10th International Workshop on Data Mining in Bioinformatics (BIOKDD 2011)

Held in conjunction with SIGKDD conference, San Diego, CA, USA, August 2011

Workshop Chairs

Mohammad Hasan
Jun Huan
Jake Y. Chen
Mohammed Zaki

BIOKDD'11 International Workshop on Data Mining in Bioinformatics San Diego CA, USA

Held in conjunction with
17th ACM SIGKDD Conference on Knowledge Discovery and Data Mining

General Chairs:

Mohammed Zaki Department of Computer Science Rensselaer Polytechnic Institute Troy, NY 12180-3590 zaki@cs.rpi.edu	Jake Chen Indiana University School of Informatics Indiana University-Purdue University Indianapolis Indianapolis, IN 46202-5132 jakechen@iupui.edu

Program Chairs:

Mohammad Al Hasan Department of Computer and Information Science Indiana University-Purdue University Indianapolis (IUPUI) Indianapolis, IN 46202-5132 alhasan@cs.iupui.edu	Jun (Luke) Huan Department of Electrical Engineering and Computer Science University of Kansas Lawrence, KS, 66047-7621 jhuan@ku.edu

REMARKS

Bioinformatics is the science of managing, mining, and interpreting information from biological data. Various genome projects have contributed to an exponential growth in DNA and protein sequence databases. Advances in high-throughput technology such as microarrays and mass spectrometry have further created the fields of functional genomics and proteomics, in which one can monitor quantitatively the presence of multiple genes, proteins, metabolites, and compounds in a given biological state. The ongoing influx of these data, the presence of biological answers to data observed despite noise, and the gap between data collection and knowledge curation have collectively created exciting opportunities for data mining researchers.

While tremendous progress has been made over the years, many of the fundamental problems in bioinformatics, such as protein structure prediction, gene-environment interaction, and regulatory pathway mapping, are still open. Beside these, new technologies such as next-generation sequencing are producing massive amount of sequence data; managing, mining and compressing these data raise challenging issues. Data mining will play an essential role in understanding these fundamental problems and development of novel therapeutic/diagnostic solutions in post-genome medicine.

The goal of this workshop is to encourage KDD researchers to take on the numerous challenges that Bioinformatics offers. This year, the workshop will feature the theme of "Data Mining Challenges in Next-generation Sequencing (NGS)". NGS is revolutionizing biological, biomedical, and health research. There are enormous data analyses and knowledge discovery challenges in the NGS technology, including expression analysis, mutational analysis, alternative slicing pattern discovery, whole transcription sequence alignment, epigenetics site discovery, storing and compression of high volume sequence data and clustering and classification of structural variations in a population.

We encourage papers that propose novel data mining techniques for areas including but not limited to

- NGS data processing
- Genome structural variation analysis
- Exome sequencing
- Comparative assessment of data qualities between NGS and microarray-based technology
- Comparative Genomics
- Metagenomics using NGS
- RNA-seq expression analysis
- Genome-wide analysis of non-coding RNAs
- Mutational analysis and disease risk assessment
- Genome-wide motif finding
- Modeling of biological networks and pathways from NGS data
- NGS and structural bioinformatics applications
- Correlating NGS with proteomics data analysis
- Biomarker discoveries in NGS data
- Gene functional annotation
- Special biological data management techniques for NGS data
- Special information visualization techniques for NGS data analysis
- Semantic webs and ontology-driven NGS data integration methods
- Knowledge discovery of genotype-phenotype associations in NGS
- Privacy and security issues in mining genomic databases

Papers should be at most 10 pages long, single-spaced, in font size 10 or larger with one-inch margins on all sides. Paper in PDF format can be sent to both of the program co-chairs by email. Camera-ready format papers may be referenced from previous BIOKDD conference proceedings.

PROGRAM

The workshop is a half day event in conjunction with the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Diego, CA USA, August 21, 2011. It was accepted into the full conference program after the SIGKDD conference organization committee reviewed the competitive proposal submitted by the workshop co-chairs. To promote this year's program, we established an Internet web site at http://bio.informatics.iupui.edu/biokdd11/.

This year, we accepted 3 full papers and 2 short papers out of 10 submissions. Each paper was peer reviewed by at least two members of the program committee and papers with declared conflict of interest were reviewed blindly to ensure impartiality. All papers, whether accepted or rejected, were given detailed review forms as a feedback. During the program, the full papers will have 25 minutes and the short papers will have 15 minutes of time. The above time is for both oral presentation and question & answers.

We have two invited speakers for this year's program, Dr. Vineet Bafna, Professor, University of California, San Diego and Dr. Harry Gao, Director, DNA Sequencing/Solexa Core Lab, City of Hope.

WORKSHOP CO-CHAIRS

- Mohammad Hasan, Computer Science, Indiana University–Purdue University Indianapolis
- Jun (Luke) Huan, University of Kansas
- Jake Chen, Indiana University School of Informatics, Indiana University–Purdue University Indianapolis
- Mohammed Zaki, Rensselaer Polytechnic Institute

PROGRAM COMMITTEE

- Vineet Chaoji (Yahoo Research, Bangalore)
- Bin Chen (Indiana University, Bloomington)
- Xiang Chen (Chinese Academy of Science)
- Md Tamjid Hoque (IUPUI)
- Jingshan Huang (University of South Alabama)
- Hasan Jamil (Wayne State University)
- Asif Javed (IBM TJ Watson Research Center)
- George Karypis (University of Minnesota, Twin Cities)
- Mei Liu (Vanderbilt University)
- Huzefa Rangwala (George Mason University)
- Chandan Reddy (Wayne State University)
- Isidore Rigoutsos (Thomas Jefferson University)
- Jianhua Ruan (University of Texas, San Antonio)
- Saeed Salem (North Dakota State University, Fargo)
- Min Song (New Jersey Institute of Technology)
- Vincent Tseng (National Taiwan University)
- Duygu Ucar (Stanford University)
- Vladimir Vacic (Columbia University)
- Jason Wang (New Jersey Institute of Technology)
- Wei Wang (University of North Carolina, Chapel Hill)
- Jinbo Xu (Toyota Technological Institute, Chicago)
- Jie Zheng (Nanyang Technological University, Singapore)

ACKNOWLEDGEMENT

We would like to thank all the program committee members, contributing authors, invited speaker, and attendees for contributing to the success of the workshop. Special thanks are also extended to the SIGKDD ’11 conference organizing committee for coordinating with us to put together the excellent workshop program on schedule.

PROGRAM SCHEDUKE

8:25-8:30: **Opening Remarks**

8:30-9:25: Invited Speaker presentation 1

Session I (9:30 am – 10:10 am)

9:30 – 9:55 [L] Zhen Hu and Raj Bhatnagar. *Algorithm for Low-Variance Biclusters to Identify Coregulation Modules in Sequencing Datasets*

9:55 – 10:10 [S] Mina Maleki, Md. Mominul Aziz and Luis Rueda. *Analysis of Obligate and Non-obligate Complexes using Desolvation Energies in Domain-domain Interactions*

10:10-10:30 **Coffee Break**

10:30-11:25: **Invited speaker presentation 2**

Session II: (11:30 am – 12:35 pm)

11:30 – 11:55 [L] K.S.M. Tozammel Hossain, Chris Bailey-Kellogg, Alan Friedman, Michael Bradley, Nathan Baker and Naren Ramakrishnan. *Using Physicochemical Properties of Amino Acids to induce Graphical Models of Residue Couplings*

11:55 – 12:20 [L] Hamching Lam and Daniel Boley. *Analyze Influenza Virus Sequences Using Binary Encoding Approach*

12:20 – 12:35 pm [S] Ankit Agrawal, Sanchit Misra, Ramanathan Narayanan, Lalith Polepeddi and Alok Choudhary. *A Lung Cancer Outcome Calculator Using Ensemble Data Mining on SEER Data*

12:35 – 12:45 **Closing Remarks**

Algorithm for Low-Variance Biclusters to Identify Coregulation Modules in Sequencing Datasets

Zhen Hu
School of Computing Sciences and Informatics
University of Cincinnati
huze@mail.uc.edu

Raj Bhatnagar
School of Computing Sciences and Informatics
University of Cincinnati
Raj.Bhatnagar@uc.edu

ABSTRACT

High-throughput sequencing (CHIP-Seq) data exhibit binding events with possible binding locations and their strengths, followed by interpretations of the locations of peaks. Recent methods tend to summarize all CHIP-Seq peaks detected within a limited up and down region of each gene into one real-valued score in order to quantify the probability of regulation in a region. Applying subspace clustering (or biclustering) techniques on these scores would discover important knowledge such as the potential co-regulation or co-factors mechanisms. The ideal biclusters generated should contain subsets of genes, and transcription factors (TF) such that the cell-values in biclusters are distributed around a mean value with low variance. Such biclusters would indicate TF sets regulating gene sets with the same probability values. However, most existing biclustering algorithms are neither able to enforce variance as a strict limitation on the values contained in a bicluster, nor use variance as the guiding metric while searching for the desirable biclusters. An algorithm that uses search spaces defined by lattices containing all overlapping biclusters and a bound on variance values as the guiding metric is presented in this paper. The algorithm is shown to be an efficient and effective method for discovering the possibly overlapping biclusters under pre-defined variance bounds. We present in this paper our algorithm, its results with synthetic and CHIP-Seq and motif datasets, and compare them with the results obtained by other algorithms to demonstrate the power and effectiveness of our algorithm.

1. BACKGROUND AND MOTIVATION

Mining biclusters (or co-clusters) from sequencing datasets is one of the important ways to discover potential biological mechanisms. Transcription Factors (TF) binding related sequencing datasets, including High-throughput Chromatin Immunoprecipitation Sequencing (CHIP-Seq) [22] and motif searching [12] data, record potential matchings on genome with many different metrics. For example, CHIP-Seq peaks records intensity and position. By balancing contributions from several metrics, many researchers summarize them into unified scores to quantify the binding strengths for gene-TF pairs. These scores are very sensitive and minor differences may reflect quite different binding scenarios. Biclusters consisting of subsets of genes and TFs having very similar cell values can help provide insights into coregulation. However, traditional methods [29, 15, 14, 11] cannot be adapted easily to analyze the sequencing datasets because most of them do not seek biclusters with specificable bounds on statistical quantities such as the standard deviation (of the cell values). We present in this paper an algorithm to solve this problem. The generated biclusters are the largest possible in size such that the cell values contained in them are distributed with variance bounded by specified low thresholds.

High-throughput Chromatin Immunoprecipitation Sequencing (CHIP-Seq) experiments generate precise short DNA sequences bound to Transcription Factors. After mapping these short sequences back to the whole-genome sequence and searching for enriched regions, CHIP-Seq datasets provide precise binding information in terms of binding locations and strengths (or peaks) [23, 26, 27, 2, 13]. Many current methods summarize all peaks within up and down regions of each gene into a unified score by combinding the information of distances from peaks to transcription start sites (TSS) and the information of binding strengths together. For example, Ouyang et. al [21] compute the score by summing up all weighted peaks' strengths, influenced by the distances to TSS. Another similar type of sequencing dataset is generated while searching for motifs matching across the whole genome. The motifs are defined as position weighted matrices (TRANSFC), and the final matching scores are computed by using the method given in [12]. Both of these two types of sequencing scores are very sensitive; slight differences in scores indicate quite different binding scenarios. For example, based on the Ouyang et. al's method, same intensity peaks (E2F1) bound at positions 500 and 800 away from TSS may lead to differences of less than 1 between the final scores.

For illustration, we consider a very small synthetic dataset shown in Figure 1(a) in which the values are quite similar to the CHIP-Seq scores and motif matching scores. Biclusters shown in Figure 1(b) are such that the values in the selected cells are all about the same (std. dev. < 0.5) and also satisfy the size constraint, which is: biclusters should contain at least two rows and two columns. For binary datasets the theory of Formal Concept Analysis [17] treats all maximum sized sub-matrices containing only 1's as concepts and ar-

BIOKDD 2011, August 2011, San Diego, CA, USA

	a	b	c	d
$g1$	2.8	3.0	3.2	4.5
$g2$	3.0	2.7	2.7	1.6
$g3$	4.9	5.0	5.3	1.2
$g4$	2.1	5.2	4.8	0.8

(a) Data table

$< \{g1, g2\}, \{a, b, c\} >$
$< \{g1, g2\}, \{a, b\} >$
$< \{g1, g2\}, \{a, c\} >$
$< \{g1, g2\}, \{b, c\} >$
$< \{g3, g4\}, \{b, c\} >$

(b) Biclusters

Figure 1: Example Biclusters

ranges them in a partially ordered lattice. Here we consider all those maximum sized sub-matrices as *concepts* for which the standard deviation of all included cell values is below some thresholds. The parent-child relationship in the lattice is still defined by the superset-subset relationship among the attributes included in the bicluster. In our extension of the analogy to FCA lattices, each node of the lattice may contain more than one bicluster. Biclusters shown in figure 1(b) meet all the above requirements and are qualified to be *concepts* in the sense outlined above.

Potential co-factor or co-regulation mechanism could be discovered from these sequencing datasets by taking subsets of genes and subsets of TFs such that all TFs have similar binding probability with select genes (or low-variance cell values of the sub-matrices). The problem of discovering the qualified biclusters, including the ones that may overlap some other biclusters, is NP-Hard [20] and most of the proposed algorithms attack the problem in a greedy manner [11, 8]. These algorithms, however, do not emphasis the cell-values' variance or STD. Some other algorithms utilize pattern recognition techniques [28] to improve the quality of clusters but they miss out on the many potential good overlapping biclusters due to imposing hard pattern restrictions on real valued data. There are also many biclustering algorithms which are based on statistical theory [15, 14, 25]. These algorithms use their own optimized metric for clustering and it is still not clear how to control the variance of the cell-values in biclusters.

One critical issue with real-valued datasets is that the standard deviation of cell-values in any selected sub matrix depends on the distribution of all of these values. This means incremental addition of rows and/or columns to construct a larger bicluster cannot be guided, in an algorithm, by a monotonically increasing/decreasing variance of all the included cell-values. The variance itself is not one such monotonic metric and therefore, one challenge addressed by us in this paper is to develop such a monotonic metric and correlate it with the variance and standard deviation of a bicluster.

A closed bicluster is one to which we cannot add either an attribute (column) or an object (row) and still maintain the standard deviation of all cells below the selected threshold. Our analogy with Formal concept analysis, and also our algorithms here, consider the lattice consisting of only the closed biclusters. A lattice of partially ordered closed biclusters is an efficient model of the search space in which a search algorithm may look for desirable closed biclusters. This approach has been adopted for finding biclusters in binary datasets [3, 7, 6, 30] and our work in this paper is the first attempt to advance the same idea to datasets with real-values entries in the cells.

In the following sections we formally define some ideas including a monotonic quality that can be used to bound the standard deviation of a non-closed bicluster. In section §3 we prove the relationship between our monotonic metric with standard deviation and present our algorithm; in section §3.5 we present results of our algorithm after some efficiency enhancing pruning is employed, and in section §4 we present results with a synthetic dataset and two genomic datasets.

2. PRELIMINARIES

We need a monotonic metric to help us guide the search for the best biclusters and we choose Range (max - min) of all the values in a biclusters to be this metric. In this paper we use dedicated symbols Δ (and δ) to denote the Range for a set of values in a submatrix.

Definition 1. The **Range** of a group of N data elements is the difference between the maximum and the minimum values of that group. That is,

$$\Delta = \max(N) - \min(N) \tag{1}$$

Given the range for a set of data elements we can derive an upper bound on the standard deviation for the data elements. This is possible because the standard deviation depends on the difference between an element and the mean and the value of *Range* is an upper bound on this difference value. Consequently, we can derive the relationship between standard deviation and range for single dimensional data in equation 2, which means by limiting Range the standard deviation is also limited.

$$s^2 \leq \delta^2 \tag{2}$$

From the point of view of formulating a search algorithm, we need a quantity that monotonically increases (or decreases) as the size of a potential biclusters increases. It is easy to see that as the size of a biclusters is enlarged, the *Range* of its values (and therefore the upper bound on its standard deviation) can only increase. This information, combined with the size of a potential biclusters, can be used to prune some potential search paths and also determine the most promising paths.

We represent a dataset as $D = (R, C)$, where R indicates the rows (or objects) of the table and C indicates the columns (or features). A bicluster is represented as $B = (sr, sc)$ where $sr \subseteq R$ and $sc \subseteq C$. We use $\widehat{B}$ to indicate the number of columns in a bicluster B, $\widetilde{B}$ to indicate its number of rows, and s_B to indicate its standard deviation.

There are many algorithms [11, 25, 5, 18, 19, 28, 14] using different metrics to define interesting biclusters. The quality of desired biclusters are based on those metrics. Here we give the definition of interesting biclusters which, compared with others, is based on the statistical restriction (standard deviation) directly:

Definition 2. A bicluster (B) is **an interesting bicluster** if it satisfies all of the following constraints: (i) $\widehat{B} \geq m$;

(ii) $\widetilde{B} \geq n$; and (iii) $s_B \leq S'$, where m and n are prespecified row and column sizes and S' is the threshold for the standard deviation.

In order to compare two biclusters, we also define several operators that will be used in pruning some branches of the search algorithm presented in section 3.3.

Definition 3. 1. A biclusters $B_1 = (sr_1, sc_1)$ is contained in biclusters $B_2 = (sr_2, sc_2)$, if and only if, $sr_1 \subseteq sr_2$ and $sc_1 \subseteq sc_2$.

2. A biclusters $B_1 = (sr_1, sc_1)$ is similar to $B_2 = (sr_2, sc_2)$, that is, $B_1 \approx B_2$ or $B_2 \approx B_1$, if and only if,

$$\frac{|sr_1 \cap sr_2|}{|sr_1 \cup sr_2|} \geq \theta; \;\; and \;\; \frac{|sc_1 \cap sc_2|}{|sc_1 \cup sc_2|} \geq \theta$$

where θ is a user defined threshold for similarity and has a value between 0 and 1.

For comparing two bicluster's interestingness based only on their sizes, we define an operator below which uses the number of cells included in each bicluster as the criterion. Intuitively, extending the **Range** bound for the biclusters to be found will lead our algorithm to generate biclusters with larger sizes. The trade-off between size and **Range** bound could be easily defined, if needed, and implemented in our algorithm.

Definition 4. A bicluster(B_1) is **more interesting than** a bicluster (B_2), that is, $B_1 \succ B_2$ or $B_2 \prec B_1$,if and only if

$$\widehat{B_1} \times \widetilde{B_1} \geq \widehat{B_2} \times \widetilde{B_2} \tag{3}$$

3. SEARCH ALGORITHM

The term *Range* is coined to restrict the statistical quality of biclusters and conduct our searching algorithm. We will prove its capability in restricting standard deviation and explain its usages in searching process. Relevant optimization strategies used in the searching algorithm are also discussed and analyzed.

3.1 Relating Range to Standard Deviation

Our criterion for choosing biclusters includes the standard deviation for all the values included in a bicluster. In order to construct relationship between range and standard deviation we define a few qualities computing standard deviations for individual rows and columns.

The *Range* for the i^{th} row of elements is denoted by $\delta_{i.}$, for the j^{th} column it is denoted by $\delta_{.j}$, and for whole bicluster it is denoted by Δ_B. The symbol $B_{i.}$ denotes all the data in the i^{th} rows of a bicluster B; $|B|_{i.}$ denotes the number of cells in the i^{th} row; $B_{.j}$ denotes the data in the j^{th} column of B; $|B|_{.j}$ denotes the number of data cells in j^{th} column; and μ and s denote the mean and standard deviation, defined as follows:

$$\begin{aligned} \mu_{i.} &= \frac{\sum_{d_{ip} \in B_{i.}} d_{ip}}{|B|_{i.}} \\ s^2_{i.} &= \frac{\sum_{d_{ip} \in B_{i.}} (d_{ip} - \mu_{i.})^2}{|B|_{i.}} \\ \mu_{.j} &= \frac{\sum_{d_{qj} \in B_{.j}} d_{qj}}{|B|_{.j}} \\ s^2_{.j} &= \frac{\sum_{d_{qj} \in B_{.j}} (d_{qj} - \mu_{.j})^2}{|B|_{.j}} \end{aligned} \tag{4}$$

Lemma 1. Given a bicluster $B = (sr, sc)$, if for $\{i \in sr|\ \delta_{i.} \leq S\}$ and $\{j \in sc|\ \delta_{.j} \leq S\}$, then $\Delta_B \leq 2 \times S$. (That is, if the *Range* for each row and each column of a bicluster is bounded by certain threshold S then the range for the whole bicluster is bounded by $2S$.)

PROOF. Let d_{ij} indicate the maximum value in the bicluster B, d_{pq} indicate the minimum value, $min(d_{i.})$ indicate the minimum value in the i^{th} row and $max(d_{.q})$ indicate the maximum value in the q^{th} column. From the definition of 1, we can derive the following inequalities:

$$\begin{aligned} d_{ij} - min(d_{i.}) &\geq d_{ij} - d_{iq} \\ max(d_{.q}) - d_{pq} &\geq d_{iq} - d_{pq}. \end{aligned} \tag{5}$$

The left hand side of each inequality is smaller than S and adding the two expressions on the left and right hand sides gives us:

$$d_{ij} - d_{pq} \leq 2 \times S. \tag{6}$$

□

Then a stronger conclusion about the relationship between the bound of the standard deviation and *Range* for the values in a bicluster can be derived.

Lemma 2. Given a bicluster $B = (sr, sc)$, if for $\{i \in sr|\ \delta_{i.} \leq S\}$ and $\{j \in sc|\ \delta_{.j} \leq S\}$, then the standard deviation s^2_B is less than $2 \times S^2$. (That is, if the range for each row and each column of a bicluster is less than S then the standard deviation of the bicluster is less than $2 \times S^2$.)

