protein-protein interactions we look for significant differences to aid in the understanding of cellular function and as well as the properties and uses of the networks. In this paper we consider pairs of networks for inter-network comparisons (physical and genetic, physical and domain fusion, domain fusion and genetic) and join the two edge relations to form a network comparison joined relation (definition 6).

The networks do not show a significant overlap, i.e., it is very common that for any given physical interaction between two proteins there will be no genetic interaction [27]. Strict network overlap for each network pair is: physical-genetic 14 transactions, physical-domain fusion 52 transactions, genetic-domain fusion 128 transactions. There are no transactions that overlap for all three. Our comparision instead uses partial overlap of the networks. Table 1 shows that even the statistical properties of the networks differ significantly: the average number of interactions of proteins that show at least one interaction varies from 3.55 in the physical network to 44.5 in the domain fusion network. Comparison of annotations across those networks has to compensate for such differences. The process of joining relations ensures that each protein that is considered for a physical interaction will also be considered for a genetic interaction.

Before looking at details of individual rules we will make some general observations regarding the number of rules we observed for different combinations of networks. When comparing physical and genetic networks we found about one order of magnitude more strong rules relating to the physical network compared with the genetic network. Physical interactions also produce the stronger rules when compared with domain fusion networks. That means that the physical network allows the most precise statements to be made. When comparing the domain fusion and the genetic network no major difference was found. That suggests that physical interactions reflect properties of the proteins better than either of the other two.

These rules are among the top 100 generated for the physical-domain fusion set. Some specific examples of interesting rules from this study are as follows:

{1.Fungal_Zn(PDOC00378)} →
{2.Zinc_finger_C2H2_type_domain(PDOC00028)}
support = 0.48% confidence = 76%

This rule was found to be supported in the domain fusion interaction set but not among the physical interactions. The motif of ORF 1 is a fungal Zinc-cysteine domain present in many transcription activator proteins which bind DNA in a zinc-dependent fashion. The motif of ORF 2 is a zinc finger which also binds DNA and commonly has cysteines and Histidine residues in them [12]. This rule tells us that the confidence of assuming a domain-fusion interaction between the fungal zinc domain and the zinc finger motif is 76%, not considering cases in which a zinc finger is also involved in a physical interaction. Further studies would be necessary to decide if the absence of a physical interaction is due to a problem with annotations or if those two proteins really do not interact. The second rule is supported by the physical network but not the domain fusion network

{0.ABC_trans_family_signature(PDOC00185)} →
{1.ATP/GTP_binding_site_motif_A(PDOC00017)}
support = 0.45% confidence = 90%

ORF 0 has the motif of an ABC transporter signature which implies it is an ABC transporter coding sequence. ABC transporters have conserved ATP binding domains as the motif in ORF 1 and help in either the import or export of molecules utilizing ATP as the energy molecule for the process [12]. From the rule we can see that these two domains physically interact but are never represented by a single gene. This supports the observation that the ATP binding domain is found in many other proteins as well [12] and both functions are combined through interactions at the protein level rather than at the genetic level. This observation would also warrant further studies.

6. CONCLUSIONS

We have described the novel concept of differential association rules. The goal of this technique is to highlight differences between items belonging to different interacting nodes or different networks. We demonstrate that such differences would not be identified by application of standard relational ARM techniques. Our technique is highly efficient and effective. It follows the ARM spirit by gaining its efficiency from a pruning step that is included even before the frequent item set generation step. We apply our framework to real examples of protein annotations and interactions. Results were able to confirm expected biological knowledge as well as identifying as yet unknown associations that were successfully supported by further inspection of the data. We have thereby provided a new tool that has potential for most network settings, and have demonstrated its successful application to bioinformatics.

7. ACKNOWLEDGMENTS

This material is based upon work supported by the National Science Foundation under Grant No. #01322899. Additional thanks are expressed for valuable feedback from the anonymous reviewers of this paper.