PROOF. When a bicluster $B = (sr, sc)$ has n rows and m columns, each of which has an upper bound of S on its Range. From equation 2, we can derive that:

$$\begin{aligned} s^2_{i.} &\leq \delta^2_{i.} \leq S^2 \\ s^2_{.j} &\leq \delta^2_{.j} \leq S^2 \end{aligned} \tag{7}$$

We use $\bar{\mu}$ to denote the mean of all individual row means, $\mu_{i.}$'s, s_μ to denote the standard deviation of all the individual row means, and μ is the mean of all the elements in the bicluster. Then we can say that:

$$\begin{aligned} s^2_B &= \frac{\sum_{i=1}^{n} \sum_{j=1}^{m} (d_{ij} - \mu)^2}{nm} \\ &= \frac{\sum_{i=1}^{n} \sum_{j=1}^{m} d^2_{ij} - nm\mu^2}{nm} \end{aligned} \tag{8}$$

$$\begin{aligned} \bar{\mu} &= \frac{\sum_{i=1}^{n} \mu_{i.}}{n} = \frac{\sum_{i=1}^{n} \sum_{j=1}^{m} d_{ij}}{nm} \\ &= \frac{\sum_{j=1}^{m} \mu_{.j}}{m} = \mu \end{aligned} \tag{9}$$

$$s^2_\mu = \frac{\sum_{i=1}^{n} (\mu_{i.} - \bar{\mu})^2}{n} = \frac{\sum_{i=1}^{n} \mu^2_{i.} - n\bar{\mu}^2}{n} \tag{10}$$

Combining equations 4 and 8 we get:

$$\begin{aligned} s^2_B &= \frac{\sum_{i=1}^{n} (s^2_i + \mu^2_i - \bar{\mu}^2)}{n} \\ &= \frac{\sum_{i=1}^{n} s^2_{i.}}{n} + s^2_\mu \\ &\leq \frac{\sum_{i=1}^{n} \delta^2_{i.}}{n} + s^2_\mu \end{aligned} \tag{11}$$

Also, for any row $p \in n$ we claim the following and then prove it by induction.

$$\left(\frac{\sum_{j=1}^{m}(d_{pj}-\mu_{.j})}{m}\right)^2 \leq \frac{\sum_{j=1}^{m}(d_{pj}-\mu_{.j})^2}{m} \tag{12}$$

The main steps of the induction proof, done on the number of columns, are as follows. Let $\phi_q = d_{pq} - \mu_{.q}$, and let k indicate the number of columns. When $k = 2$, equation 12 is satisfied. Now assuming the equation 12 to be correct for $k = \tau$, we get:

$$\left(\sum_{q=1}^{\tau}\phi_q\right)^2 \leq \tau * \sum_{q=1}^{\tau}\phi_q^2 \tag{13}$$

Then for $k = \tau + 1$

$$\begin{aligned}\left(\sum_{q=1}^{\tau+1}\phi_q\right)^2 &= \left(\sum_{q=1}^{\tau}\phi_q\right)^2 + 2*\left(\sum_{q=1}^{\tau}\phi_q\right)*\phi_{\tau+1} + \phi_{\tau+1}^2 \\ &\leq \left(\sum_{q=1}^{\tau}\phi_q\right)^2 + \sum_{q=1}^{\tau}\phi_q^2 + \tau*\phi_{\tau+1}^2 + \phi_{\tau+1}^2 \\ &\leq (\tau+1)*\sum_{q=1}^{\tau+1}\phi_q^2 \end{aligned} \tag{14}$$

The above is done for a single row, and summing for all rows we can derive that:

$$\begin{aligned} s_\mu^2 &= \frac{\sum_{i=1}^{n}(\mu_{i.}-\bar{\mu})^2}{n} \\ &= \frac{\sum_{i=1}^{n}(\sum_{j=1}^{m}(d_{ij}-\mu))^2}{m^2*n} \\ &\leq \frac{\sum_{i=1}^{n}\sum_{j=1}^{m}(d_{ij}-\mu)^2}{m*n} \\ &= \frac{\sum_{j=1}^{m}s_{.j}^2}{m} \leq \frac{\sum_{j=1}^{m}\delta_{.j}^2}{m} \end{aligned} \tag{15}$$

Combining conclusions from equations 7,11 and 15 we can derive that:

$$s_B^2 \leq 2 * S^2. \tag{16}$$

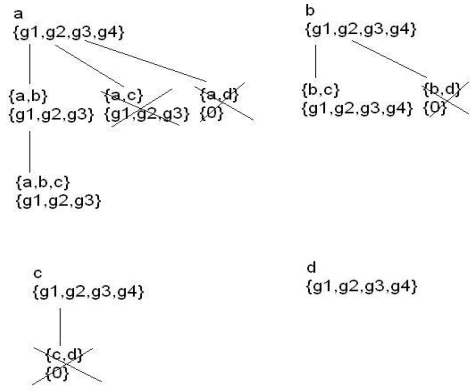
This means that by bounding the *Range* for each row and column ($\delta \leq S$), the standard deviation of the whole bicluster also gets bounded ($s_B \leq \sqrt{2}S$) which is also denoted as S' in definition 2. This conclusion is an important theoretical support for our search algorithm, which looks at biclusters as combinations of rows and columns and advances in the search space by adding columns or deleting rows. If the search algorithm wants to find biclusters with some bound on the standard deviation of its values, it could focus on a bound on the Range for each row and column separately. □

3.2 Enumerate Biclusters

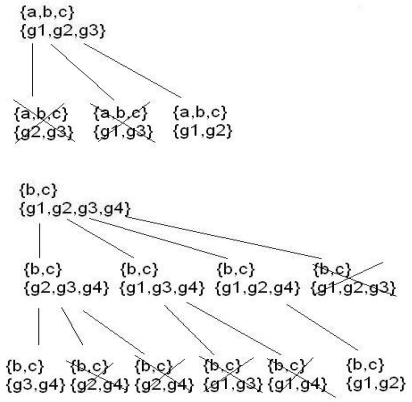
There are many ways incorporating *Range* in clustering procedure. What we are interested, also believed to be more practical to real problems, is to discover most interesting clusters base on definition 2 and 4. In order to discover biclusters with largest possible size, we need to limit the range of rows and columns separately which is also the reason we need supports from Lemma 2. We start our searching process by setting every single column with all rows as one searching branch of lattice. It is also applied for setting rows as one search branch. The basic operations for our search algorithm performed on intermediate biclusters are adding columns and removing rows. Biclusters which is covered by larger ones will not occurred in the final results. For example, in figure 1(b), $< \{g1, g2\}, \{a, b\} >$, $< \{g1, g2\}, \{a, c\} >$ and $< \{g1, g2\}, \{b, c\} >$ will not appear since they are covered by $< \{g1, g2\}, \{a, b, c\} >$.

Prefix-based equivalence classes have been used to formulate many search algorithms. We can form prefixes either from column headings or from row headings and in our case we have chosen to use the column headings. This helps divide the search space into independent sub-spaces of the search space at each level. This approach has been successfully adopted in [30] and [3] for searching biclusters in binary datasets.

Our search process can be viewed as made up of two independent phases. In the first phase, we generate children candidates by adding columns to each parent bicluster, updating the range for each row in the context of newly added columns, and removing those rows whose range values exceed the specified threshold. Such prefix tree based enumeration guarantees that every possible combinations of



(a) Phase1



(b) Phase2

Figure 2: Prefix Tree

columns will be examined. The first phase of the search algorithm for the example given earlier in figure 1 is shown in figure 2(a) here. In the second phase we re-examine all the generated candidate biclusters and check the Range values for each column. If a column's Range exceeds the specified threshold, the offending rows are removed from the candidate to make each column comply with the Range threshold.

At top level of each search branch, we enumerate every single column combined with all rows as one candidate bicluster. Since the order of the columns (a, b, c, d) are fixed, each candidate can only add columns that follow it. For example the only column that could be added to candidate $< \{g1, g2, g3\}, \{a, c\} >$ is d. After adding some columns and then removing rows, some candidate biclusters may violate the size constraint, such as $< \{\phi\}, \{a, d\} >$, $< \{\phi\}, \{b, d\} >$, and may be removed. The candidate $< \{g1, g2, g3\}, \{a, c\} >$ is removed because it is contained in an already generated bicluster.

Phase-one guarantees that all prefix combinations will be enumerated but the candidates may not comply with the constraint on the Range value for each column; In phase-two the algorithm removes some rows to bring each column within the acceptable Range limit. For example, in figure 2(b) $< \{g1, g2, g3, g4\}, \{b, c\} >$ generates $< \{g1, g2, g4\}, \{b, c\} >$ by removing $g3$, thus the only row could drop next is $g4$. After the removal the final results are $< \{g1, g2, g3\}, \{a, c\} >$, $< \{g1, g2\}, \{b, c\} >$ and $< \{g3, g4\}, \{b, c\} >$.

3.3 Pruning

To reduce the computational cost of the search, we employ a number of pruning strategies while still guaranteeing that all interesting biclusters will still be retained. These strategies are outlined below.

Pruning based on containment: In figure 2(a), our algorithms prune the candidate $< \{g1, g2, g3\}, \{a, c\} >$ since the depth first ordering of the search has already generated the hypothesis $< \{g1, g2, g3\}, \{a, b, c\} >$ which contains the former.

Pruning based on size: As stated in definition 4 we may compare the sizes of candidate biclusters and if only top k biclusters were needed from the entire search, we may without any loss, keep only top k candidates in each top level branch of the search and prune the rest.

Pruning based on similarity: In most real world datasets, a large number of the biclusters are *similar* to each other (definition 3). Our algorithm prunes the smaller sized biclusters among similar pairs of biclusters. All of the experiments reported in this paper have a threshold value of $\theta = 0.8$ as cutoff for pruning.

Pruning based on redundancy: Many real world datasets contain biclusters with very large number of rows. Instead of keeping all permutations of fewer rows as biclusters hypotheses, we delete from the parent bicluster those rows that reduce the Range the most and keep the rest of the rows in the hypotheses.

Even after previous prunings, the searching bicluster still could be more succinct. During the whole process, lattice could generate millions of biclusters. However what scientists really want to find is the biclusters which satisfy their own interestingness definitions. With modifications of definition 4, our algorithm could cut off the search branch which can not generate more interesting biclusters than those already had. Although this kind of pruning bring in some computations, compared to its reduction of searching space, it is really deserve. We also paralleled our algorithm by setting each searching branch as one thread. Hence the algorithm top K interesting biclusters for each thread and generate final biclusters by comparing those biclusters from each threads.

```
   input  : Data matrix DMX, Range δ, Share
            memory all current final biclusters Result,
            top interesting biclusters number K
   output: semi-qualified bicluster set BS
 1 begin
 2   Initialize BS each bs ∈ BS has one column with
     all rows ;
 3   while ∃bs ∈ BS can add more column do
 4     c = Next column id satisfying depth-first
       search prefix-tree;
 5     bs' = Add c to bs ;
 6     bs' remove rows which exceed δ;
 7     if bs' is interesting then
 8       Remove
         {bs''|bs'' ∈ BS ∧ bs'' ≈ bs' ∧ bs'' ≺ bs'} ;
 9       if no
         {bs''|bs'' ∈ BS ∧ bs'' ≈ bs' ∧ bs'' ≻ bs'}
         then
10         Add bs' to BS ;
11   if Result has K members then
12     for bs' ∈ Result do
13       Remove {bs''|bs'' ∈ BS ∧ bs'' ≺ bs'} ;
14   Return(BS)
15 end
```

Procedure 1. `Adding Columns`

3.4 Pseudo Code

The prefixes are constructed by each column and its combinations with those that follow it in the column ordering. Here we give the pseudo code of the algorithm to show how one of the prefix branches is pursued by the search algorithm (we call each branch a thread). The complete algorithm can be easily parallelized by having each thread run on a separate processor which also saves running times. We search biclusters from a real genetic dataset which contains 24190 rows and 5 columns on a computer equipped with a Intel Core 2 Quad 2.66GHz processor. By setting size limitation as 2000 rows, 2 column, range limitation as 2.1 and 4 threads working simultaneously, the searching process finishes in only 59 seconds.

3.5 Pruning Efficiency:

In order to analyze the effect of pruning we created a synthetic dataset with 25 rows and 25 columns, as shown in Figure 3(a); with the gray scale reflecting the cell values. There are four big blocks (biclusters) embedded in Figure 3(a), right-top, center, left-bottom and background, and the cell values within each block are distributed uniformly within a narrow range.

We count the number of intermediate candidate biclusters generated before the biclusters are output. The performances for various pruning strategies are shown in Figure 3(b). The x-axis shows the value of pre-specified Range

```
input  : Data matrix DMX, Range δ,
         semi-qualified bicluster set BS, top
         interesting biclusters number K,
output: Share memory all current final biclusters
         Result
1  begin
2    while ∃bs ∈ BS do
3      if System memory is not enough then
4        r = Next row id increasing interest the
         most;
5        bs' = Remove r from bs ;
6        Remove bs ;
7      else
8        r = Next row id satisfying depth-first
         search prefix-tree;
9        bs' = Remove r from bs ;
10     if bs' is interesting then
11       Remove
         {bs''|bs'' ∈ BS ∧ bs'' ≈ bs' ∧ bs'' ≺ bs'} ;
12       if no
         {bs''|bs'' ∈ BS ∧ bs'' ≈ bs' ∧ bs'' ≻ bs'}
         then
13         Add bs' to BS ;
14   Add BS to Result;
15   Keep top K interesting biclusters;
16   Return(Result)
17 end
```

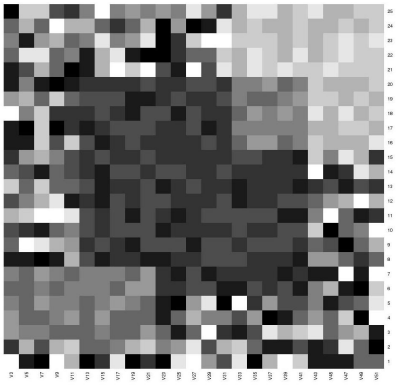
Procedure 2. `Removing Rows`

value and y-axis shows the number of intermediate biclusters. There are four cases plotted in Figure 3(b): the line with circles represents the performance of the original search algorithm based on pruning based on containment only; the line with triangles represents the performances of search using pruning based on size and containment; the line with crosses represents search with pruning based on similarity and containment; and the line with rectangles represents search with all the pruning strategies combined. Thus we can see that each pruning strategy has its impact on reducing the number of intermediate biclusters and using all the pruning techniques simultaneously performs the best.

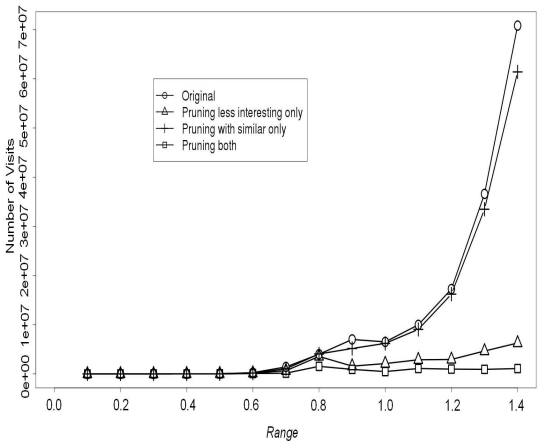
In real world dataset, the strategy of *pruning based on similarity* will greatly improve the performance. The reason why it dose not achieve much optimizations is that the the standard judging two biclusters are similar is very critical for this synthetic dataset. In our algorithm we set θ equals to 0.8 which means if two biclusters both have 9 columns and 9 rows, they are similar to each other only when 8 columns and 8 rows are the same.

4. EMPIRICAL EVALUATION

There are many biclustering algorithms can be compared with, we choose Cheng et al.'s algorithm [11] delegating direct biclustering algorithm and SAMBA[25] algorithm delegating graph theory based biclustering algorithm for synthetic dataset. We compare both accuracy and effectiveness of our algorithm with them. We also test our algorithm with two datasets from genomics domain to show the biological significance of output biclusters and compared with



(a) 25x25(uniform)



(b) Number Of Visits

Figure 3: Efficiency Test

two more recent algorithms: Co-clustering [14], OPSM [5] and ISA [18, 19].

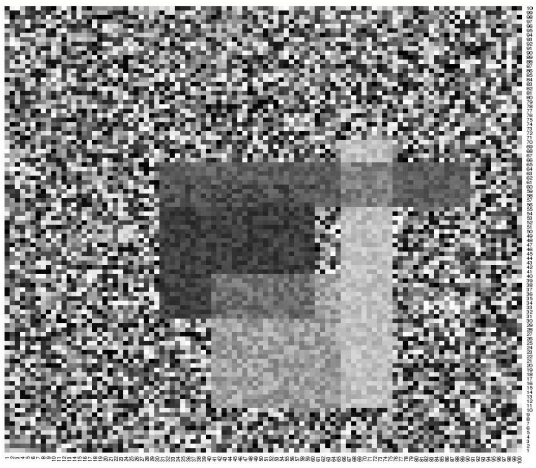
4.1 Synthetic Data Analysis

We consider the following metric for determining the quality of a bicluster found in a dataset. This evaluation metrics is not objective function for our algorithms and we do permit users to define their own interestingness setting by altering definition 4.

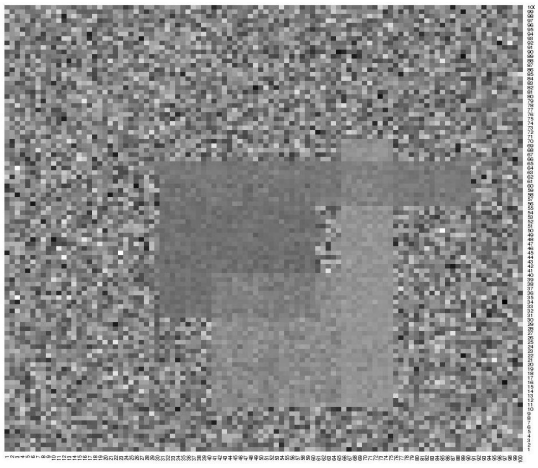
$$\lambda = \frac{\widehat{B_i} \times \widetilde{B_i} / s^2_{B_i}}{\widehat{D} \times \widetilde{D} / s^2_D}. \tag{17}$$

Here D represents the whole dataset, $\widehat{D}$ denotes its number of columns, $\widetilde{D}$ denotes its number of rows, and s_D denotes the standard deviation of D. This metric gives larger values for biclusters with larger sizes and smaller standard deviations. The metric is also normalized by s_D so that it is still meaningful across different datasets.

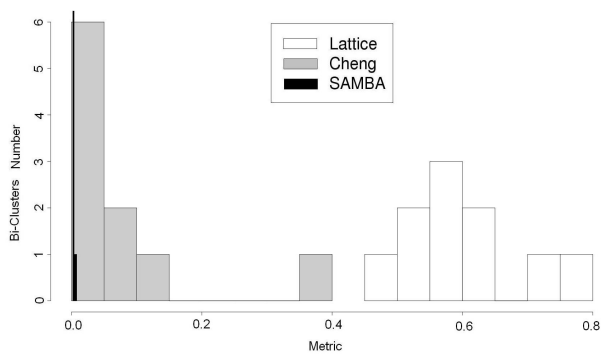
We have used two synthetic datasets, shown in Figure 4(a) and 4(b). The dataset in Figure 4(a) is 100 rows by 100 columns and values are reflected by the gray code intensity. There are five biclusters embedded in this dataset and all of them follow a uniform distribution of values. Four of the clusters are uniformly distributed around different cen-



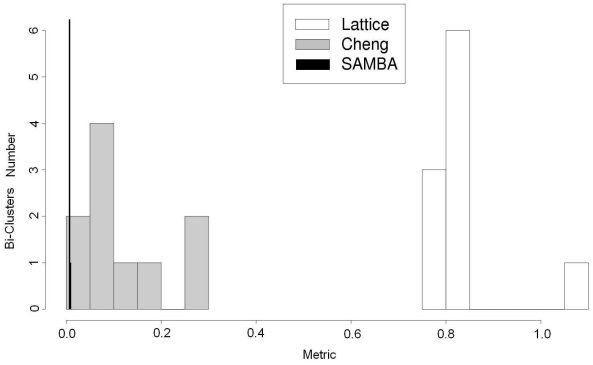
(a) 100x100(uniform)



(b) 100x100(normal)



(c) 100x100(uniform distribution)



(d) 100x100(normal distribution)

Figure 4: Synthetic Data

ters and values are within a range 1.2. The background cluster is distributed in the range of 2. Dataset in Figure 4(b) has the same size but the values of data cells in each biclusters follow normal distributions. Four overlapping biclusters are distributed normally with different μ and σ (less than 1.2). The background cluster is distributed normally with μ equals to zero and σ less than 2. We ran Cheng et al's algorithm by setting the size limit to 20 rows by 20 columns and our algorithm by setting the Range limit to 1.3. We also run SAMBA by setting option files type valsp_3p, with an overlap factor of 0.8, hashing kernel range from 1 to 7, and all other parameters as default value. We record the top 10 interesting biclusters for our clustering algorithm, first 10 biclusters generated by Cheng's algorithm and top 10 best biclusters based on metric value. The performance are presented in Figure 4(c). The x-axis in figure shows the metric value (λ) and y-axis shows the number of biclusters. There are three kind of bars in the figure: white bars represent the histogram of metric value for biclusters discovered by our algorithms; gray bars represent histogram of Cheng's algorithm and black bars represent histogram of SAMBA algorithm. biclusters discovered by our algorithm are shown to achieve the best quality as per the above metric.

Figure 4(d) shows the clustering comparisons for dataset in Figure 4(b). For Cheng et al's algorithm and SAMBA, parameter are set the same as in dataset in Figure 4(c). For our algorithm we extend range limit to 2.5 which covers more than two times the standard deviation for the normal distribution of biclusters ($\pm 2\sigma$). We see again from the second histogram that our algorithm performs significantly better that the other two. SAMBA achieve worse performance in both cases since it can not find biclusters with large size.