8. ADDITIONAL AUTHORS

Ron Hutchison & Marc Anderson
Biology Department NDSU
email: ron.hutchison & marc.Anderson @ndsu.nodak.edu

9. REFERENCES

[1] R. Agrawal, T. Imielinski, and A. N. Swami. Mining association rules between sets of items in large databases. In *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data*, pages 207–216, Washington, D.C., 26–28 1993.

[2] R. Agrawal and R. Srikant. Mining sequential patterns. In *Eleventh International Conference on Data Engineering*, pages 3–14, Taipei, Taiwan, 1995. IEEE Computer Society Press.

[3] A. Bateman, L. Coin, R. Durbin, R. D. Finn, V. Hollich, S. Griffiths-Jones, A. Khanna, M. Marshall, S. Moxon, E. L. L. Sonnhammer, D. J. Studholme, C. Yeats, and S. R. Eddy. The pfam protein families database. *Nucleic Acids Research: Database Issue*, 32:D138–D141, 2004.

[4] C. Besemann and A. Denton. Unic: Unique item counts for association rule mining in relational data. Technical report, North Dakota State University, 6, 2004.

[5] C. Borgelt. Apriori. *http://fuzzy.cs.uni-magdeburg.de/~borgelt/software.html*, accessed August 2003.

[6] D. J. Cook and L. B. Holder. Graph-based data mining. *IEEE Intelligent Systems*, 15(2):32–41, 2000.

[7] L. Cristofor and D. Simovici. Mining association rules in entity-relationship modeled databases. Technical report, University of Massachusetts Boston, 2001.

[8] L. Cristofor and D. Simovici. Generating an informative cover for association rules. In *Proceedings of International Conference on Data Mining*, Maebashi, Japan, 2002.

[9] CYGD. *http://mips.gsf.de/genre/proj/yeast/index.jsp*, accessed March 2004.

[10] L. Dehaspe and L. D. Raedt. Mining association rules in multiple relations. In *Proceedings of the 7th International Workshop on Inductive Logic Programming*, volume 1297, pages 125–132, Prague, Czech Republic, 1997.

[11] Elmasri and Navathe. *Fundamentals of Database Systems*. Pearson, Boston, 4th edition, 2004.

[12] L. Falquet, M. Pagni, P. Bucher, N. Hulo, C. J. Sigrist, K. Hofmann, and A. Bairoch. The prosite database, its status in 2002. *Nucleic Acids Research*, 30:235–238, 2002.

[13] J. Han and Y. Fu. Discovery of multiple-level association rules from large databases. In *Proceedings of the 21th International Conference on Very Large Data Bases*, San Francisco, CA, 1995.

[14] O. C. I. Ikura Lab. Domain fusion database. *http://calcium.uhnres.utoronto.ca/pi/pub_pages/download/index.htm*, accessed March 2004.

[15] A. Inokuchi, T. Washio, and H. Motoda. An apriori-based algorithm for mining frequent substructures from graph data. In *Proceedings of the 4th European Conference on Principles of Data Mining and Knowledge Discovery*, pages 13–23, Lyon, France, 2000.

[16] T. Ito, T. Chiba, R. Ozawa, M. Yoshida, M. Hattori, and Y. Sakaki. A comprehensive two-hybrid analysis to explore the yeast protein interactome. *Proc Natl Acad Sci U S A*, 98(8):4569–74, 2001.

[17] V. C. Jensen and N. Soparkar. Frequent itemset couting across multiple tables. In *Proceedings of PAKDD*, pages 49–61, 2000.

[18] A. J. Knobbe, H. Blockeel, A. Siebes, and D. M. G. van der Wallen. Multi-relational data mining. Technical Report INS-R9908, Maastricht University, 9, 1999.

[19] E. M. Marcotte, M. Pellegrini, H. L. Ng, D. W. Rice, T. O. Yeates, and D. Eisenberg. Detecting protein function and protein-protein interactions from genome sequences. *Science*, 285(5428):751–3, 1999.