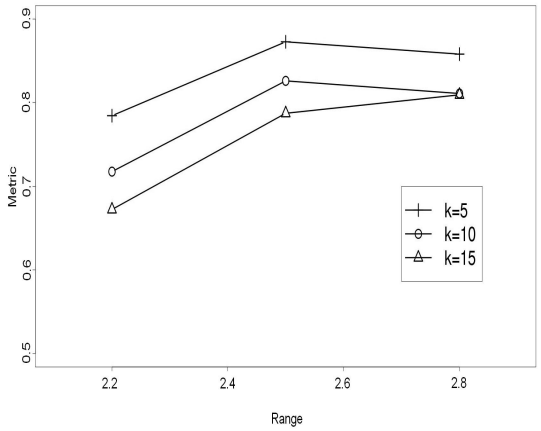


Figure 5: Effect of Parameter Changes

The impact of parameters on the performance of our algorithm can be analyzed by examining the evaluation metric

Algorithm	CLEAN Score	Variance Of Biclusters	Size (Rows X Cols)	Average Column STD
Lattice(our algorithm)	80.95	0.21	97x2	0.43
CC	33.65	1.27	847x12	0.89
OPSM	50.15	0.49	1726x6	0.53
Co-Clustering	40.44	0.99	469x2	0.99

Figure 6: Mouse Embryonic Stem Cell

Equation 17 for different values of the range and the value of k used for selecting the top k candidates. Apparently, our algorithm should generate more biclusters if we extend the range limit and/or increase the number k. Intuitively, extending range limit will increase allowable standard deviation but may also increase the size of the bicluster, and thus the metric may or may not be affected much. Meanwhile keeping more of the less interesting biclusters will reduce average of the metric value for the resulting biclusters. We use the dataset in Figure 4(b) to analyze the performance trend by modifying the parameters. For each parameter combination we record the average metric value for the obtained biclusters. Figure 5 shows the relationship between the Range parameter and the average of the metric value obtained. There are three lines in the figure: the top line with crosses shows the performance when we keep the top 5 most interesting biclusters based on size criterion; the line with circles shows the performance while keeping the top 10; and line with triangles shows the performance while keeping the top 15. When we extend the range from 2.2 to 2.5, the metric value increase due to a faster increase in the sizes of the resulting biclusters than the increases in their variances. However, for range values larger than 2.5 the increase in sizes is slower than the increase in the variances; see Figure 5; (This dataset consists of values that follow a normal distribution). Consequently, we can conclude that extending the **Range** does not always improve the performance because after certain point the increase of bicluster's size can not compensate for the decrease of accuracy (or increase in STD). Also, we see that keeping a smaller number of most interesting biclusters will always increase the performance for a specific value of Range because the standard deviation of those biclusters is relatively stable.

4.2 Mouse Embryonic Stem Cell Dataset

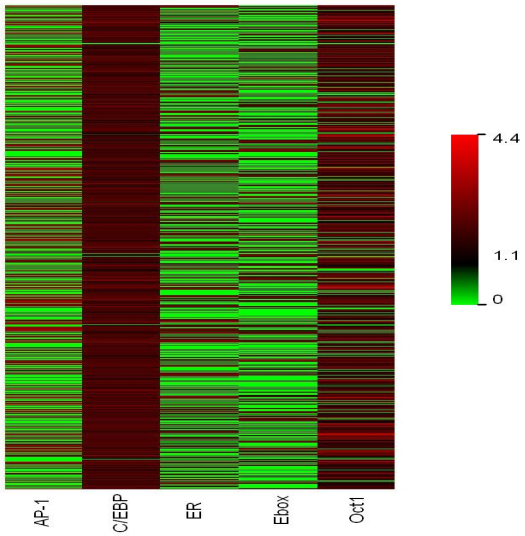
The authors Ouyang et al. [21] have reported their CHIP-Sequence scores on mouse embryonic stem cell in [10]. In this dataset the rows represent genes, the columns represent transcription factors (proteins), and the cell-values represent the strength of binding between the row and column elements. Twelve proteins and 18936 genes included in this dataset have been known to show correlations in some other studies. Using our algorithm on their original data without any normalization, we seek to discover underlying co-factor mechanisms. One bicluster unveiled co-regulated TFs (Nanog and Oct4) with a variance of 0.21 and that is well corroborated by [10]. In order to demonstrate the functional coherence of the genes co-regulated in the bicluster, we use the CLEAN [16] metric to check the functional enrichment with Gene Ontology terms [4]. The higher the CLEAN score the better is the functional coherence of the genes. Low-Variance biclusters found by our algorithm show the highest CLEAN-Score values and the lowest variance when compared with the biclusters found by other methods (first row in table of Figure 6 shows our results). It should be noted that traditional biclustering algorithms could discover biclusters with relatively low standard deviations within each column of a bicluster but the variance of the whole data blocks is larger, and therefore they could not find highly functionally correlated gene sets; and therefore their CLEAN scores are lower. For each algorithm, the data shown in the table represents the bicluster with highest CLEAN score (if many biclusters were found then the best one was selected and reported); the table also lists the bicluster variance and the average column standard deviation for each reported bicluster (Figure 6).

4.3 Human Genomic Dataset

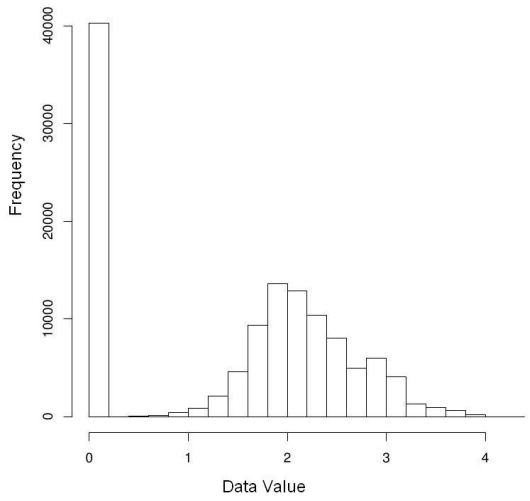
We consider the dataset from human genome from hg18 [1] and calculate the maximum possible relative probability associated with each gene-motif pair using the Sequence Motif-Matching Scoring model [12]. The data contains 24190 genes (rows) and 287 motifs (columns). Relative probability values in data cells are in the range of $[0, 4.3]$. The data is taken from [24]. The other source of data that we use is based on experiments [9]. They present the distributions of five motifs (ERE, AP-1, Oct, FKH and C/EBP) for these genes and also the pair-wise relationships between those motifs. The heat map of these five motifs with 24190 genes is shown in Figure 7(a) and the distribution of values is shown in Figure 7(b) in the format of histogram.

We want to see whether biclusters found by our algorithm in the theoretically obtained data match the ones reported in the experimental results. We use the *Fisher's Exact Test* to determine whether our clustering algorithm could really generate the biclusters predicted by the experimental results. The criteria used here is the negative of log 10 based p-value, meaning the higher the value the more significantly the two sets match. Results of our algorithm for various parameter values are shown in Figure 7(d). As we expected, keeping Range the same and increasing the minimum row-limit size reduces the number of clusters discovered (the first row and the second row), and increasing the Range bound discovers more biclusters but makes the accuracy worse (the second, third, and fourth rows). The best p-value of this test is 2.90×10^{-7} (or 6.54 for $-\log_{10}(pvalue)$) in the third row which is much smaller than conventional cutoff 0.05 (or 1.3 for $-\log_{10}(pvalue)$). Therefore we conclude that the biclusters discovered by our algorithm significantly overlap with the experimental results.

We also compare the results with some well known algorithms [11, 5, 18, 19, 14] using the metric defined in equation 17. Parameters used in all of the algorithms are kept as the default ones. For Cheng et al.'s algorithm, we re-



(a) Heatmap of Motif Data



(b) Motif Data Distribution

Algorithm	λ	$\widehat{\lambda}$	Average Column STD
Lattice(our algorithm)	0.91;0.80;0.81	0.84	0.31;0.25;0.32
CC	0.45	0.45	0.94
OPSM	0.19;0.22;0.23;0.23	0.22	0.31;0.57;0.77;0.87
Co-Clustering	0.43;0.29;0.19	0.30	0.27;0.63;0.34

(c) Motif Bicluster Comparison

δ	Row Limit	biclusters	$-\log_{10}(pvalue)$
2.1	6000	2	3.56
2.1	4000	3	6.54
2.3	4000	7	5.17
2.5	4000	8	5.35

(d) Motif Bicluster Statistics

Figure 7: Genomic Data Validation

tained the first bicluster that is generated; for biclustering algorithm ISA [18, 19], we could not find any biclusters; for biclustering algorithm OPSM [5], we kept all the biclusters generated; for our algorithm, we kept the top 3 biclusters with minimum size of 4000 rows and 2 columns in which data values have a range of 2.1 and for Co-clustering [14] algorithm we keep the same number of biclusters as our algorithm. The biclustering results are summarized using the metric (λ from equation 17) in Figure 7(c). We first listed all metric values for each bicluster generated in the second column, then we took the average of those metric values for each algorithm and it is reported in the third column of the table. ISA did not find any biclusters, so we did not put in the table. It is clear from the results that our algorithm could discover biclusters with largest metric values, either considering individual biclusters or their average.

5. CONCLUSION

We have presented a search based algorithm for discovering low-variance biclusters in sequencing datasets and have shown that it performs much better than several other competing algorithms using a statistical metric for merit. Our algorithm can enumerate overlapping biclusters and generate the top K interesting biclusters based on the specified size and standard deviation requirements. Other algorithms are not capable of discovering all overlapping biclusters and controlling the variance at the same time. Challenges still exist for discovering the complete set of low-variance biclusters because our algorithm presented here generates only those biclusters that satisfy the low-variance criterion but it cannot discover all low-variance biclusters - particularly those that have low variance despite a large range for the included values. But the combination of large range and low variance is not desirable for the sequencing data applications and our algorithm is therefore very suitable.

6. REFERENCES

[1] Ucsc genome browser website:. *http://genome.ucsc.edu/*.

[2] V. A, J. DS, S. A, M. C, A. E, and et al. Genome-wide analysis of transcription factor binding sites based on chip-seq data. *Nat Methods*, 5:829–834, 2008.

[3] F. Alqadah and R. Bhatnagar. An effective algorithm for mining 3-clusters in vertically partitioned data. *In Proceeding of the 17th ACM conference on*

Information and knowledge management, pages 1103–1112, 2008.

[4] M. Ashburner, C. Ball, J. Blake, D. Botstein, H. B. J. Cherry, A. Davis, K. Dolinski, S. Dwight, J. Eppig, and et al. Gene ontology: tool for the unification of biology. *Nature Genetics*, 25(1), 2000.

[5] B. C. Ben-Dor, R. Karp, and Z. Yakhini. Discovering local structure in gene expression data: The order-preserving submatrix problem. *In Proceedings of the 6th International Conference on Computational Biology (RECOMB-02)*, pages 49–57, 2002.

[6] H. Bian and R. Bhatnagar. An algorithm for lattice-structured subspace clustering. *Proceedings of the SIAM International Conference on Data Mining*, 2005.

[7] H. Bian, R. Bhatnagar, and B. Young. An efficient constraint-based closed set mining algorithm. *In Proceedings of the 6th international confernce on Machine Learning*, pages 172–177, 2007.

[8] K. Bryan, P. Cunningham, and BolshakovaN. Biclustering of expression data using simulated annealing. *In Proceedings of the 18th IEEE symposium on computer-based medical systems*, pages 383–388, 2005.

[9] J. S. Carroll, C. A. Meyer, J. Song, W. Li, T. R. Geistlinger, J. Eeckhoute, A. S. Brodsky, E. K. Keeton, K. C. Fertuck, G. F. Hall, Q. Wang, S. Bekiranov, V. Sementchenko, E. A. Fox, P. A. Silver, T. R. Gingeras, X. S. Liu, and M. Brown. Genome-wide analysis of estrogen receptor binding sites. *Nature Genetics*, 38:1289–1297, 2006.

[10] X. Chen, H. Xu, P. Yuan, F. Fang, M. Huss, V. B. Vega, E. Wong, Y. L. Orlov, W. Zhang, J. Jiang, Y.-H. Loh, H. C. Yeo, Z. X. Yeo, V. Narang, K. Ramamoorthy, Govindarajan, B. Leong, A. Shahab, Y. Ruan, G. Bourque, W.-K. Sung, N. D. Clarke, C.-L. Wei, and H.-H. Ng. Integration of external signaling pathways with the core transcriptional network in embryonic stem cells. *Cell*, 133:1106âĂŞ1117, 2008.

[11] Y. Cheng and G. Church. Biclustering of expression data. *In Proceedings of the 8th international conference on intelligent systems for molecular biology*, pages 93–103, 2000.

[12] E. Conlon, X. Liu, J. Lieb, and J. Liu. Integrating regulatory motif discovery and genome-wide expression analysis. *Proc. Natl. Acad. Sci. U.S.A.*, 100(6):3339–3344, 2003.

[13] N. D, C. S, and B. K. Empirical methods for controlling false positives and estimating confidence in chip-seq peaks. *BMC Bioinformatics*, 9:523, 2008.

[14] I. Dhillon. Co-clustering documents and words using bipartite spectral graph partitioning. *In Proceedings of the 7th ACM SIGKDD International Conference On Knowledge Discovery And Data Mining(KDD)*, 2001.

[15] I. Dhillon, S. Mallela, and D. S. Modha. Information-theoretic co-clustering. *In Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 89–98, 2003.

[16] J. M. Freudenberg, V. K. Joshi, Z. Hu, and M. Medvedovic. Clean: Clustering enrichment analysis. *BMC Bioinformatics*, 10(234), 2009.

[17] B. Ganter and R. Wille. Formal concept analysis: Mathematical foundations. *Springer-Verlag,Heidelber*, 1999.

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

[19] 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, 2002.

[20] S. C. Madeira and A. L. Oliveira. Biclustering algorithms for biological data analysis: A survey. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 1(1):24–45, 2004.

[21] Z. Ouyang, Q. Zhou, and W. H. Wong. Chip-seq of transcription factors predicts absolute and differential gene expression in embryonic stem cells. *PNAS*, 106(51):21521–21526, 2009.

[22] P. PJ. Chip-seq: advantages and challenges of a maturing technology. *Nat Rev Genet*, 10:669âĂŞ680, 2009.

[23] P. S, W. B, and M. A. Computation for chip-seq and rna-seq studies. *Nat Methods*, 6:S22âĂŞ32, 2009.

[24] K. Shinde, M. Phatak, J. M. Freudenberg, J. Chen, Q. Li, V. Joshi, Z. Hu, K. Ghosh, J. Meller, and M. Medvedovic. Genomics portals: Integrative web-platform for mining genomics data. *BMC Genomics*, 11(1), 2010.

[25] A. Tanay, R. Sharan, and R. Shamir. Discovering statistically significant bilcusters in gene expression data. *Bioinformatics*, 18:136–144, 2002.

[26] L. TD, R. S, T. S, L. R, A. T, and et al. A practical comparison of methods for detecting transcription factor binding sites in chip-seq experiments. *BMC Genomics*, 10:618, 2009.

[27] Z. Y, L. T, M. C, E. J, J. D, and et al. Model-based analysis of chip-seq (macs). *Genome Biology*, 9:R137, 2008.

[28] J. Yang, W. Wang, H. Wang, and P. Yu. δ-clusters: capturing subspace correlation in a large data set. *In Proceedings of the 18th IEEE International Conference On Data Engineering*, pages 517–528, 2002.

[29] S. Yoon, L. Benini, and D. M. G. Co-clustering: A versatile tool for data analysis in biomedical informatics. *Information Technology in Biomedicine,IEEE Transactions on*, 11:493–494.

[30] M. J. Zaki and K. Gouda. Fast vertical mining using diffsets. *In 9th International Conference on Knowledge Discovery and Data Mining*, 2003.

Analysis of Obligate and Non-obligate Complexes using Desolvation Energies in Domain-domain Interactions

Mina Maleki , Md. Mominul Aziz , and Luis Rueda
School of Computer Science
University of Windsor
401 Sunset Avenue, Windsor, Ontario N9B 3P4, Canada
{maleki,azizc,lrueda}@uwindsor.ca

ABSTRACT

Protein-protein interactions (PPI) are important in most biological processes and their study is crucial in many applications. Identification of types of protein complexes is a particular problem that has drawn the attention of the research community in the past few years. We focus on obligate and non-obligate complexes, their prediction and analysis. We propose a prediction model to distinguish between these two types of complexes, which uses desolvation energies of domain-domain interactions (DDI), pairs of atoms and amino acids present in the interfaces of such complexes. Principal components of the data were found and then the prediction is performed via linear dimensionality reduction (LDR) and support vector machines (SVM). Our results on a newly compiled dataset, namely binary-PPID, which is a joint and modified version of two well-known datasets consisting of 146 obligate and 169 non-obligate complexes, show that the best prediction is achieved with SVM (77.78%) when using desolvation energies of atom type features. Furthermore, a detailed analysis shows that different DDIs are present in obligate and non-obligate complexes, and that homo-DDIs are more likely to be present in obligate interactions.

Categories and Subject Descriptors

I.5.2 [**Pattern Recognition**]: Design Methodology—*Classifier design and evaluation, Feature evaluation and selection*

General Terms

Algorithms, Performance, Experimentation

Keywords

protein-protein interaction; domain-domain interaction; complex type prediction

1. INTRODUCTION

Protein interactions are important in many essential biological processes in living cells, including signal transduction, transport, cellular motion and gene regulation. As a consequence of this, the identification of protein-protein interactions (PPIs) is a key topic in life science research. Prediction of PPIs has been studied mostly using computational approaches and from many different perspectives. Prediction of interfaces (interactions between subunits) in different molecules includes analysis of patches, sites, amino acids, or even specific atoms. The physicochemical and geometric arrangement of subunits in protein complexes is best known as docking. An important aspect that has recently drawn the attention of the research community is to predict "when" the interactions will occur – this is mostly studied at the protein interaction network level. Another important aspect in studying PPIs is the identification of different types of complexes, including similarities between subunits (homo/hetero-oligomers), number of subunits involved in the interaction (dimers, trimers, etc.), duration of the interaction (transient vs. permanent), stability of the interaction (non-obligate vs. obligate), among others; we focus on the latter problem.

Obligate interactions are usually considered as permanent, while non-obligate interaction can be either permanent or transient [1]. Non-obligate and transient interactions are more difficult to study and understand due to their instability and short life, while obligate and permanent interactions last for a longer period of time, and hence are more stable [2]. For these reasons, an important problem is to distinguish between obligate and non-obligate complexes. To study the behavior of obligate and non-obligate interactions, in [3], it was shown that non-obligate complexes are rich in aromatic residues and arginine, while depleted in other charged residues. The study of [4] suggested that mobility differences of amino acids are more significant for obligate and large interface complexes than for transient and medium-sized ones.

Some studies in PPI consider the analysis of a wide range of parameters, including desolvation energies, amino acid composition, conservation, electrostatic energies, and hydrophobicity for predicting obligate and non-obligate complexes. In [1], a classification of obligate and non-obligate interactions was proposed where interactions are classified based on the lifetime of the complex. In [5], three different types of interactions were studied, namely crystal packing, obligate and non-obligate interactions. That study was based on using solvent accessible surface area, conservation scores, and the shapes of the interfaces. After classifying obligate and transient protein interactions based on 300 different interface attributes in [6], the difference in molecular weight between interacting chains was reported as the best single feature to distinguish transient from obligate interactions. Based on their results, interactions with the same molecular weight or large interfaces are obligate.

Different studies have claimed that only a few highly conserved residues are crucial for protein interactions [7, 8]. Moreover, it has been shown that physical interactions between proteins are mostly

BIOKDD 2011, August 2011, San Diego, CA, USA

controlled by their domains, as a domain is often the minimal and fundamental module corresponding to a biochemical function [7, 8]. Thus, in previous studies, the physical interaction between proteins is analyzed in terms of the interaction between residues of their structural domains. For example, in [7], interactions between residues were used for finding obligate and non-obligate residue contacts of PPIs. That study concluded that non-obligate interfaces occupy less than 2% of the area of the domain surfaces, while the number of obligatory interfaces is between 0–6%. In [8], the interface of 750 transient DDIs, interactions between domains that are part of different proteins, and 2,000 obligate interactions were studied. The interactions between domains of one amino acid chain were analyzed to obtain a better understanding of molecular recognition and identify frequent amino acids in the interfaces and on the surfaces of PPIs. Also, in [9], the domain information from protein complexes was used to predict four different types of PPIs including transient enzyme inhibitor/non enzyme inhibitor and permanent homo/hetero obligate complexes.

In a recent work [10], an approach to distinguish between obligate and non-obligate complexes has been proposed in which desolvation energies of amino acids and atoms present in the interfaces of PPIs are considered as the input features of the classifiers. The results of that classifier show that desolvation energies are better discriminant than solvent accessibility and conservation properties. In this paper, we present an analysis of PPIs that uses properties of DDIs present in the interface to predict obligate and non-obligate protein-protein interactions. Desolvation energies of atom and amino acid pairs present in the interface of DDIs as well as desolvation energies of all atom and amino acid pairs present in the interface of interacting complexes are used in the prediction. We have also performed an analysis on the DDIs present in the two types of interactions. A visual analysis shows that that unique pairs can be identified for both types of interactions, and highlight the presence of homo-DDIs in obligate interactions. The prediction approach resorts on two state-of-the-art classification techniques of linear dimensionality reduction (LDR) and support vector machines (SVM). Ten-fold cross validation of the proposed scheme on our binary-PPID dataset, which is an extended dataset that we compiled from two well-known datasets of [5] and [11], demonstrates that (a) using desolvation energies of atom type features are better than the features used in [5] for predicting obligate and non-obligate complexes, achieving 77.78% classification accuracy in comparison to 71.80% (b) atom type features are better than amino acid type features for prediction of these two types of complexes (c) although the prediction accuracies by considering atom and amino acid pairs present in the interacting domains instead of all interacting atom and amino acid pairs of two chains are low, they are still acceptable and provide additional information about the specific domains.