[20] H. Mewes, D. Frishman, U. Gldener, G. Mannhaupt, K. Mayer, M. Mokrejs, B. Morgenstern, M. Mnsterkoetter, S. Rudd, and B. Weil. Mips: a database for genomes and protein sequences. *Nucleic Acids Research*, 30(1):31–44, 2002.

[21] K. Michihiro and G. Karypis. Frequent subgraph discovery. In *Proceedings of the International Conference on Data Mining*, pages 313–320, San Jose, California, 2001.

[22] T. Oyama, K. Kitano, K. Satou, and T. Ito. Extraction of knowledge on protein-protein interaction by association rule discovery. *Bioinformatics*, 18(8):705–14, 2002.

[23] O. Ozier, N. Amin, and T. Ideker. Global architecture of genetic interactions on the protein network. *Nat Biotechnol*, 21(5):490–1, 2003.

[24] J. Pei, J. Han, B. Mortazavi-Asl, H. Pinto, Q. Chen, U. Dayal, and M.-C. Hsu. PrefixSpan mining sequential patterns efficiently by prefix projected pattern growth. In *Proceedings of the 17th International Conference on Data Engineering*, pages 215–226, Heidelberg, Germany, 2001.

[25] B. Schwikowski, P. Uetz, and S. Fields. A network of protein-protein interactions in yeast. *Nature Biotechnol.*, 18(12):1242–3, 2000.

[26] A. H. Y. Tong, M. Evangelista, A. B. Parsons, H. Xu, G. D. Bader, N. Pag, M. Robinson, S. Raghibizadeh, C. W. V. Hogue, H. Bussey, B. Andrews, M. Tyers, and C. Boone. Systematic genetic analysis with ordered arrays of yeast deletion mutants. *Science*, 294(5550):2364–8, 2001.

[27] A. H. Y. Tong, M. Evangelista, A. B. Parsons, H. Xu, G. D. Bader, N. Pag, M. Robinson, S. Raghibizadeh, C. W. V. Hogue, H. Bussey, B. Andrews, M. Tyers, and C. Boone. Global mapping of the yeast genetic interaction network. *Science*, 303(5695):808–815, 2004.

[28] K. Truong and M. Ikura. Domain fusion analysis by applying relational algebra to protein sequence and domain databases. *BMC Bioinformatics*, 4:16, 2003.

[29] A. K. H. Tung, H. Lu, J. Han, and L. Feng. Breaking the barrier of transactions: Mining inter-transaction association rules. In *Proceedings of the International Conference on Knowledge Discovery and Data Mining*, San Diego, CA, 1999.

[30] P. Uetz, L. Giot, G. Cagney, T. A. Mansfield, R. S. Judson, J. R. Knight, D. Lockshon, V. Narayan, M. Srinivasan, P. Pochart, A. Qureshi-Emili, Y. Li, B. Godwin, D. Conover, T. Kalbfleisch, G. Vijayadamodar, M. Yang, M. Johnston, S. Fields, and J. M. Rothberg. A comprehensive analysis of protein-protein interactions in saccharomyces cerevisiae. *Nature*, 403(6770):623–7, 2000.

[31] X. Yan and J. Han. gspan: Graph-based substructure pattern mining. In *Proceedings of the International Conference on Data Mining*, Maebashi City, Japan, 2002.

[32] X. Yan, J. Han, and R. Afshar. Clospan: Mining closed sequential patterns in large datasets. In *Proceedings 2003 SIAM Int.Conf. on Data Mining*, San Francisco, California, 2003.

[33] M. J. Zaki. Generating non-redundant association rules. In *Knowledge Discovery and Data Mining*, pages 34–43, Boston, MA, 2000.

[34] M. J. Zaki. SPADE: An efficient algorithm for mining frequent sequences. *Machine Learning Journal*, 42:31–60, 2001.