2. MATERIALS AND METHODS

2.1 Dataset

We have compiled a new dataset by merging two existing, preclassified datasets of protein complexes obtained from the studies of Zhu et al. [5], and Mintseris and Weng [11]. The former dataset contains 75 obligate and 62 non-obligate interactions while the latter contains 115 obligate and 212 transient interactions. There are 39 common interactions between these two datasets and hence the redundant complexes were removed. In addition, we carefully examined all the interactions and removed complexes with contradicting class labels. For example "*1eg9,A:B*" is classified as both obligate and non-obligate in [5] and [11]. In total, seven complexes: *1eg9, 1hsa, 1i1a, 1raf, 1d09, 1jkj* and *1cqi*, showed this contradiction and were then removed from the new dataset. After this pre-processing stage, the new dataset resulted in 417 complexes from which 182 were obligate and 235 were non-obligate. In this study, each complex is considered as the interaction of two chains (two single sub-units). Since the dataset of [11] considers the interaction of two units in which each may contain more than one chain, e.g., "*1qfu,AB:HL*", all these complexes were converted to interactions between two single chains (binary interactions). For this, all binary interactions of each of the 93 multiple-chain complexes were identified, obtaining 289 interactions, and each of these was converted into a separate complex in the new dataset. For example, the multiple-chain of *1qfu* was transformed to four binary chains as follows: *A:H*, *A:L*, *B:H* and *B:L*. Another step involves taking the whole dataset of binary complexes and filtering non-interacting pairs. Using the interface definition of [12], complexes with interacting chains with less than five interface residues were removed. Two residues (from different chains) are considered to be interacting if at least one pair of atoms from these residues is 5Å or less apart from each other. This resulted in a dataset that contains 516 complexes, from which 303 are non-obligate and 213 are obligate binary interactions. In a final step, we collected the domains contained in each interacting chain from the Pfam database [13]. The complexes that do not have any domain in at least one of their subunits were discarded in the analysis. This resulted in our final dataset of 315 complexes, from which 146 are obligate complexes and 169 are non-obligate complexes - we call this dataset binary protein-protein interactions by considering domain definitions (binary-PPID). The PDB IDs of these complexes and the interacting chains are shown in Table 1.

2.2 Features

We use desolvation energies as the predicting properties, which are shown to be very efficient for prediction of obligate and non-obligate complexes [10]. Knowledge-based contact potential that accounts for hydrophobic interactions, self-energy change upon desolvation of charged and polar atom groups, and side-chain entropy loss compose the so-called binding-free energy. In [14], the total desolvation energy is defined as follows:

$$\Delta G_{des} = g(r)\Sigma\Sigma e_{ij}. \quad (1)$$

If we are considering the interaction between the i^{th} atom of a ligand and the j^{th} atom of a receptor then e_{ij} is the atomic contact potential (ACP) [15] between them, and $g(r)$ is a smooth function based on their distance. The value of $g(r)$ is 1 for atoms that are less than 5 Å apart [14]. For simplicity, we consider the smooth function to be linear. Within the range of 5 and 7 Å, the value of $g(r)$ is $(7 - r)/2$.

We collected the structural data from the Protein Data Bank (PDB) [16] for each complex in our dataset. After adding domain information obtained from Pfam to each atom present in the chain, each PDB file was divided into two different ligand and receptor files based on its side chains. From [15], we know that there are 18 atom types. Thus, for each protein complex a feature vector with 18^2 values was obtained, where each feature contains the desolvation energy of a pair of atom types. As the order of interacting atom pairs is not important, the final length of feature vector for each complex was 171 that correspond to unique pairs. We also considered pairs of amino acids, and for this, we computed desolvation energy values for each pair of atoms using Eq. (1) and accumulated the values for each pair of amino acids. Avoiding repeated pairs resulted in 210 different features (unique pair of amino acids).

Table 1: binary-PPID dataset (146 obligate and 169 non-obligate binary complexes).

Obligate Complexes							
1a0f , A:B	1byk , A:B	1eex , A:B	1hcn , A:B	1jk0 , A:B	1li1 , A:C	1qbi , A:B	2hdh , A:B
1a6d , A:B	1c3o , A:B	1eex , A:G	1hfe , L:S	1jk8 , A:B	1li1 , B:C	1qdl , A:B	2hhm , A:B
1ahj , A:B	1c7n , A:B	1efv , A:B	1hgx , A:B	1jkm , A:B	1lti , C:G	1qfe , A:B	2kau , A:C
1aj8 , A:B	1ccw , A:B	1ep3 , A:B	1hjr , A:C	1jmx , A:G	1lti , C:H	1qfh , A:B	2kau , B:C
1ajs , A:B	1cmb , A:B	1ezv , D:H	1hr6 , A:B	1jnr , A:B	1lti , C:D	1qu7 , A:B	2min , A:B
1aq6 , A:B	1cnz , A:B	1ezv , C:F	1hss , A:B	1jro , A:B	1lti , C:F	1sgf , A:B	2mta , A:H
1b34 , A:B	1coz , A:B	1f6y , A:B	1ihf , A:B	1jwh , A:C	1lti , C:E	1sgf , A:Y	2nac , A:B
1b3a , A:B	1cpc , A:B	1ffu , A:C	1jb0 , B:E	1jwh , A:D	1luc , A:B	1spp , A:B	2pfl , A:B
1b4u , A:B	1dce , A:B	1ffv , A:B	1jb0 , B:E	1k3u , A:B	1mro , A:B	1spu , A:B	2utg , A:B
1b5e , A:B	1dii , A:C	1fm0 , D:E	1jb0 , B:D	1k8k , A:B	1mro , B:C	1trk , A:B	3gtu , A:B
1b7b , A:C	1dj7 , A:B	1g8k , A:B	1jb0 , B:D	1k8k , B:F	1mro , A:C	1vcb , A:B	3pce , A:M
1b7y , A:B	1dkf , A:B	1gka , A:B	1jb0 , A:E	1k8k , A:E	1msp , A:B	1vlt , A:B	3tmk , A:B
1b8j , A:B	1dm0 , A:D	1go3 , E:F	1jb0 , A:E	1k8k , C:F	1poi , A:B	1wgj , A:B	4rub , A:T
1b8m , A:B	1dm0 , A:E	1gpe , A:B	1jb0 , A:C	1k8k , D:F	1pp2 , L:R	1xso , A:B	
1b9m , A:B	1dor , A:B	1gpw , A:B	1jb0 , C:E	1kpe , A:B	1prc , C:H	1ypi , A:B	
1be3 , G:A	1dtw , A:B	1gux , A:B	1jb0 , B:C	1kqf , B:C	1prc , C:L	1ytf , C:D	
1bjn , A:B	1dxt , A:B	1h2a , L:S	1jb0 , A:D	1ktd , A:B	1prc , C:M	2aai , A:B	
1brm , A:B	1e8o , A:B	1h2r , L:S	1jb0 , A:D	1l7v , A:C	1qae , A:B	2ae2 , A:B	
1byf , A:B	1e9z , A:B	1h8e , A:D	1jb0 , C:D	1ld8 , A:B	1qax , A:B	2ahj , A:B	

Non-obligate Complexes							
1a14 , L:N	1bml , A:C	1eai , A:C	1fq1 , A:B	1i4d , B:D	1jsu , B:C	1n2c , B:E	2btc , E:I
1a14 , H:N	1buh , A:B	1eay , A:C	1fqj , A:C	1i4d , A:D	1jsu , A:C	1n2c , A:E	2btf , A:P
1a2k , B:C	1c1y , A:B	1ebd , A:C	1frv , A:B	1i7w , A:B	1jtg , A:B	1n2c , B:F	2mta , A:L
1a4y , A:B	1c4z , A:D	1ebd , B:C	1fss , A:B	1i85 , B:D	1jw9 , B:D	1nbf , A:D	2mta , A:C
1acb , E:I	1cc0 , A:E	1eer , A:B	1gaq , A:B	1i8l , A:C	1k5d , A:B	1nf5 , A:B	2mta , H:L
1agr , E:A	1cgi , E:I	1efu , A:B	1gcq , B:C	1ib1 , B:E	1kcg , A:C	1noc , A:B	2pcb , A:B
1akj , B:D	1cmx , A:B	1efx , A:D	1gh6 , A:B	1ib1 , A:E	1kcg , B:C	1pdk , A:B	2pcc , A:B
1akj , A:D	1cs4 , A:C	1eja , A:B	1gl1 , A:I	1icf , B:I	1kkl , A:H	1qbk , B:C	2prg , B:C
1ar1 , B:D	1cs4 , B:C	1es7 , C:B	1gla , F:G	1ijk , A:B	1kkl , C:H	1qgw , A:C	2sic , E:I
1avg , H:I	1cse , I:E	1es7 , A:B	1gp2 , A:B	1ijk , A:C	1kmi , Y:Z	1rlb , A:E	2tec , E:I
1avw , A:B	1cvs , A:C	1eth , A:B	1grn , A:B	1is8 , C:M	1kxp , A:D	1rlb , C:E	3hhr , A:B
1avx , A:B	1d4x , A:G	1euv , A:B	1gvn , A:B	1is8 , B:L	1kyo , O:W	1rlb , B:E	3sgb , E:I
1avz , B:C	1d5x , A:C	1evt , A:C	1gzs , A:B	1is8 , E:O	1lb1 , A:B	1rrp , A:B	3ygs , C:P
1awc , A:B	1de4 , C:A	1f02 , I:T	1h2k , A:S	1is8 , D:N	1lpb , A:B	1stf , E:I	4htc , H:I
1ay7 , A:B	1dev , A:B	1f34 , A:B	1h59 , A:B	1is8 , A:K	1m10 , A:B	1t7p , A:B	4sgb , E:I
1azz , A:D	1df9 , B:C	1f3v , A:B	1hlu , A:P	1is8 , D:O	1m1e , A:B	1tab , E:I	
1azz , A:D	1dfj , E:I	1f80 , A:E	1hwg , A:C	1is8 , A:L	1m4u , A:L	1tgs , I:Z	
1b6c , A:B	1doa , A:B	1fak , H:T	1hwg , A:B	1is8 , E:K	1mah , A:F	1toc , B:R	
1b9y , A:C	1du3 , A:D	1fg9 , B:C	1hzz , B:C	1is8 , C:N	1mbu , A:C	1uea , A:B	
1bdj , A:B	1du3 , A:F	1fg9 , A:C	1i2m , A:B	1is8 , B:M	1ml0 , A:D	1wq1 , G:R	
1bi8 , A:B	1dx5 , M:I	1fin , A:B	1i3o , D:E	1itb , A:B	1mr1 , A:D	1ycs , A:B	
1bkd , R:S	1e6e , A:B	1fle , E:I	1i3o , B:E	1jch , A:B	1n2c , A:F	1zbd , A:B	

Table 2: Description of the subsets of features used in this study.

Name	Feature Type	Interacting Chains	DDIs
PPID-AT	atom type	✓	-
PPID-AA	amino acid	✓	-
PPID-ATD	atom type	-	✓
PPID-AAD	amino acid	-	✓

A posterior step was to identify the 317 unique domains present in the interface of at least one complex in the dataset. Considering all pairs of domains, the desolvation energies for all atoms and amino acids present in each interacting domains were calculated using Eq. (1) and finally each complex had 171 atom type and 210 amino acid type features. By using desolvation energies for different types of features, four subsets of features for prediction and evaluation were generated (Table 2). The names of the subsets are PPID-X where X is AT for atom type, AA for amino acid pairs, ATD for atoms in interacting domains (DDIs) or AAD for amino acid pairs in interacting domains.

2.3 Prediction Methods

2.3.1 Linear Dimensionality Reduction

One of the approaches we have used for prediction is LDR. The basic idea of LDR is to represent an object of dimension n as a lower-dimensional vector of dimension d, achieving this by performing a linear transformation. We consider two classes, ω_1 and ω_2, represented by two normally distributed random vectors $\mathbf{x}_1 \sim N(\mathbf{m}_1, \mathbf{S}_1)$ and $\mathbf{x}_2 \sim N(\mathbf{m}_2, \mathbf{S}_2)$, respectively, with p_1 and p_2 the *a priori* probabilities. After the LDR is applied, two new random vectors $\mathbf{y}_1 = \mathbf{A}\mathbf{x}_1$ and $\mathbf{y}_2 = \mathbf{A}\mathbf{x}_2$, where $\mathbf{y}_1 \sim N(\mathbf{A}\mathbf{m}_1; \mathbf{A}\mathbf{S}_1\mathbf{A}^t)$ and $\mathbf{y}_2 \sim N(\mathbf{A}\mathbf{m}_2; \mathbf{A}\mathbf{S}_2\mathbf{A}^t)$ with $\mathbf{m}_i$ and $\mathbf{S}_i$ being the mean vectors and covariance matrices in the original space, respectively. The aim of LDR is to find a linear transformation matrix $\mathbf{A}$ in such a way that the new classes ($\mathbf{y}_i = \mathbf{A}\mathbf{x}_i$) are as separable as possible. Let $\mathbf{S}_W = p_1\mathbf{S}_1 + p_2\mathbf{S}_2$ and $\mathbf{S}_E = (\mathbf{m}_1 - \mathbf{m}_2)(\mathbf{m}_1 - \mathbf{m}_2)^t$ be the within-class and between-class scatter matrices respectively. Various criteria have been proposed to measure this separability [17]. We consider the following two LDR methods:

(a) The heteroscedastic discriminant analysis (HDA) approach [17], which aims to obtain the matrix $\mathbf{A}$ that maximizes the following function, which is optimized via eigenvalue decomposition:

$$J_{HDA}(\mathbf{A}) = tr\left\{(\mathbf{A}\mathbf{S}_W\mathbf{A}^t)^{-1}\left[\mathbf{A}\mathbf{S}_E\mathbf{A}^t - \mathbf{A}\mathbf{S}_W^{\frac{1}{2}}\frac{p_1\log(\mathbf{S}_W^{-\frac{1}{2}}\mathbf{S}_1\mathbf{S}_W^{-\frac{1}{2}}) + p_2\log(\mathbf{S}_W^{-\frac{1}{2}}\mathbf{S}_2\mathbf{S}_W^{-\frac{1}{2}})}{p_1p_2}\mathbf{S}_W^{\frac{1}{2}}\mathbf{A}^t\right]\right\}. \quad (2)$$

(b) The Chernoff discriminant analysis (CDA) approach [17], which aims to maximize the following function, which is maximized via a gradient-based algorithm:

$$J_{CDA}(\mathbf{A}) = tr\{p_1p_2\mathbf{A}\mathbf{S}_E\mathbf{A}^t(\mathbf{A}\mathbf{S}_W\mathbf{A}^t)^{-1} + \log(\mathbf{A}\mathbf{S}_W\mathbf{A}^t) - p_1\log(\mathbf{A}\mathbf{S}_1\mathbf{A}^t) - p_2\log(\mathbf{A}\mathbf{S}_2\mathbf{A}^t)\}. \quad (3)$$

In order to classify each complex, first a linear algebraic operation $\mathbf{y} = \mathbf{A}\mathbf{x}$ is applied to the n-dimensional vector, obtaining $\mathbf{y}$, a d-dimensional vector, where d is ideally much smaller than n. The linear transformation matrix $\mathbf{A}$ corresponds to the one obtained by one of the LDR methods, namely HDA or CDA. The resulting vector $\mathbf{y}$ is then passed through a Quadratic Bayesian (QB) classifier [17], which is the optimal classifier for normal distributions. For additional tests, a linear Bayesian (LB) classifiers is considered, by deriving a Bayesian classifier with a common covariance matrix: $\mathbf{S} = \mathbf{S}_1 + \mathbf{S}_2$.

2.3.2 Support Vector Machines

SVMs are well known machine learning techniques used for classification, regression and other tasks. The aim of SVM is to find the support vectors (most difficult vectors to be classified), and derive a linear classifier, which ideally separates the space into two regions. Classification is normally inefficient when using a linear classifier, because the data is not linearly separable, and so the use of kernels is crucial in mapping the data onto a higher dimensional space in which the classification is much more efficient. There are number of kernels that can be used in SVM models. In our model, we use polynomial, radial basis function (RBF) and sigmoid.

3. RESULTS AND DISCUSSIONS

3.1 Experimental Settings

For the LDR schemes, four different classifiers were implemented and evaluated, namely the combinations of HDA and CDA, and QB and LB classifiers. In a 10-fold cross validation setup, reductions to dimensions $d = 1, \ldots, 20$ were performed, followed by QB and LB, and the maximum average classification accuracy was recorded for each classifier. The best accuracy for each method for each dataset is bolded to indicate the classifier that performed the best for that dataset. Principal component analysis (PCA) was used as a pre-processing step to eliminate ill-conditioned matrices present in the LDR step. To select the principal components, we used different threshold values (from $\lambda_{max}10^{-2}$ to $\lambda_{max}10^{-7}$), where λ_{max} is the largest eigenvalue of the scatter matrix. The results for the threshold that achieves the highest accuracy are reported.

The SVM was also trained in a 10-fold cross validation setup with three kernels: RBF, polynomial and sigmoid. The training was carried out with the LIBSVM package [18]. A grid search was performed on the parameters gamma and C, choosing the ones that gives the maximum average accuracy for all kernels. For the polynomial kernel, the degree of the polynomial was set to 3.

The subsets of features shown in Table 2 were used for prediction. To analyze the power of desolvation energy in discriminating obligate and non-obligate complexes, NOXclass [5] was also applied to our binary-PPID dataset. The following four interface properties were analyzed, since in [5], these properties were recognized as the best ones for prediction of different types of protein protein interactions:

- Interface area
- Interface area ratio
- Amino acid composition of the interface
- Correlation between amino acid compositions of interface and protein surface

We used NACCESS [19] to calculate solvent accessible surface area (SASA). After running the classifiers in a 10-fold cross validation procedure for all subsets of features, the average accuracies were computed. The accuracy for each individual fold was computed as follows: $acc = (TP + TN)/N_f$, where TP and TN are the true positive (obligate) and true negative (non-obligate) counters respectively, and N_f is the total number of complexes in the test set of the corresponding fold.

Table 3: Prediction results for SVM and LDR classifiers on binary-PPID dataset.

	SVM			LDR			
	RBF	Polynomial	Sigmoid	Linear		Quadratic	
				HDA	CDA	HDA	CDA
PPID-AT	**77.78**	76.83	72.70	71.76	74.08	72.73	**74.55**
PPID-AA	**75.56**	71.43	71.11	71.46	**71.81**	71.46	65.07
PPID-ATD	**70.30**	67.62	67.43	68.66	68.06	**70.25**	68.97
PPID-AAD	**69.84**	67.62	66.35	67.34	66.12	**68.32**	62.80
PPID-NOXclass	**72.38**	69.84	69.52	68.89	**71.80**	67.71	68.97

3.2 Analysis of Prediction

The results of SVM and LDR classifiers with different subsets of features are depicted in Table 3. For SVM, it is clearly seen that the RBF kernel performs better that polynomial and sigmoid kernels for all subsets of features. The atom type features present in interacting chains (PPID-AT) are best classified with SVM and a RBF kernel, achieving an average accuracy of 77.78%, while accuracy for atom type features present in interacting domains (PPID-ATD) is 70.30%. Similarly, the subset of amino acid type features present in interacting chains (PPID-AA) with 75.56% classification accuracy yields more efficient predictions than using the subset of amino acid type features present in DDIs (PPID-AAD) with 69.84% classification accuracy. Furthermore, the subset based on NOXclass features (with best accuracy of 72.38%) perform worse than the best subset based on desolvation energy properties (PPID-AT) on a SVM classifier.

For LDR, the best accuracy, 74.55%, is achieved by CDA with the quadratic classifier, which is still lower than the best accuracy achieved by SVM. Note that both of them are on the PPID-AT subset. Additionally, as in SVM, subsets of atom and amino acid type features present in interacting chains perform better than those in DDIs. Also, the NOXclass subset of features (PPID-NOXclass) yields lower accuracy (71.80%) than PPID-AT, which is based on calculation of desolvation energies only, and also DDI subsets.

Generally, it can be concluded that in our binary-PPID dataset:

(a) SVM with RBF kernel performs better than LDR methods in all subsets of features

(b) Amino acid type features (for both PPID-AA and PPID-AAT subsets) yeild lower accuracies than atom type features (PPID-AT and PPID-ATD) for both LDR and SVM classifiers

(c) Although the performance of both SVM and LDR classifiers are lower for subsets of DDI features (PPID-ATD and PPID-AAD) than subsets of interacting chain features (PPID-AT and PPID-AA), they are acceptable results.

(d) Desolvation energy properties are more powerful than four properties of NOXclass (interface area, interface area ratio, amino acid composition of the interface and correlation between amino acid compositions of interface and protein surface) in predicting obligate and non-obligate complexes.

3.3 Analysis of DDIs

As discussed earlier, the total number of DDIs among 317 existing domains of our binary-PPID dataset is $100,489$. After preprocessing and removing all zero-columns, we obtain only 256 DDI pairs of which 125 are obligate and 131 are non-obligate DDIs.

The most salient feature in our binary-PPID dataset is the fact that all DDIs are presented in either obligate or non-obligate complexes and there are no DDIs in both obligate and non-obligate. This clearly implies that the type of complex could just be predicted by the DDIs present in the interactions, achieving nearly perfect prediction rate of 100%. One could design a simple classifier that contains binary features and indicates the presence or absence of the DDI in the complex, and then a simple rule that checks those binary flags. However, this would not be the case when predicting new unknown complexes (not in this dataset). That is, when using the training data to test the classifier. When cross-validation is applied, as it is done in this paper, presence of a DDI in the training set may not imply its presence or absence in the test set. In addition, it is expected, though it would not be the case, that the DDI desolvation properties are much more informative than simply binary features indicating the presence or absence of the DDI in the complex.

We performed a visual analysis on our DDIs and discovered that from 317 existing domains in our binary-PPID dataset, 135 are present only in obligate DDIs, 158 are present only in non-obligate DDIs and 21 domains are in both obligate and non-obligate DDIs. We re-ordered the domain IDs based on their types (obligate, both and non-obligate). To provide a visual insight of the distribution of the DDIs in the different complexes, a schematic view of the DDIs in the dataset is shown in Figure. 1. It is clearly seen that the most homo-domain pairs are in obligate complexes (i.e. they lie on the diagonal line ($x = y$) of the plot). Only a small part of the domain IDs are common. This also implies we can achieve a reasonable prediction only by finding the domains of each unknown complex. This is an interesting issue that deserves a lot of attention, and that we are currently investigating.

4. CONCLUSION

We have proposed an approach for prediction and analysis of obligate and non-obligate protein complexes. We have investigated various interface properties of these interactions including atom and amino acid types present in interacting chains or domains. Various features are extracted from each complex, including the desolvation energies for atom and amino acid type pairs and also NOXclass properties. The classification is performed via different LDR methods that involve homoscedastic and heteroscedastic criteria and SVM with different kernels, namely RBF, polynomial and sigmoid.

The results on our binary-PPID dataset, which is a joint and modified version of two well-known datasets, show that the SVM classifier with 77.78% accuracy achieves much better classification performance, even better than LDR schemes coupled with quadratic and linear classifiers for all subset of features. The results also demonstrate that desolvation energy is better than interface area and composition for predicting obligate and non-obligate complexes.

Furthermore, visual and numerical analysis on DDIs show that (i) most homo-domain pairs are in obligate interactions and (ii) no common DDI is present in obligate and non-obligate complexes

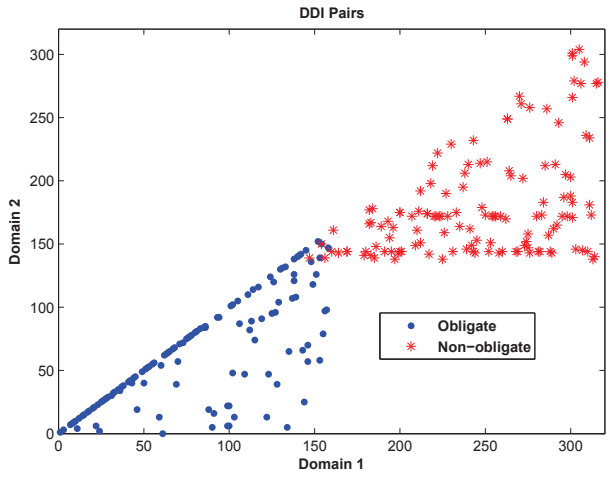


Figure 1: Schematic view of the DDI pairs in obligate and non-obligate interactions.

and all DDIs are present in either obligate or non-obligate complexes.

Our future work involves the use of other features such as residual vicinity, shape of the structure of the interface, secondary structure, planarity, physicochemical features, hydrophobicity, structure of domains and many others in our dataset, and also identifying pseudo-domains and motifs present in interacting proteins.

Acknowledgments

This research work has been partially supported by NSERC, the Natural Sciences and Research Council of Canada, grant No. RG-PIN 261360, and the University of Windsor, internal Start-up and Office of Research Services equipment grants.

5. REFERENCES

[1] I. Nooren and J. Thornton, "Diversity of protein-protein interactions," *EMBO Journal*, vol. 22, no. 14, pp. 3846–3892, 2003.

[2] S. Jones and J. M. Thornton, "Principles of protein-protein interactions," *Proc. Natl Acad. Sci, USA*, vol. 93, no. 1, pp. 13–20, 1996.

[3] L. LoConte, C. Chothia, and J. Janin, "The atomic structure of protein-protein recognition sites," *J Mol Biol*, vol. 285, no. 5, pp. 2177–2198, 1999.

[4] O. K. A. Zen, C. Micheletti and R. Nussinov, "Comparing interfacial dynamics in protein-protein complexes: an elastic network approach," *BMC Structural Biology*, vol. 10, no. 26, 2010, doi: 10.1186/1472-6807-10-26.

[5] H. Zhu, F. Domingues, I. Sommer, and T. Lengauer, "Noxclass: Prediction of protein-protein interaction types," *BMC Bioinformatics*, vol. 7, no. 27, 2006, doi:10.1186/1471-2105-7-27.

[6] M. S. S. Kottha, "Classifying permanent and transient protein interactions," *German Conference on Bioinformatics*, vol. 83GI, pp. 54–63, 2006.

[7] S. De, O. Krishnadev, N. Srinivasan, and N. Rekha, "Interaction preferences across protein-protein interfaces of obligatory and non-obligatory components are different." *BMC Structural Biology*, vol. 5, no. 15, 2005.

[8] J. V. Eichborn, S. Günther, and R. Preissner, "Structural features and evolution of protein-protein interactions." *Intenational Conference of Genome Informatics.*, vol. 22, pp. 1–10, 2010.

[9] S. H. Park, J. Reyes, D. Gilbert, J. W. Kim, and S. Kim, "Prediction of protein-protein interaction types using association rule based classification," *BMC Bioinformatics*, vol. 10, no. 36, 2009, doi:10.1186/1471-2105-10-36.

[10] L. Rueda, S. Banerjee, M. M. Aziz, and M. Raza, "Protein-protein interaction prediction using desolvation energies and interface properties," proceedings of the 2nd. IEEE International Conference on Bioinformatics & Biomedicine (BIBM 2010), pp. 17–22, 2010.

[11] J. Mintseris and Z. Weng, "Structure, function, and evolution of transient and obligate protein-protein interactions," *Proc Natl Acad Sci, USA*, vol. 102, no. 31, pp. 10 930–10 935, 2005.

[12] S. G. J. V. Eichborn and R. Preissner, "Structural features and evolution of protein-protein interactions," *Genome Inform*, vol. 22, pp. 1–10, 2010.

[13] [Online]. Available: http://pfam.sanger.ac.uk/

[14] C. Camacho and C. Zhang, "FastContact: rapid estimate of contact and binding free energies," *Bioinformatics*, vol. 21, no. 10, pp. 2534–2536, 2005.

[15] C. Zhang, G. Vasmatzis, J. L.Cornette, and C. DeLisi, "Determination of atomic desolvation energies from the structures of crystallized proteins," *J. Mol. Biol.*, vol. 267, pp. 707–726, 1997.

[16] H. Berman, J. Westbrook, Z. Feng, G. Gilliland, T. Bhat, H. Weissig, I. Shindyalov, and P. Bourne, "The Protein Data Bank," *Nucleic Acids Research*, vol. 28, pp. 235–242, 2000.

[17] L. Rueda and M. Herrera, "Linear Dimensionality Reduction by Maximizing the Chernoff Distance in the Transformed Space," *Pattern Recognition*, vol. 41, no. 10, pp. 3138–3152, 2008.

[18] C. L. C. Chang, "Libsvm: a library for support vector machines," last date accessed: May 31, 2011. [Online]. Available: http://www.csie.ntu.edu.tw/ cjlin/papers/libsvm.pdf

[19] S. Hubbard and J. Thornton, "Naccess," last date accessed: May 31, 2011. [Online]. Available: www.bioinf.manchester.ac.uk/naccess/

Using Physicochemical Properties of Amino Acids to induce Graphical Models of Residue Couplings

K. S. M. Tozammel Hossain[†], Chris Bailey-Kellogg[‡], Alan M. Friedman[§], Michael J. Bradley[*], Nathan Baker[**], and Naren Ramakrishnan[†]

[†]Department of Computer Science, Virginia Tech, Blacksburg, VA 24061, USA
[‡]Department of Computer Science, Dartmouth College, Hanover, NH 03755, USA
[§]Department of Biological Sciences, Purdue University, West Lafayette, IN 47907, USA
[*] Department of Molecular Biophysics and Biochemistry, Yale University, New Haven, CT 06520, USA
[**]Pacific Northwest National Laboratory, Richland, WA 99352, USA

tozammel@vt.edu, cbk@cs.dartmouth.edu, afried@purdue.edu, michael.bradley@yale.edu, nathan.baker@pnl.gov, naren@cs.vt.edu

ABSTRACT

Residue coupling in protein families is an important indicator for structural and functional conservation. Two residues are coupled if changes of amino acid at one residue location are correlated with changes in the other. Many algorithmic techniques have been proposed to discover couplings in protein families. These approaches discover couplings over amino acid combinations but do not yield mechanistic or other explanations for such couplings. We propose to study couplings in terms of amino acid classes such as polarity, hydrophobicity, size, and reactivity, and present two algorithms for learning probabilistic graphical models of amino acid class-based residue couplings. Our probabilistic graphical models provide a sound basis for predictive, diagnostic, and abductive reasoning. Further, our methods can take optional structural priors into account for building graphical models. The resulting models are useful in assessing the likelihood of a new protein to be a member of a family and for designing new protein sequences by sampling from the graphical model. We apply our approaches to understand couplings in two protein families: Nickel-responsive transription factors (NikR) and G-protein coupled receptors (GPCRs). The results demonstrate that our graphcial models based on sequences, physicochemical properties, and protein structure are capable of detecting amino acid class-based couplings between important residues that play roles in activities of these two families.

Keywords

Residue coupling, graphical models, amino acid classes, evolutionary co-variation.

BIOKDD 2011 August 2011, San Diego, CA, USA

1. INTRODUCTION

Proteins are grouped into families based on similarity of function and structure. It is generally assumed that evolutionary pressures in protein families to maintain structure and function manifest in the underlying sequences. Two well-known types of constraints are conservation and coupling. The most widely studied constraint is conservation of individual residues. Within a protein family, a particular residue position is *conserved* if a particular amino acid occurs at that residue position for most of the members in the family [3]. Conservation of residues usually occurs at functionally and/or structurally important sites within a protein fold (shared by the protein family). For example in Figure 1(a), a multiple sequence alignment (MSA) of 10 sequences, the second residue is 100% conserved with occurrence of amino acid "W".

A variety of recent studies have used MSAs to calculate correlations in mutations at several positions within an alignment and between alignments [15, 10, 19, 14]. These correlations have been hypothesized to result from structural/functional coupling between these positions within the protein [8]. Two residues are *coupled* if certain amino acid combinations occur at these positions in the MSA more frequently than others [15, 7]. For example, residues 3 and 8 are coupled in Fig. 1(d) because the presence of "K" (or "M") at the third residue co-occurs with "T" (or "V") at the eighth residue position. Going beyond sequence conservation, couplings provide additional information about potentially important structural/functional connections between residues within a protein family. Previous studies [15, 8, 10] show that residue couplings play key roles in transducing signals in cellular systems.

In this paper, we study residue couplings that manifest at the level of amino acid classes rather than just the occurrence of particular letters within an MSA. Our underlying hypothesis is that if structural and functional behaviors are the underlying cause of residue couplings within MSAs, then couplings are more naturally studied at the level of amino acid properties. We are motivated by the prior work of Thomas et al. [9, 10] which proposes probabilistic graphical models for capturing couplings in a protein family in terms of amino acids. Graphical models are useful for support-

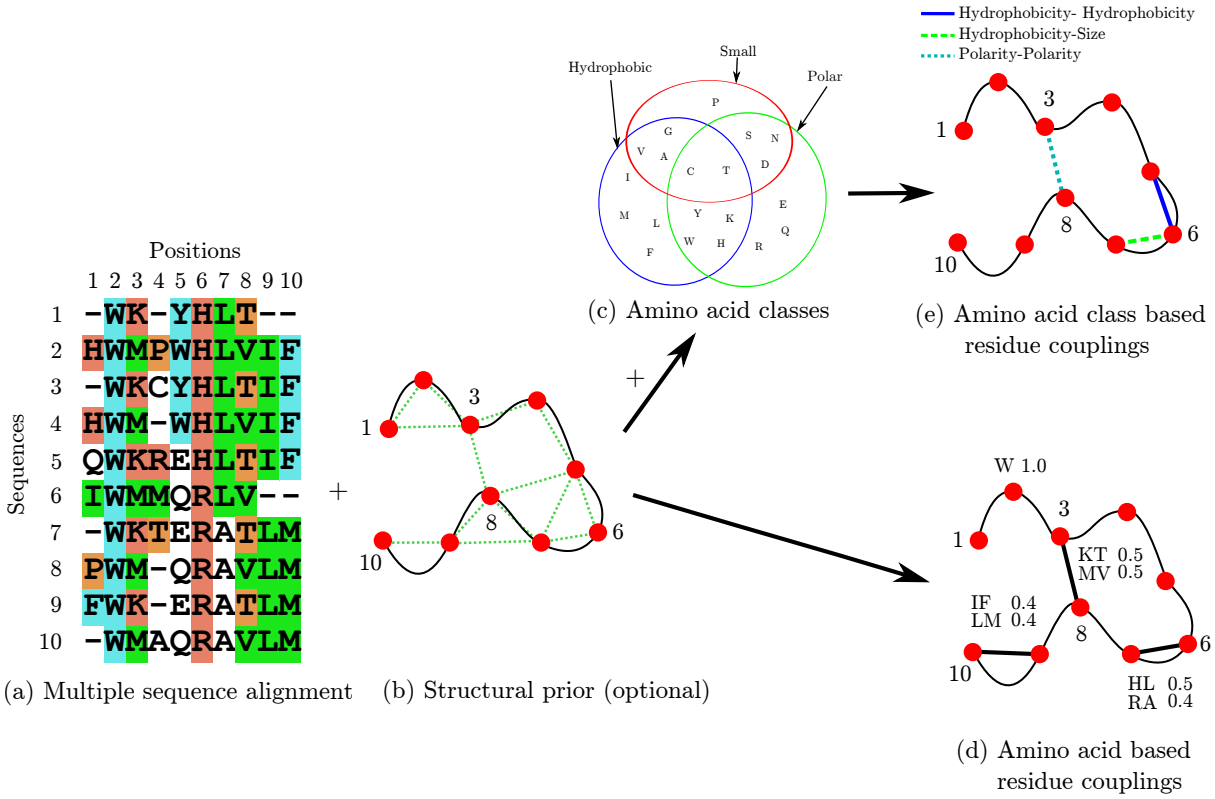


Figure 1: Inferring graphical models from an MSA of a protein family: (a)-(c) illustrate input to our models and (d),(e) illustrate two different residue coupling networks.

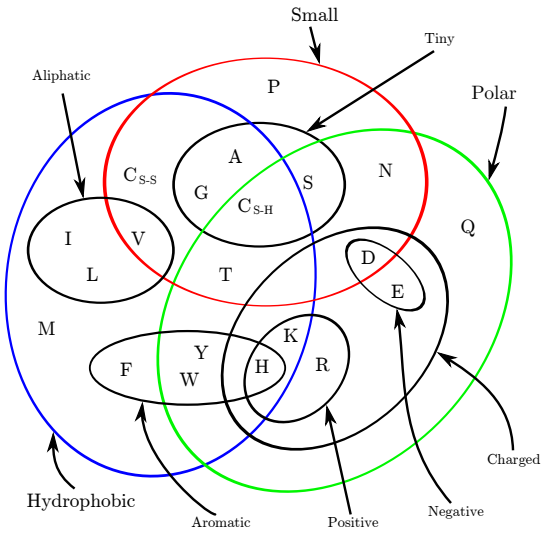


Figure 2: Taylor's classification: a Venn diagram depicting classes of amino acids based on physicochemical properties. Figure redrawn from [17].

ing better investigation, characterization, and design of proteins. The above works infer an undirected graphical model for couplings given an MSA where each node (variable) in the graph corresponds to a residue (column) in the MSA and an edge between two residues represents significant correlation between them. Figure 1(a),(b) illustrates the typical input (an MSA and a structural prior) and Figure 1(d) is an output (undirected graphical model) of the procedure of Thomas et al. In the output model (see Fig. 1(d)), three residue pairs—(3,8), (6,7), and (9,10)—are coupled.

Evolution is the key factor determining the functions and structures of proteins. It is assumed that the type of amino acid at each residue position within a protein structure is (at least somewhat) constrained by its surrounding residues. Therefore, explaining the couplings in terms of amino acid classes is desirable. To achieve this, we consider amino acid classes based on physicochemical properties (see Fig. 2).

Graphical models can be made more expressive if we represent the couplings (edges in the graphs) in terms of underlying physicochemical properties. Figure 1(c) is a Venn diagram of three amino acid classes–polarity, hydrophobicity, and size. Figure 1(e) illustrates three couplings in terms of amino acid classes. For example, residue 3 and residue 8 are coupled in term of "polarity-polarity", which means correlated changes of polarities occur at these two positions – a change from polar to nonpolar amino acids at residue 3, for instance, induces concomitant change from polar to nonpolar amino acid at residue 8. Similarly, residue 6 and residue 7 are also correlated since a change from hydrophobic to hydrophilic amino acids at residue 6 induces a change

from big to small amino acids at residue 7. There is no edge between residue 5 and residue 7, however, because they are independent given residue 6. Hence, the coupling between residue 5 and residue 7 is explained via couplings (5,6) and (6,7). This is one of the key features of undirected graphical models as they help distinguish direct couplings from indirect couplings. Note that the coupling between residue 9 and residue 10 (originally present in Fig. 1(d)) does not occur in Figure 1(e) due to class conservation in residues 9 and 10. Also note that the coupling between residue 5 and residue 6 in Figure 1(e) is not apparent in Figure 1(d). Class-based representations of couplings hence recognize a different set of relationships than amino acid value-based couplings. We show how the class-based representation leads to more explainable models and suggest alternative criteria for protein design.

The key contributions of this paper are as follows:

1. We investigate whether residue couplings manifest at the level of amino acid classes and answer this question in the affirmative for the two protein families studied here.

2. We design new probabilistic graphical models for capturing residue coupling in terms of amino acid classes. Like the work of Thomas et al. [10] our models are precise and give explainable representations of couplings in a protein family. They can be used to assess the likelihood of a protein to be in a family and thus constitute the driver for protein design.

3. We demonstrate successful applications to the NikR and GPCR protein families, two key demonstrators for protein constraint modeling.

The rest of the paper is organized as follows. We review related literature in Section 2. Methodologies for inferring graphical models are described in Section 3. Experimental results are provided in Section 4 followed by a discussion in Section 5.

2. LITERATURE REVIEW

Early research on correlated amino acids was conducted by Lockless and Ranganathan [15]. Through statistical analysis they quantified correlated amino acid positions in a protein family from its MSA. Their work is based on two hypotheses, which are derived from empirical observation of sequence evolution. First, the distribution of amino acids at a position should approach their mean abundance in all proteins if there is a lack of evolutionary constraint at that position; deviance from mean values would, therefore, indicate evolutionary pressure to prefer particular amino acid(s). Second, if two positions are functionally coupled, then there should be mutually constrained evolution at the two positions even if they are distantly positioned in the protein structure. The authors developed two statistical parameters for conservation and coupling based on the above hypothesis, and use these parameters to discover conserved and correlated amino acid positions. In their SCA method, a residue position in an MSA of the family is set to its most frequent amino acid, and the distribution of amino acids at another position (with deviant sequence at the first position removed) is observed. If the observed distribution of amino acids at the other position is significantly different from the distribution in the original MSA, then these two positions are considered to be coupled. Application of their method on the PDZ protein family successfully determined correlated amino acids that form a protein-protein binding site.

Valdar surveyed different methods for scoring residue conservation [17]. Quantitative assessment of conservation is important because it sets a baseline for determining coupling. In particular, many algorithms for detecting correlated residues run into trouble when there is an 'in between' level of conservation at a residue position. In this survey, the author investigates about 20 conservation measures and evaluates their strengths and weaknesses.

Fodor and Aldrich reviewed four broad categories of measures for detecting correlation in amino acids [11]. These categories are: 1) Observed Minus Expected Squared Covariance Algorithm (OMES), 2) Mutual Information Covariance Algorithm (MI), 3) Statistical Coupling Analysis Covariance Algorithm (SCA; mentioned above), and 4) McLachlan Based Substitution Correlation (McBASC). They applied these four measures on synthetic as well as real datasets and reported a general lack of agreement among the measures. One of the reasons for the discrepancy is sensitivity to conservation among the methods, in particular, when they try to correlate residues of intermediate-level conservation. The sensitivity to conservation shows a clear trend with algorithms favoring the order McBASC > OMES > SCA > MI.

Although current research is successful in discovering conserved and correlated amino acids, they fail to give a formal probabilistic model. Thomas *et al.* [10] is a notable expection. This paper differentiates between direct and indirect correlations which previous methods did not. Moreover, the models discovered by this work can be extended into *differential* graphical models which can be applied to protein families with different functional classes and can be used to discover subfamily-specific constraints (conservation and coupling) as opposed to family-wide constraints.

The above research on coupling and conservation do not aim to model evolutionary processes directly. Yeang and Haussler, in contrast, suggest a new model of correlation in and across protein families employing evolution [19]. They refer to their model as a *coevolutionary model* and their key claims are: coevolving protein domains are functionally coupled, coevolving positions are spatially coupled, and coevolving positions are at functionally important sites. The authors give a probabilistic formulation for the model employing a phylogenetic tree for detecting correlated residues.

A more recent work, by Little and Chen [14], studies correlated residues using mutual information to uncover evolutionary constraints. The authors show that mutual information not only captures coevolutionary information but also non-coevolutionary information such as conservation. One of the strong non-coevolutionary biases is stochastic bias. By first calculating mutual information between two residues which have evolved randomly (referred to as random mutual information), the authors then study relationships with other mutual information quantities to detect the presence of non-coevolutionary biases.

3. METHODS

A multiple sequence alignment $\mathcal{S}$ allows us to summarize each residue position in terms of the probabilities of encoun-

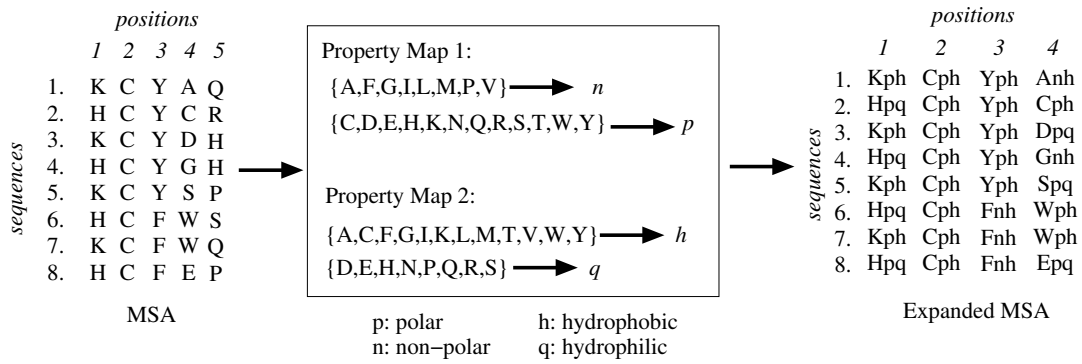


Figure 3: Expansion of a multiple sequence alignment into an 'inflated MSA'. Two classes—polarity and hydrophobicity—are used for illustration. Each column in the MSA is mapped to three columns in the expanded MSA.

tering each of the 20 amino acids (or a gap) in that position. Let $V = \{v_1, \ldots, v_n\}$ be a set of random variables, one for each residue position. The MSA then gives a distribution of amino acids for each random variable. We present two different classes of probabilistic graphical models to detect couplings. These inferred graphical models capture conditional dependence and independence among residues, as revealed by the MSA. The first approach uses an undirected graphical model (UGM), also known as a Markov random field. The second method employs a specific hierarchical latent class model (HLCM) which is a two-layered Bayesian network.

3.1 UGMs from Inflated MSAs

This approach can be viewed as an extension of the work of Thomas et al. [10]. It induces an undirected graphical model, $G = (V, E)$, where each node, $v \in V$, corresponds to a random variable and each edge, $(u, v) \in E$, represents a direct relationship between random variables u and v. In our problem setting, a node of G corresponds to a residue position (a column of the given MSA) and each edge represents a coupling between two residues. In this method, we redefine the approach of Thomas et al. [10] to discover MSA residue position couplings in terms of amino acid classes rather than residue values.

3.1.1 Inflated MSA

We augment the MSA $\mathcal{S}$ of a protein family by introducing extra 'columns' for each residue. Let l be the number of amino acid classes and $\mathcal{A}_i$ be the alphabet for the ith class where $1 \leq i \leq l$. Legal vocabularies for the classes can be constructed with the help of Taylor's diagram (see Fig. 2). For example, possible classes are polarity, hydrophobicity, size, charge, and aromaticity. Moreover, we may consider the amino acid sequence of a column as a "amino acid name" class. These classes take different values; e.g., the polarity class takes two values: polar and non-polar. Each column of $\mathcal{S}$ is mapped to l subcolumns to obtain an inflated MSA $\mathcal{S}_e$ where the extra columns (referred to as subcolumns) encode the corresponding class values. We use v_{ik} to denote the kth subcolumn of residue v_i. Figure 3 illustrates the above procedure for obtaining an inflated alignment $\mathcal{S}_e$. (A gap character in $\mathcal{S}$ is mapped to a gap character in $\mathcal{S}_e$.)

3.1.2 Detecting Coupled Residues

Couplings between residues can be quantified by many statistical and information-theoretic metrics [11]. In our model, we use conditional mutual information because it allows us to separate direct from indirect correlations. Recall that the *mutual information* (MI), $I(v_i, v_j)$, between residues v_i and v_j is given by:

$$I(v_i, v_j) = \sum_{a \in \mathcal{A}} \sum_{b \in \mathcal{A}} P(v_i = a, v_j = b) \cdot \log \frac{P(v_i = a, v_j = b)}{P(v_i = a) P(v_j = b)} \quad (1)$$

where the probabilities are all assessed from $\mathcal{S}$. If $I(v_i, v_j)$ is non-zero, then they are dependent, and each residue position (v_i or v_j) encodes information that can be used to predict the other. In the original *graphical models of residue coupling* (GMRC) model [10], Thomas et al. use conditional mutual information:

$$I(v_i, v_j | v_k) = \sum_{c \in \mathcal{A}^*} \sum_{a \in \mathcal{A}} \sum_{b \in \mathcal{A}} P(v_i = a, v_j = b | v_k = c) \cdot \log \frac{P(v_i = a, v_j = b | v_k = c)}{P(v_i = a | v_k = c) P(v_j = b | v_k = c)} \quad (2)$$

to construct edges, where the conditionals are estimated by subsetting residue k to its most frequently occurring amino acid types ($\mathcal{A}^* \subset \mathcal{A}$). The most frequently occurring amino acid types are those that appear in at least 15% of the original sequences in the subset. As discussed [15], such a bound is required in order to ensure sufficient fidelity to the original MSA and allow for evolutionary exploration.

For modeling residue position couplings in terms of amino acid classes, we use Eq. 2. As each residue in $\mathcal{S}_e$ has l columns, we consider all $O(l^2)$ pairs of columns for estimating mutual information between two residues. For calculating conditional mutual information in an inflated MSA, we condition a residue to its most appropriate class. The most appropriate class is the one that reduces the overall network score the most. The modified equation for conditional mutual information is as follows:

$$I_e(v_i, v_j | v_{kr}) = \sum_{p=1}^{l} \sum_{q=1}^{l} I_e(v_{ip}, v_{jq} | v_{kr}) \quad (3)$$

where

$$I_e(v_{ip}, v_{jq}|v_{kr}) = \sum_{c\in\mathcal{A}_r^*}\sum_{a\in\mathcal{A}_p}\sum_{b\in\mathcal{A}_q} P(v_{ip}=a, v_{jq}=b|v_{kr}=c) \cdot \log \frac{P(v_{ip}=a, v_{jq}=b|v_{kr}=c)}{P(v_{ip}=a|v_{kr}=c)P(v_{jq}=b|v_{kr}=c)} \quad (4)$$

Here $\mathcal{A}_i$ denote the alphabet of the ith amino acid class where $1 \leq i \leq l$. The conditional variable v_k is set to the rth class. If $I_e(v_i, v_j|v_{kr}) = 0$, then it implies that residue v_i and v_j are independent conditioned on the rth class of v_k. Observe that we can subset the residue v_k to any class out of l classes. We take the minimum of $I_e(v_i, v_j|v_{kr})$ for $1 \leq r \leq l$ to obtain the final mutual information between v_i and v_j.

3.1.3 Normalized Mutual Information

In an inflated MSA, the subcolumns corresponding to a residue take values from different alphabets of different sizes. Let v_{ip} and v_{jq} be two subcolumns that take values from alphabets $\mathcal{A}_p$ and $\mathcal{A}_q$ respectively. To understand the effect of the sizes of alphabets in mutual information score, we calculate pairwise mutual information of subcolumns for every residue pair and produce a scatter plot (see Fig. 4(a)).

In Fig. 4(a), we see that $MI(A, A)$ is dominating over $MI(P, P)$, $MI(H, H)$, and $MI(S, S)$. This is expected, because amino acids are of 21 types whereas polarity, hydrophobicity, and size have 3 types. We adopt the following equation to normalize mutual information scores proposed by Yao [18]:

$$I_{norm}(v_{ip}, v_{jq}|v_{kr}) = \frac{I(v_{ip}, v_{jq}|v_{kr})}{\min(H(v_{ip}|v_{kr}), H(v_{jq}|v_{kr})} \quad (5)$$

where $H(v_{ip}|v_{kr})$ and $H(v_{jq}|v_{kr})$ denote the conditional entropy.

3.1.4 Learning UGMs

Given an expanded MSA $\mathcal{S}_e$, we infer a graphical model by finding *decouplers* which are sets of variables that makes other variables independent. If two residues v_i and v_j are independent given v_k, then v_k is a decoupler for v_i and v_j. In this case, we add edges (v_i, v_k) and (v_j, v_k) to the graph. Thus the relationship between v_i and v_j is explained transitively by edges (v_i, v_k) and (v_j, v_k). Moreover, we can consider a prior that can be calculated from a contact graph of a representative member of the family. A prior gives a set of edges between residues which are close in three-dimensional structure. When a residue contact network is given as a prior, we consider each edge of the residue contact network as a potential candidate for couplings. Without a prior, we consider all pairwise residues for coupling. Algorithm 1 gives the formal details for inferring a graphical model.

Our algorithm builds the graph in a greedy manner. At each step, the algorithm chooses the edge from a set of possible couplings which scores best with respect to the current graph. The score of the graph is given by:

$$\mathrm{S}_{UGM}(G=(V,E)) = \sum_{v_i\in V}\sum_{v_j\notin \mathrm{N}(v_i)} I_e(v_i, v_j|\mathrm{N}(v_i)) \quad (6)$$

where $\mathrm{N}(v_i)$ is the set neighbors of v_i.

Algorithm 1 GMRC-Inf($\mathcal{S}, P$)

Input: $\mathcal{S}$ (multiple sequence alignment), P (possible edges)
Output: G (a graph that captures couplings in $\mathcal{S}$)

```
V = {v_1, v_2, ..., v_n}
E ← φ
s ← S_UGM(G = (V, E))
for all e = (v_i, v_j) ∈ P do
    C_e ← s − S_UGM(G = (V, {e}))
while stopping criterion is not satisfied do
    e ← arg max_{e∈P−E} C_e
    if e is significant then
        E ← E ∪ {e}
        label e based on the score
        s ← s − C_e
        for all e' ∈ P − E s.t e and e' share a vertex do
            C_{e'} ← s − S_UGM(G = (V, E ∪ {e'}))
return G = (V, E)
```

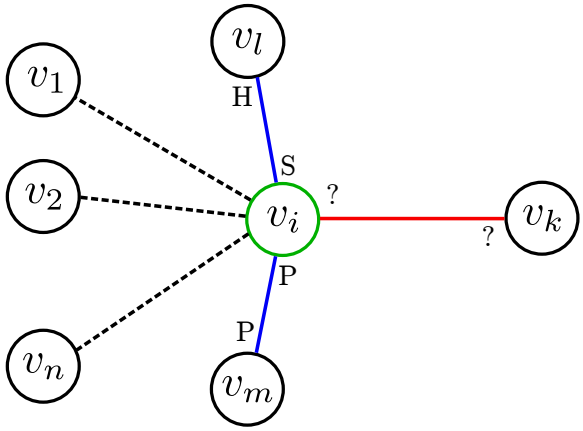


Figure 5: Class labeling of coupled edges. The blue edges are already added to the network and dashed edges are not. The red edge is under consideration for addition in the current iteration of the algorithm. The "?" takes any of the four classes: polarity (P), hydrophobicity (H), size (S), or the default amino acid values (A).

The calculation of conditional mutual information and labeling of edges with different properties is illustrated in Fig. 5. In Fig. 5, we consider edge (v_i, v_k) for addition to the graph where v_i already has two neighbors v_l and v_m. The edge (v_i, v_l) has the label S-H which means the coupling models v_i with respect to size and v_l with respect to hydrophobicity. Similarly, the edge (v_i, v_m) has the label P-P which means the coupling between v_i and v_m can be described with respect to their polarities. To evaluate the edge (v_i, v_k), we condition on v_m and v_l first and then condition v_k on any of the properties. We then sum up all $I_e(v_i, v_j)$, where $v_j \notin \{v_l, v_m, v_k\}$. The subsetting class of v_k for which we obtain a maximum for $\sum I_e(v_i, v_j)$ is the label that we finally assign to v_k (the question mark in Fig. 5) if the edge (v_i, v_k) is added. Similarly, we do the same calculation for v_k while subsetting only v_i, as the residue v_k does not have any neighbors in the current network.

Algorithm 1 can incorporate various stopping criteria: 1)

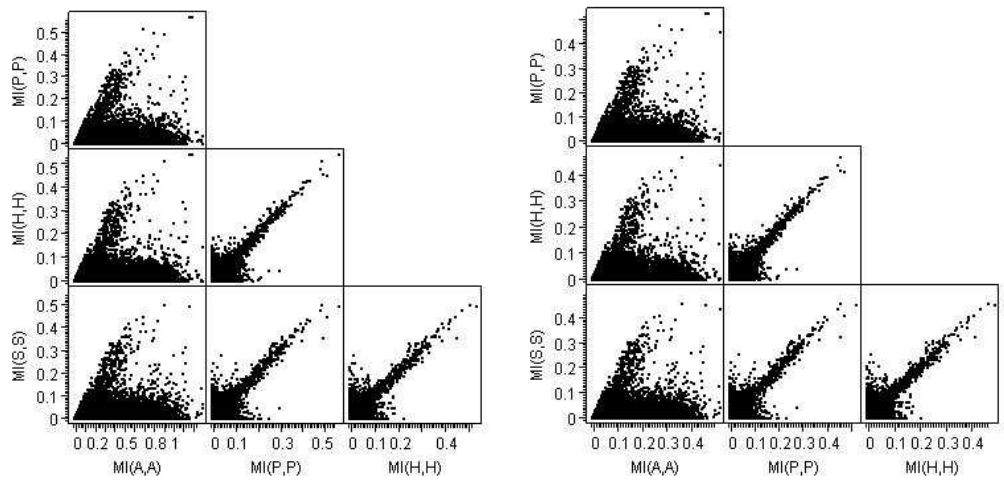


Figure 4: Effect of alphabet length on mutual information. Here, A,P,H,S denote amino acid, polarity, hydrophobicity, and size column respectively. (a) Scatter plot of mutual information for every residue pair without normalization. (b) Scatter plot of mutual information for every residue pair with normalization. Notice the different scales of plots between (a) and (b).

stop when a newly added edge does not contribute much to the score reduction of the graph, 2) stop when a designated number of edges have been added, and 3) stop when the likelihood of the model is within acceptable bounds. We use the first criterion in our model. With naive implementation of Algorithm 1 the running time is $O(dn^2)$ where n is the number of residues in a family and d is the maximum degree of nodes in the prior. By caching and preprocessing the complexity can be reduced to $O(dn)$.

3.2 Hierarchical Latent Class Models

A *latent class model* (LCM) is a hidden-variable model which consists of a hidden (class) variable and a set of observed variables [13]. The semantics of an LCM are that the observed variables are independent given a value of the class variable. Let u and v be two observed variables. The latent class model of u and v introduces a latent variable z, so that

$$P(u,v) = \sum_k P(z=k)P(u|z=k)P(v|z=k) \qquad (7)$$

When the number of observed variables increases, the LCM model performs poorly due to the strong assumption of local independence. To improve the model, Zhang et al. proposed a richer, tree-structured, latent variable model [20]. Our hierarchical model is a restricted case of the model proposed by Zhang et al. We propose a two-layered binary hierarchical latent class model where the lower layer consists all the observed variables and the upper layer consists of hidden class variables. In our problem setting, observed variables correspond to residues and the hidden class variables take values from all possible permutations of pairwise amino acid classes. Figure 6 illustrates a hypothetical hierarchical latent class model.

Let Z be the set of all hidden variables and V be the set of observed variables. The joint probability distribution of the model is as follows:

$$P(Z)\prod_{i=1}^{n} P(v_i|\mathrm{Pa}(v_i)) \qquad (8)$$

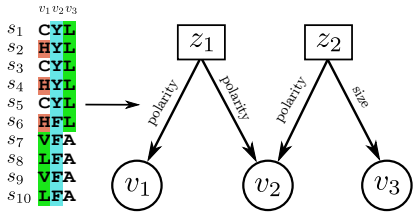


Figure 6: A hypothetical residue coupling in terms of amino acid classes using a two-layered Bayesian network.

where $\mathrm{Pa}(v_i)$ denotes the set of parents of v_i.

3.2.1 Learning a HLCM

We learn this model in a greedy fashion as before. We define the following scoring function:

$$\mathrm{S}_{\mathrm{HLCM}}(G=(\{V,Z\},E)) = \sum_{v_i \in V} \sum_{v_j \notin \mathrm{Pa}(v_i)} I_e(v_i, v_j|\mathrm{Pa}(v_i)) \qquad (9)$$

where $\mathrm{Pa}(v_i)$ is the set neighbors of v_i. When we condition on the parent nodes, we use a 35% support threshold for the sequences. This support threshold is required in order to ensure sufficient fidelity to the original MSA and allow for evolutionary exploration. From extensive experiments with this parameter (data not shown), we found that while there is some variation in the edges with changes of this parameter from 15% to 60%, many of the best edges are retained when support threshold is 35%. Moreover, the model has less number of couplings when support threshold is 35% which is an indication in the reduction of the overfitting effect. Besides, we use a parameter *minsupport* which is set to 2; minsupport is used to avoid class conservation between sequences. The value of minsupport for two residue positions is the number of class-values combinations for which the number of sequences in each subset is greater

Algorithm 2 HLCM($\mathcal{S}, P$)

Input: $\mathcal{S}$ (multiple sequence alignment), P (possible pairs of residues)
Output: G (a graph that captures couplings in $\mathcal{S}$)

1. $V = \{v_1, v_2, \ldots, v_n\}$
2. $Z \leftarrow \phi$ ▷ set of hidden nodes
3. $E \leftarrow \phi$
4. $T \leftarrow \phi$ ▷ tabu list of residue pairs
5. $s \leftarrow \mathrm{S}_{\mathrm{HLCM}}(G = (V, E))$
6. **for all** $e = (v_i, v_j) \in P$ **do**
7. $E' \leftarrow \{(h_e, v_i), (h_e, v_j)\}$ ▷ h_e is a hidden class between v_i and v_j
8. $C_e \leftarrow s - \mathrm{S}_{\mathrm{HLCM}}(G = (\{V, \{h_{ij}\}\}, E'))$
9. **while** stopping criterion is not satisfied **do**
10. $e \leftarrow \arg\max_{e \in P-T} C_e$
11. **if** e is significant for coupling **then**
12. $E \leftarrow E \cup \{(h_e, v_i), (h_e, v_j)\}$
13. $Z \leftarrow Z \cup \{h_e\}$
14. $T \leftarrow T \cup \{e\}$
15. label two edges of h_e based on the score
16. $s \leftarrow s - C_e$
17. **for all** $e' = (v_k, v_l) \in P - T$ s.t e and e' share a vertex **do**
18. $E'' \leftarrow \{(h_{e'}, v_k), (h_{e'}, v_l)\}$
19. $C_{e'} \leftarrow s - \mathrm{S}_{\mathrm{HLCM}}(G = (\{V, Z\}, E \cup E''))$
20. **return** $G = (V, E)$

than the support threshold. When minsupport is 1 for two residue positions, we consider that a class conservation has occurred in these residue positions. The algorithm chooses a pair of residues for which introducing a hidden variable reduces the current network score the most. We then add the hidden variable if it is statistically significant. Algorithm 2 gives the formal details for learning HLCMs. We can employ various stopping criteria: 1) stop when a newly added hidden node does not contribute much to the score reduction of the graph, 2) stop when a designated number of hidden nodes have been added, and 3) stop when the likelihood of the model is within acceptable bounds. We use the first criterion in our model.

3.3 Statistical significance

While learning the edges, hidden nodes or factors of the above graphical models, we assess the significance of each coupling imputed. In both algorithms, we perform a statistical significance test on potential pairs of residues before adding an edge or hidden variable to the graph. To compute the significance of the edge, we use p-values to assess the probability that the null hypothesis is true. In this case, the null hypothesis is that two residues are truly independent rather than coupled. We use the χ-squared test on potential edges. If p-value is less than a certain threshold p_θ, we add the edge to the graph. In our experiment, we use $p_\theta = 0.005$.

3.4 Classification

The graphical models learned by algorithm are useful for annotating protein sequences of unknown class membership with functional classes. To demonstrate the classification methodology, we consider HLCM as an example. We adopt Eq. 10 to estimate the parameters of a residue in the HLCM model. The reason for using this estimator is that the MSA may not sufficiently represent every possible amino acid value for each residue position. Therefore, we must consider the possibility that an amino acid value may not occur in the MSA but still be a member of the family. In Eq. 10, $|\mathcal{S}|$ is number of sequences in the MSA and α is a parameter that weights the importance of missing data. We employ a value of .1 for α but tests (data not shown) indicate that results are similar for values in $[0.1, 0.3]$.

$$P(v = a) = \frac{freq(v = a) + \frac{\alpha|\mathcal{S}|}{21}}{|\mathcal{S}|(1 + \alpha)} \tag{10}$$

Given two different graphical models, G_{C_1} and G_{C_2}, say for two different classes, we can classify a new sequence s into either functional class C_1 or C_2 by computing the log likelihood ratio LLR:

$$LLR = \log \frac{\mathcal{L}_{G_{C_1}}}{\mathcal{L}_{G_{C_2}}} \tag{11}$$

If LLR is greater than 0 then, then we classify s to the class C_1; otherwise, we classify it to the class C_2.

4. EXPERIMENTS

In this section, we describe the datasets that we use to evaluate our model and show results that reflect the capabilities of our models. We seek to answer the following questions using our evaluation:

1. How do our graphical models fare compared to other methods? Do our learned models capture important covariation in the protein family? (Section 4.2)
2. Do the learned graphical models have discriminatory power to classify new protein sequences? (Section 4.3)
3. What forms of amino acid class combinations are prevalent in the couplings underlying a family? (Section 4.4)

4.1 Datasets

4.1.1 Nickel receptor protein family

The Nickel receptor protein family (NikR) consists of repressor proteins that bind nickel and recognize a specific DNA sequence when nickel is present, thereby repressing gene transcription. In the *E. coli* bacterium, nickel ions are necessary for the catalytic activity of metalloprotein enzymes under anaerobic conditions; NikABCDE permease acquires Ni^{2+} ions for the bacterium [2]. NikR is one of the two nickel-responsive repressors which control the excessive accumulation of Ni^{2+} ions by repressing the expression of NikABCDE. When Ni^{2+} binds to NikR, it undergoes conformational changes for binding to DNA at the NikABCDE operator region and represses NikABCDE [2].

NikR is a homotetramer consisting of two distinct domains [16]. The N-terminal domain of each chain has 50 amino acids and constitutes a ribbon-helix-helix (RHH) domains that contact the DNA. The C-terminal of each chain consisting of 83 amino acids form a tetramer composed of four ACT domains that together contain the high-affinity Ni^{2+} binding sites [2]. Figure 7 shows a representative NikR structure determined by X-ray crystallography [2].

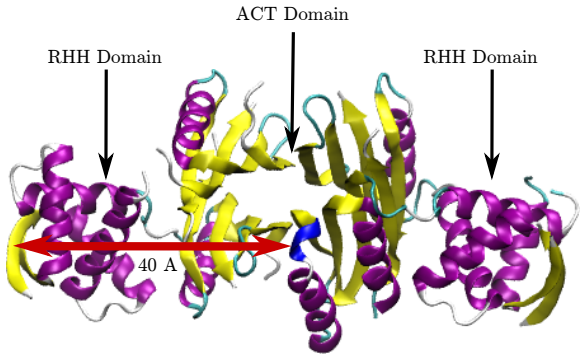


Figure 7: A rendering of NikR protein (PDB id 1Q5V) showing two domains: ACT domain (Nickel binding site) and RHH domain (DNA binding site). The distance between these two domains is 40ÅṪhe molecular image is generated using VMD 1.9 [5].

We organized an MSA of the NikR family that has 82 sequences which are used to study allosteric communication in NikR [2]. Each sequence has 204 residues. For a structural prior, we use Apo-NikR (pdb id 1Q5V) as a representative member of the NikR family and calculate prior edges from its contact map. Residue pairs within 7Å of each other are considered to be in contact which gives us 734 edges as a prior. We use this prior for the analysis to ensure that all identified relationships have direct mechanistic explanations.

4.1.2 G-protein coupled receptors

G-protein coupled receptors (GPCRs; see Fig. 8) represent a class of large and diverse protein family and provide an explicit demonstration of allosteric communication. The primary function of this proteins is to transduce extracellular stimuli into intracellular signals [6]. GPCRs are a primary target for drug discovery.

We obtained an MSA of 940 GPCR sequences used in the statistical coupling analysis by Ranganathan and colleagues [8]. Each sequence has 348 residues. GPCRs can be organized into five major classes, labeled A through E. The MSA that we obtained is from class A; using the GPCRDB [4], we annotate each sequence with functional class information according to the type of ligand the sequence binds to. The three largest functional classes—Amine, Peptide, and Rhodopsin—have more than 100 sequences. There are 12 other functional classes having less than 45 sequences. There are 66 orphan sequences which do not belong to any family. For prior couplings, we constructed a contact graph network from the 3D structure of a prominent GPCR member, viz. bovine rhodopsin (pdb id 1GZM). We identify 3109 edges as coupling priors using a pairwise distance threshold of 7Å.

4.2 Evaluation of couplings

We evaluate four methods on the NikR and GPCR datasets: the traditional GMRC method proposed by Thomas et al. [10, 9]; GMRC-Inf from this paper; GMRC-Inf* (a variant of GMRC-Inf) where the inflated alignment uses only class-based information; and HLCM. We consider three physicochemical properties—polarity, hydrophobicity, and size—of amino acids as classes. Although GMRC discovers couplings in terms of amino acids, we compare our methods with GMRC with respect to the number of discovered important

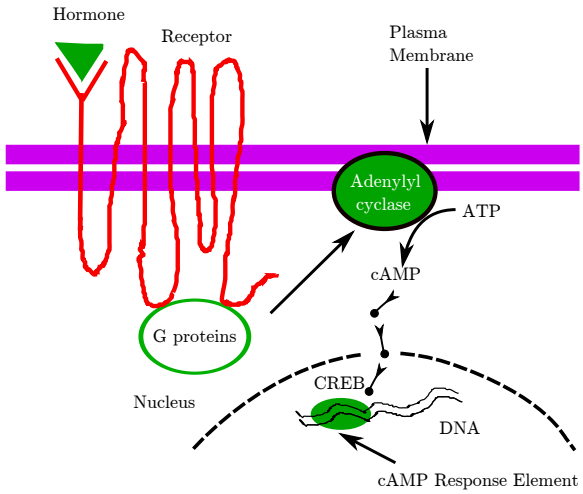


Figure 8: A cartoon describing GPCR functionality. Figure redrawn from [12].

Table 1: Important residues for allosteric activity in NikR collected from [2]. Residues are mapped from indices with respect to Apo Nikr (PDB id 1Q5V) to the indices of NikR MSA column. Important residues having conservation greater than 90% are not shown.

Residue	Sequence Conservation	Significance
3	0.83	Specific_DNA_binding
5	0.62	Specific_DNA_binding
7	0.81	Specific_DNA_binding
9	0.58	Unknown
22	0.45	Unknown
27	0.64	Nonspecific_DNA_contact
30	0.81	Low-affinity_Metal_Site
33	0.87	Nonspecific_DNA_contact
34	0.71	Low-affinity_Metal_Site
37	0.85	Unknown
42	0.41	Unknown
58	0.60	Ni2+_site_H-bond_network
60	0.86	Close_proximity_to_Ni2+_site
62	0.83	Close_proximity_to_Ni2+_site
64	0.38	Nonspecific_DNA_contact
65	0.52	Nonspecific_DNA_contact
69	0.51	Unknown
75	0.74	Ni2+_site_H-bond_network
109	0.49	Unknown
114	0.47	Unknown
116	0.39	Low-affinity_Metal_Site
118	0.45	Low-affinity_Metal_Site
119	0.62	Nonspecific_DNA_contact
121	0.82	Low-affinity_Metal_Site

Table 2: Comparisons of methods for various feature on NikR dataset.

Features	GMRC	GMRC-INF	GMRC-INF*	HLCM
Support Threshold (%)	15	15	35	35
Num of couplings	80	65	26	51
Num of important residues (out of 24)	15	11	9	15
Unique residues in the network	81	61	38	74
Num of components	11	6	13	23

residues (we desire to investigate whether our models can recapitulate important residues identified by previous methods). In Table 1, we list 24 important residues for NikR activity from [2] which are not conserved. (We exclude seven important residues for NikR which have a conservation of more than 90%.) Table 2 gives comparisons between methods for these two datasets.

Likewise, we identify 47 important residues for the GPCR family from [8]. The support threshold for GMRC and GMRC-INF is set to 15%; the support threshold and min-support for HLCMis set to 35% and 2 respectively. (To be more confident about the quality of the model, the support for HLCMis set to a higher value.)

Bradley et al. [2] identify four residues (Res 9, Res 37, Res 62, and Res 118) as highly connected "hubs". In our models, Res 9 and Res 118 are present, but Res 37 and Res 62 are not present since these residues are highly conserved. Important residues discovered by four methods are shown in Table 3. We see that GMRC-INF and GMRC-INF* are progressively more strict than GMRC in the number of important residues discovered but GMRC-INF* has a greater ratio of important residues discovered to the total residues in the network. HLCM provides as good performance as the GMRC method in terms of the important residues but compacts them into a smaller set of couplings.

Table 3: Important residues discovered by HLCM, GMRC-INF, GMRC-INF*, and GMRC in NikR.

Method	Important Residues
HLCM	3, 7, 9, 27, 30, 34, 42, 60, 97, 109, 114, 116, 118, 119, 121
GMRC-INF	27, 30, 33, 34, 37, 58, 60, 97, 116, 118, 121
GMRC-INF*	3, 5, 27, 33, 37, 42, 60, 116, 121
GMRC	3, 7, 9, 27, 30, 33, 34, 37, 58, 60, 97, 116, 118, 119, 121

4.3 Classification performance

Although our goal is to represent amino acid class-based residues couplings in a formal probabilistic model, we demonstrate that our models can also classify protein sequences. We use the GPCR dataset to assess the classification power of our models. The GPCR datasets has 16 subclasses with, as stated earlier, the three major subclasses being amine, peptide, and rhodopsin. We performed a five-fold cross-validation test for these three major classes. A comparison between our HLCM model and the vanilla GMRC is given in Table 4. We see an improved performance for the Amine subclass and a slightly decreased performance for the Rhodopsin subclass.

Table 4: Classification of GPCR subclasses.

Functional Class	Total Sequence	Accuracy (%)	
		GMRC	HLCM
Amine	196	99.5	100
Peptide	333	100	100
Rhodopsin	143	98.6	95.8

Recall that there are 66 orphan sequences in GPCR family which are not assigned to any functional class. We apply our model to classify these orphan sequences to any of the three major classes: Amine, Peptide, and Rhodopsin. Toward this end, we build models for the three classes using HLCM method by considering all of the sequences. Of the 66 sequences, 3 are classified to Amine and the rest are classified to the Peptide class. This result is the same as the GMRC result reported in [10].

4.4 Finding coupling types

We determine the frequency of each class-coupling type for the various models on the NikR dataset. Histograms are shown in Figure 9. We see that there are a significant number of class-based residue coupling relationships discovered, although in the case of GMRC-INF, there are many value-based couplings as well (as expected). Many of the couplings discovered by GMRC-INF* and HLCM have polarity as one of the properties, but there are interesting differences as well: HLCM identifies a significant number of P-S couplings whereas GMRC-INF* finds P-P, P-H, and S-S couplings.

5. DISCUSSION

Our results on the NikR dataset demonstrate that employing amino acid types is useful for learning couplings and the underlying properties of those couplings. This approach provides us with a way to build an expressive model for residue couplings. We have shown that our extended graphical model is more powerful than the previous graphical model approach of Thomas et al. [10].

Our use of conditional mutual information as a correlation measure is subject to different biases [14]. Removing possible biases is a direction for future work. A more unifying probabilistic approach for residue couplings would be a factor graph representation since it can capture couplings among more than two residues. A *factor graph* is a bipartite graph that represents how a joint probability distribution of several variables factors into a product of local probability distributions [1]. Let $G = (\{F, V\}, E)$ be a factor graph, where $F = \{f_1, f_2, \ldots, f_m\}$ is a set of factor nodes and $V = \{v_1, \ldots, v_n\}$ is a set of observed variables. A *scope* of a factor f_i is set a set of observed variables. Each factor

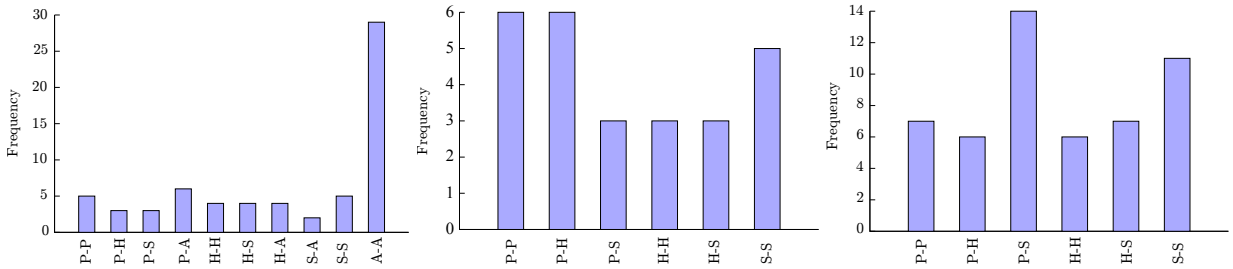


Figure 9: Histograms for class-coupling types on the NikR dataset using three methods: (a) GMRC-Inf (b) GMRC-Inf*, and (c) HLCM.

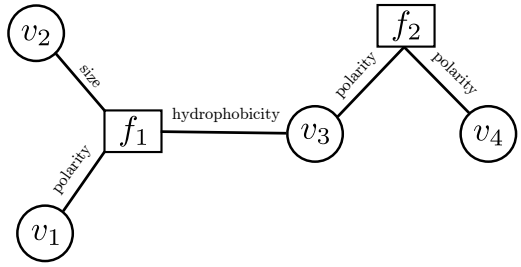


Figure 10: A hypothetical residue coupling in terms of amino acid classes using a factor graph model.

f_i with scope C is a mapping from $\text{Val}(C)$ to $\mathbb{R}^+$. The joint probability distribution of V is as follows:

$$P(v_1, v_2, \ldots, v_n) = \frac{1}{Z} \prod_{j=1}^{m} f_j(C_j) \tag{12}$$

where C_j is the scope of the factor f_j and the normalizing constant Z is the partition function. Figure 10 illustrates a hypothetical residue coupling network for four residues with two factors. Observe how such a model can capture couplings involving more than two residues.

While there are polynomial time algorithm for learning factor graphs from polynomial samples [1], such methods require a canonical parameterization which constraints the applicability of factor graphs to learn couplings from an MSA. Canonical parameterizations are defined relative to an arbitrary but fixed set of assignments to the random variable, and it is hard to define such a 'default sequence' for an MSA. Hence, newer algorithms need to be developed.

Acknowledgement

This work is funded by NSF grant IIS-0905313. We convey our thanks to Debprakash Patnaik for his suggestions.

6. REFERENCES

[1] Abbeel et al. Learning factor graphs in polynomial time and sample complexity. *Journal of Machine Learning Research*, 7:1743–1788, 2006.

[2] Bradley et al. Molecular dynamics simulation of the Escherichia coli NikR protein: equilibrium conformational fluctuations reveal interdomain allosteric communication pathways. *JMB*, 378(5):1155–1173, May 2008.

[3] Durbin et al. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge University Press, 1998.

[4] Horn et al. Collecting and harvesting biological data: The GPCRDB and NucleaRDB databases. *Nucleic Acids Research*, 29(1):346–349, 2001.

[5] Humphrey et al. VMD – Visual Molecular Dynamics. *Journal of Molecular Graphics*, 14:33–38, 1996.

[6] Kroeze et al. G-protein-coupled receptors at a glance. *Journal of Cell Science*, 116:4867–4869, 2003.

[7] Lichtarge et al. An evolutionary trace method defines binding surfaces common to protein families. *Journal of Molecular Biology*, 257:342–358, 1996.

[8] Suel et al. Evolutionary conserved networks of residues mediate allosteric communication in proteins. *Nature Structural Biology*, 10(1):59–69, Jan 2003.

[9] Thomas et al. Graphical models of residue coupling in protein families. In *5th ACM SIGKDD Workshop on Data Mining in Bioinformatics (BIOKDD)*, pages 1–9, 2005.

[10] Thomas et al. Graphical models of residue coupling in protein families. *IEEE/ACM TCBB*, 5(2):183–97, 2007.

[11] A.A. Fodor and R.W. Aldrich. Influence of conservation on calculations of amino acid covariance in multiple sequence alignments. *Proteins: Structure, Function, and Bioinformatics*, 56:211–221, 2004.

[12] John W. Kimball. Cell signaling, June 2006.

[13] P. F. Lazarsfeld and N. W. Henry. *Latent Structure Analysis*. Boston, Mass.: Houghton Mifflin., 1968.

[14] Little. Identification of coevolving residues and coevolution potentials emphasizing structure, bond formation and catalytic coordination in protein evolution. *PloS One*, 4(3):e4762, January 2009.

[15] S.W. Lockless and R. Ranganathan. Evolutionarily conserved pathways of energetic connectivity in protein families. *Science*, 286(5438):295–299, Oct 1999.

[16] Eric R. Schreiter, Michael D. Sintchak, Yayi Guo, Peter T. Chivers, Robert T. Sauer, and Catherine L Drennan. Crystal structure of the nickel-responsive transcription factor nikr. *Nature Structural and Molecular Biology*, 10:794–799, September 2003.

[17] William S J Valdar. Scoring residue conservation. *Proteins*, 48(2):227–41, August 2002.

[18] Y. Y Yao. Information-theoretic measures for knowledge discovery and data mining. *Entropy Measures, Maximum Entropy Principle and Emerging Applications*, pages 115–136, 2003.

[19] Chen-Hsiang Yeang and David Haussler. Detecting coevolution in and among protein domains. *PLoS Computational Biology*, 3(11):13, 2007.

[20] Nevin L. Zhang and Tomás Kocka. Efficient learning of hierarchical latent class models. *IEEE International Conference on Tools with Artificial Intelligence*, 0:585–593, 2004.

Analyze Influenza Virus Sequences Using Binary Encoding Approach

HamChing Lam
Dept. of Computer Science and Engineering
University of Minnesota
Minneapolis, Minnesota 55455, USA
hamching@cs.umn.edu

Daniel Boley
Dept. of Computer Science and Engineering
University of Minnesota
Minneapolis, Minnesota 55455, USA
boley@cs.umn.edu

ABSTRACT

Capturing mutation patterns of each individual influenza virus sequence is often challenging; in this paper, we demonstrated that using a binary encoding scheme coupled with dimension reduction technique, we were able to capture the intrinsic mutation pattern of the virus. Our approach looks at the variance between sequences instead of the commonly used p-distance or Hamming distance. We first convert the influenza genetic sequence to a binary string and then apply Principal Component Analysis (PCA) to the converted sequence. PCA also provides a prediction capability for detecting reassortant virus by using data projection technique. Due to the sparsity of the binary string, we were able to analyze large volume of influenza sequence data in a very short time. For protein sequences, our scheme also allows the incorporation of biophysical properties of each amino acid. Here, we present various results from analyzing influenza nucleotide, protein and genome sequences using the proposed approach. With the Next-Generation Sequencing (NGS) promises of sequencing DNA at unprecedented speed and production of massive quantity of data, it is imperative that new technique needs to be developed to provide quick and reliable analysis of any sequence data. Here, we believe our approach can be used at the upstream stage of sequence data analysis pipeline to gain insight as to which direction should be continued on in analyzing the available data.

Keywords

Influenza virus, Evolution, Binary Encoding, Principal Component Analysis

1. INTRODUCTION

The influenza A virus is a negative stranded RNA virus with eight gene segments that code for 10 proteins in its genome. It is categorized by the serology and genetics of its two surface glycoproteins hemagglutinin (HA) and neuraminidase (NA). The virus is capable of infecting about twenty five percent of the worldwide human population each year [13]. 16 HA antigenetically distinct subtypes have been isolated from mammalian and avian hosts, with the H3N2 being the most widespread and dominant circulating strain in the human population [9]. Selective pressure exists for the virus to generate immunological escape variants that are antigenetically different and to diversity because immunized hosts are resistant to infection with influenza they have been exposed to for several years [10].

Large effort and vast amount of sequence data have been used together to piece together the evolutionary history of Influenza viruses. The influenza evolutionary tree itself is one of the most popular and powerful tools we have in understanding the evolution of the virus. The continuous evolving of the virus makes it challenging to get a global picture of how all the viruses are inter-related. With an evolutionary tree, we can: (1) build a more complete understanding of how and where virus evolved that would help explain how certain changes ended up in certain clade along the evolutionary tree. (2) enable us to more easily decipher what's in the virus samples we already have and to make prediction on what antigenic property we'll find in newly isolated viruses. Evolution turns out to be a good structural framework for understanding influenza virus evolution dynamic [7, 9]. However, with the large number of sequence data continuously being deposited to the influenza database, the data often appears to be clouded, unclear, and even redundant. An approach that can quickly provide an overview of the virus evolution under these challenges is most valuable to influenza analysis.

Our aim in this paper is to present an alternative sequence representation method that is capable of capturing the intrinsic patterns of mutation of the virus and extract these patterns through a dimension reduction technique. To show the utility and flexibility of the encoding scheme, we performed influenza sequence analysis to expose avian-host to human-host cross-overs using both nucleotide, protein and genome sequences downloaded from NCBI Influenza database [1].

2. RESULTS

In this section, we present various results from applying our encoding scheme to influenza genetic sequences using Principal Component Analysis as the processing algorithm. We illustrate the evolution trajectory of H3N2 virus obtained from using nucleotide sequences. We then provide a global view of all the subtypes of influenza viruses based on their HA surface protein. Next, we give result from in-

BIOKDD 2011, August 2011, San Diego, CA, USA

tegrating biophysical information of each amino acid to enhance the distinguishing feature of each virus sequence. We tested this approach on H3 and H5 subtype viruses. Last, we present results of the predictive power of PCA based on our encoding scheme by detecting reassortant virus using complete virus genome sequence.

2.1 H3N2 evolution trajectory

Multi-Dimensional Scaling (MDS) was used as a dimension reduction technique by [14] to project genetic and antigenic influenza data to visualize the relationship between strains on a two dimensional plane. MDS must first compute the pairwise distance between strains and then proceed to optimize an objective function to preserve the pairwise distance between strains as best as possible. MDS is often used to provide visualization of influenza clusters to gain a first hand understanding of their evolution trajectory. On the other hand, the same objective can be achieved by using PCA where strains' pairwise distance computations are not needed. To achieve this objective, PCA uses the covariance between each strain and find the new and reduced dimensions to visualize the data (please see Materials and Methods section for more detail on PCA). The results from using the proposed encoding scheme on nucleotide sequences show that the evolution trajectory of the H3N2 virus produced from Principal Component Analysis (PCA) is the same as that produced from Multi-Dimensional Scaling algorithm when the Euclidean metric was used for pairwise distance calculation between strains. In the PCA case, two dimensions are usually sufficient to explain most of the variability of the data. Here, in figure 1 top plot, we show that it produced the same H3N2 evolution trajectory as MDS using H3N2 nucleotide sequences. We colored the vaccine strains in red in top figure and also listed them in table 1. Each vaccine strain follows nicely in a chronological manner in the curved pattern (from lower left to lower right) among all other H3N2 strains. This trajectory indicates that H3N2 virus is evolving away from its earliest 1968 isolated strain.

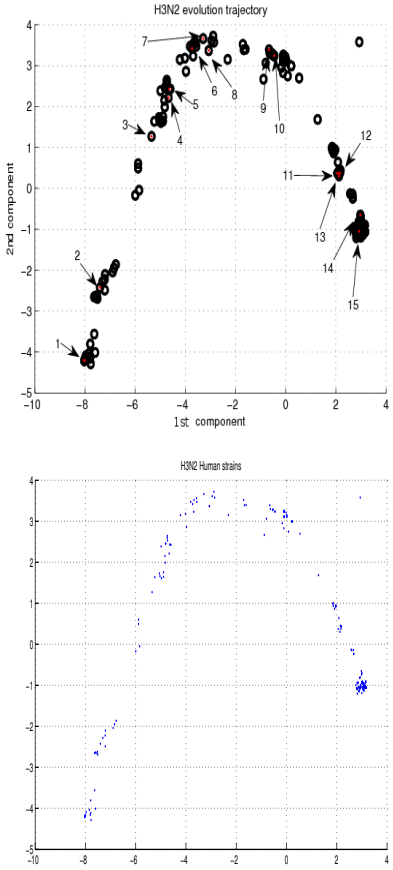


Figure 1: H3N2 evolution trajectory using PCA (top plot) and MDS (bottom plot)

Table 1: Vaccine strains shown in red in figure 1 (top).

Number	Vaccine strain
1	A/Aichi/1968
2	A/Port Chalmers/1/1973
3	A/Philippines/2/1982
4	A/leningrad/360/1986
5	A/Shanghai/11/1987
6	A/Beijing/353/1989
7	A/Shangdong/9/1993
8	A/Johannesburg/33/1994
9	A/Sydney/5/1997
10	A/Moscow/10/1999
11	A/Fujian/411/2002
12	A/California/7/2004
13	A/Wisconsin/67/2005
14	A/Brisbane/10/2007
15	A/Perth/16/2009

2.2 Incorporating amino acid biophysical information

The proposed encoding scheme with the inclusion of amino acids' biophysical properties leads to substantially better results in distinguishing different subtype when protein sequences are used. The biophysical property we have used in this study is the hydrophobicity property of amino acids. Ray [12] carried out a study to determine the most suitable biophysical properties to use with unsupervised classifiers [12] and found that three properties: Volume, Hydrophobicity, and Isoelectric property are best suited for classification purposes. In our study, we have tried all three of the said properties and found that hydrophobicity is best suited for influenza sequences. We demonstrate this result by applying our coding scheme combined with hydrophobicity values (H-value) on H3 and H5 subtypes nucleotide sequences. We obtained the hydrophobicity values for all the amino acids published from the study conducted by Ray and Kepler [12]. After appending each H-value to the binary string of each amino acid and converted all the protein H3 and H5 sequences into binary strings, PCA was used to provide visualization (figure 2 and 3) between the two subtypes on two dimensional plane. For comparison purpose, we produced a projection of H3 and H5 sequence without using the H-value, as shown in figure 2. Although we see data separation in both cases, the projection result with H-

value applied clearly explained more variance (at 70 percent) than the one without (at upper 30 percent). The separation between H3 and H5 also has become more pronounced with less overlapping strains from each subtype.

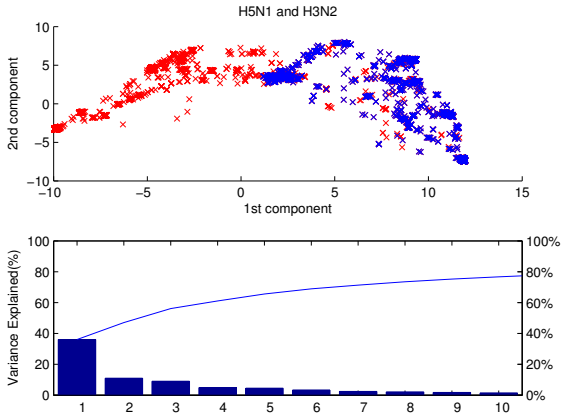


Figure 2: PCA projection of H3 and H5 protein sequences without applying hydrophobicity information.

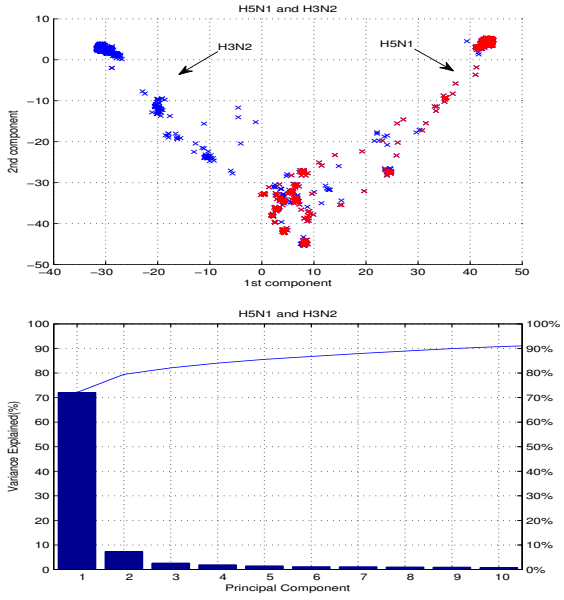


Figure 3: PCA projection of H3 and H5 protein sequences with hydrophobicity information incorporated.

2.3 Complete view of all subtypes of influenza viruses

The diversity and distribution of the influenza virus has been studied by [3, 7] by building a panorama of phylogenetic trees. Here, we decided to apply our encoding scheme to all 16 subtypes of the influenza virus hemagglutinin nucleotide sequences totalling 16993 to produce a two dimensional whole view of all subtypes. After converting the hemagglutinin nucleotide sequences to binary strings, we used PCA to project all the subtypes (H1 to H16), obtaining a global view of the virus. From figure 4, we see a tripod shape with H1N1, H3N2, and H5 each occupying a tripod leg (each of the green dots designates the earliest of each isolate subtype). All the other subtypes remain in the center of the tripod, showing very little change. This indicates that the three subtypes H1N1, H3N2, and H5 are evolving faster than the other subtypes. On the H3N2 leg, the black dots represent H3N2 vaccine strains used from 1968 to 2007. Among the 16 subtypes, H13 and H16 are very close to each other. This is in agreement with [7]. On the other hand, H2, H4, H9, H10, and H15 appear to be close to each other. Subtypes H2 and H9 are are very close to each other, but phylogenetic analysis indicates that these two subtypes were derived from different lineages. One explanation is that there is small synonymous differences (mutation at nucleotide level but does not change the encoded amino acid) exist between these two subtypes based on sequence level analysis. The lineage different can come from viruses evolving within the same host type (e.g. Human H1N1 and Human H3N2) but with different antigentic property for each lineage. Subtypes H4, H10, and H15 are clustered together in the plot, and phylogenetic analysis from [7] showed that they were derived from the same lineage.

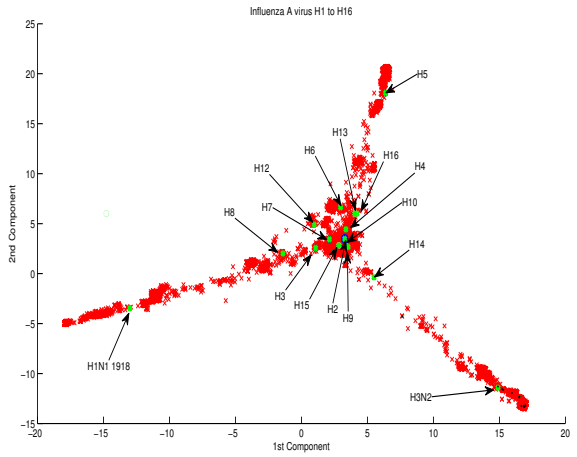


Figure 4: A complete view of all subtypes from 1918 to 2009. The three active evolving subtypes (H1, H3, and H5) are spread out to each tripod leg indicating their dominance in establishing their own lineage.

2.4 Detecting reassortants

Due to the segmented nature of influenza virus genome (8 individual segments of single stranded RNA that encodes 2 surface proteins and 8 internal proteins), reassortment between influenza viruses are common and can lead to the generation of novel strains of the virus [8]. In fact, pandemic strains have been found to carry gene segments originating from multiple hosts within their genome [11]. Here, we desire to test the predictive power of PCA coupled with our

binary encoding scheme with hydrophobicity information incorporated. We wish to identify influenza viruses originating from a single host but carrying gene segments belonging to multiple hosts. Our objective is to see whether PCA is able to identify virus's surface proteins that have gone through reassortment process. For the first test, we built an artificial reassortant virus (RV) dataset consisting of viruses with surface proteins HA and NA from avian hosts but internal proteins originating from a human host. Each RV genome is constructed by replacing the flu virus's (FV1) human-host HA and NA proteins with avian-host HA and NA proteins. We first pre-computed the principal components using flu virus (FV1) genome sequences whose genes all originated from human host only. Then we projected the reassortant virus (RV) genome sequences containing avian HA and NA genes onto these pre-computed FV1 principal components. From figure 5, we see that reassortant virus (RV) with proteins originating from human host (green) are closely "attached" to the human proteins (black) of the flu virus(FV1). On the other hand, its surface proteins (red dots) are clearly isolated from the surface proteins of human-host origin (blue dots).

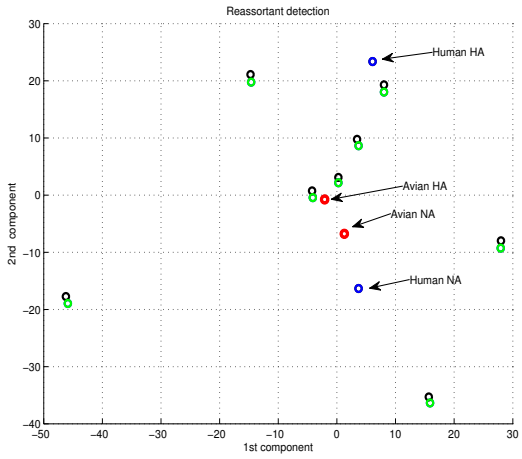


Figure 5: Plot of reassortant virus (RV) genome projected onto principal components computed using flu virus (FV1) genome of human origin. Each dot represents a gene sequence from the genome. RV genome are represented by green dots (internal proteins) and red dots (surface proteins). FV1 genome are represented by black dots (internal proteins) and blue dots (surface proteins).

We performed a second analysis test using a real reassortant virus H3N2 A/SW/CO/77 genome sequence identified in [5] to test the predictive power of our approach. We selected this isolate because its genetic characterization by [5] using phylogenetic trees indicated that
SW/CO/77 pig isolate's HA and NA proteins are closely related to the human influenza virus. In this second analysis, we conducted two tests: an experiment test and a control test. For the experiment test (result shown in figure 6), we first computed the principal components using field isolates of human origin flu viruses (see Materials and Methods for human virus genomes used) and then projected the A/Swine/CO/77 genome onto these precomputed principal components. We see that the HA and NA proteins of SW/CO/77 are closely "attached" to the human HA and NA counterparts, which suggests that these two surface proteins were originated from a human-host type virus during reassortment event.

For the control test (result shown in figure 7), we selected the H3N2 A/swine/Wisconsin/2/1970 swine virus as the control genome because SW/CO/77 was isolated in 1977. The reason for selecting a 1977 strain as a control is that the swine flu virus lineage at that time had not diverged into multiple lineages that carried gene segments with mixed host type [5]. This is also to assure that the control strain contains only gene segments from a single host type of swine origin. Based on phylogenetic analysis,
A/swine/Wisconsin/2/1970 does not contain foreign host type gene. In this control test, we precomputed the principal components using the control genome sequence and then projected the A/SW/CO/77 genome onto the first two components. Clearly, we can see that A/SW/CO/77 strain's HA and NA proteins (red dots) are clearly distantly apart from the swine origin counterparts (blue dots). From the results of these two reassortant detection tests, we can see that there is an unique feature or a signature pattern that represent each specific host type. With the right feature representation, PCA can quickly isolate and identify these type of attributes in the dataset.

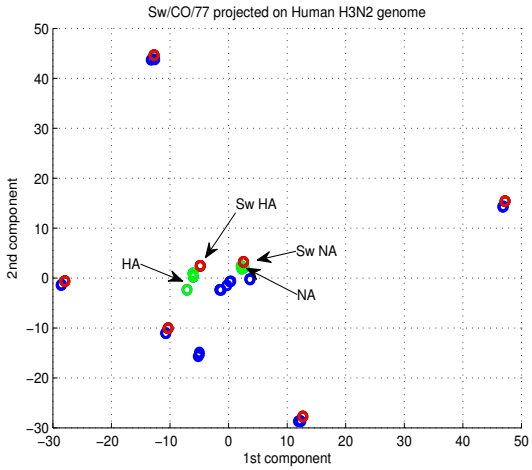


Figure 6: SW/CO/77 genome projected onto principal components computed using human origin flu viruses genomes. Green dots represent the Human HA and NA surface genes, red dots are the SW/CO/77 genes, and blue dots are the internal genes from human host genome.

3. DISCUSSION

In this paper, we have shown that using a flexible encoding scheme to convert influenza virus's nucleotide or protein sequence can enable us to automatically extract unique mutation pattern that carries evolution information of the virus. We have highlighted some analysis results using our approach that are important in the field of influenza se-

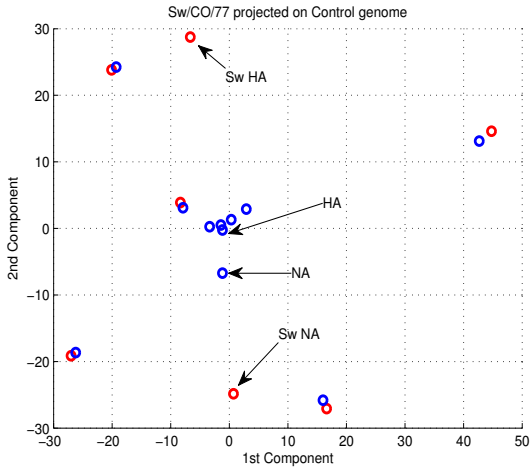


Figure 7: SW/CO/77 genome projected onto principal components computed using swine virus genome as control. Red dots are the SW/CO/77 genes and blue dots are the Control A/Swine/Wisconsin/ virus genes.

quence analysis. For example, a hidden difficulty when analyzing sequences from each flu season is that we do not know which strains in the data evolved from which other strains in the data, there is no indication or extra information showing the relationship between strains [2]. A pairwise comparison with this uncertainty can give results that could be biased because pairwise comparison implicitly assumes that one virus of the pair is the progenitor of the other [2].

The encoding approach proposed here still involves pairwise comparisons as part of the covariance calculation in PCA, but the encoding scheme introduced here allows PCA to automatically capture the locations of the mutation patterns. This is to say that the location of mutation along the sequence is more important than the pairwise distance information. We have demonstrated this with the plots of H3N2 evolution trajectory using PCA (figure 1). With PCA, we can quickly examine the variances associated with strains instead of relying on pairwise Hamming or P distance between strains. Usually only a small number K of components is needed to capture a large fraction of the total variance. The largest K variances are associated with the first K principal components, and there is usually a precipitous drop off in the variances after the K-th. Therefore, the most interesting dynamic of the data can be captured in the first K dimensions. With influenza virus sequences showing a very high genetic similarity characteristic within subtypes [15], this means that most of the sites carry redundant information and only a portion of the sequence contains vital genetic variation signal. This underlying phenomenon seems to be tailor made for PCA. We have shown that after converting the sequences to binary strings, PCA has no problem in capturing the intrinsic pattern of the virus sequence data. Although PCA and MDS yield approximately the same trajectory results, an advantage of using PCA is that PCA carries prediction capability. The prediction power of PCA comes from the fact that one can pre-compute a set of principal components with existing data (or training data) and then project a set of new data (test data) onto the precomputed principal components. This simple procedure can highlight the differences or similarity between the two data sets. We illustrated this by using it to detect reassortant viruses. To detect reassortant, we precomputed principal components from existing virus dataset that do not contain any mixed-host proteins within its genome. We then project new virus genome dataset suspected to contain reassortant proteins onto the precomputed principal components to detect any outliner or abnormally. Here, we have shown that PCA can quickly identify the mixing of human and avian genes in a virus genome. This aspect of prediction power from PCA is far more useful than using multidimensional scaling approach.

Feature representation schemes for amino acids usually employ a simple categorical representation where each amino acid is grouped together according to its pre-defined characteristic. Commonly found groups are charge group, polarity group and structure group. Each amino acid within each group is implicitly regarded as having equidistant from every other amino acid. Only the category of each amino acid is used, while the specifics for each individual amino acid is discarded. To overcome this distance bias introduced by the grouping strategy, we elected to directly incorporate each individual amino acid's property, including the individual identities. In our case, we have shown examples using the hydrophobicity property of amino acids as an extra information as it is one of the key properties relating to protein binding[4]. The extra information allows for a more accurate representation for each amino acid. Through using PCA, the results are encouraging as only two principal components were enough to capture the hidden pattern of the data.

With the Next-Generation Sequencing (NGS) promises of sequencing DNA at unprecedented speed and production of massive quantity of data, it is imperative that new technique needs to be developed to provide quick and reliable analysis of any sequence data. Here, we believe our approach can be used at the upstream stage of sequence data analysis pipeline to gain insight as to which direction should be continued on in analyzing the available data.

4. MATERIALS AND METHODS

4.1 Data

All influenza virus nucleotide, protein, and genome sequences used in this study were downloaded from NCBI Influenza Virus Database [1] as of February 2011.239 H3N2 HA1 nucleotide sequences were used for the trajectory analysis (accession numbers available upon request). H3N2 and H5N1 subtypes HA protein sequences totalling 5708 were used in the analysis presented in section 2.2. 16,993 hemagglutinin nucleotide sequences representing all subtypes of the flu virus were used to obtain the whole view plot of the virus. The majority of sequences were from H1 with 6632 sequences, H3 with 4071 sequences, and H5 with 3088 sequences. For reassortant detection, we selected human host flu genome sequences isolated from early 1970s to 1980s for the experiment test. This test set consists of genome sequences of strain Port Chalmers: A/Port Chalmers/1/1973, Udorn: A/udorn/1972, and Memphis: A/Memphis/15/1988 (accession numbers available upon request). For the control test, we selected
A/swine/Wisconsin/2/1970 genome from NCBI flu genome

database. Each influenza virus genome is named by its subtype, host, geographic location, strain number and year. The strain name refers to the virus genome which consists of 8 segments that codes for 10 proteins. In our study, we use the term genome to refer to a collection of 10 protein sequences that belong to one influenza strain. The term "sequence" is used to refer to a biological sequence of either nucleotide or amino acid of each individual protein within a genome.

4.2 Binary encoding

Transforming nucleotide or protein sequence to a feature vector that captures the mutation pattern is the key in determining the evolution trajectory of the influenza virus. Our approach is simple and has the ability to capture the mutation pattern of the virus. The feature vector is a string of zeros and ones that represents a biological sequence directly. This encoding is an embedding in high-dimensional Euclidean space with the property that the distance between each different "letter", or "nucleotide" or "amino acid" is the same. It also allows one to add almost arbitrary weightings to account for biological effects like hydrophobic vs. hydrophylic amino acids. Using the usaul ASCII representation encoding would introduce a biologically meaningless ordering to the individual letters. In addition, if protein sequences are used, our approach allows the incorporation of biophysical properties of each amino acid into each protein sequence which further enhances the differences between each amino acid. For nucleotide sequences, we encode Adenine (A) to "1000", Guanine (G) to "0100", Cytosine (C) to "0010" and Thymine (T) to "0001". Each nucleotide base is uniquely represented by a 4 digits binary string. For example, to encode a nucleotide sequence of "AGA" and another of "ACA", AGA is transformed to 0 0 0 1 0 1 0 0 0 0 0 1 and ACA is transformed to 0 0 0 1 0 0 1 0 0 0 0 1. When these two sequences are compared, the mutation in the second position is captured by the different between 0100 and 0010. This encoding scheme allows for direct capture of mutation information between sequences and facilitate direct subsequent computational analysis. For protein sequences, we convert each amino acid to a binary string of length twenty and each string is different by only one bit. For example, Alanine is coded as "1 0 0 0...0 0 0" and Cysteine is coded as "0100...000". In addition, the biophysical properties data of each amino acid can be directly append to the end of the twenty bits string. For example, the hydrophobicity value of Alanine is 1.8 and the binary string of Alanine becomes "1 0 0 0 ... 0 0 0 1.8" which further distinguishes the differences between each amino acid. Even though the length of the nucleotide sequence has been increased by a factor of 4 and protein sequence by a factor of 20, the sparsity of the representation does not incur a high computational overhead. In fact, we were able to analyze over five thousand protein sequences in a time of less than 15 minutes running on a moderately powerful (2.1 GHz with 4GB memory) desktop computer.

4.3 Principal Component Analysis

Principal Component Analysis (PCA) is used in all forms of analysis from bioinformatics to computer vision. It is a simple non-parametric method of extracting relevant information from unstructured data sets. The extraction can be viewed as dimensional reduction where a complex high dimension data set is reduced to a lower dimension in order to reveal hidden, simplified structure buried within the data. In order to find the best lower dimension to capture the structure of the high dimensional data, PCA proceeds by diagonalizing the covariance matrix of the data set, consistent with the goal to maximize the variance captured in the projected data onto the lower dimensions. One restriction is that PCA requires the directions of projection be orthogonal to each other and the variance associated with each direction be maximized. The orthogonal requirement makes PCA solvable with highly efficient linear algebra decomposition techniques. Here, we briefly introduce the working mechanism of PCA from a linear algebra perspective. Consider a data matrix $X_{m,n}$ with dimensions of m by n with m being the number of strains and n being the number of sites. Each row of X corresponds to a strain of virus and each column of X corresponds to a particular site. We first need to center the rows of the data matrix X (i.e. replace X with $X - \frac{1}{m} ee^T X$, where e is a column vector of all ones) and then obtain the covariance matrix C from X by $C = \frac{1}{(m-1)} XX^T$. C is a square symmetric $m \times m$ matrix whose diagonal entries are the variances of the individual strains across sites and the off-diagonal terms are the covariances between different strains. If one wishes to reduce the row dimensions, one can simply apply this entire computation to the transpose of the data matrix. The goal of PCA is to find a set of orthonormal axes that diagonalizes matrix C. The diagonalization of C is computed by finding its eigenvectors. Since C is symmetric and square, its eigenvectors are the orthonormal principal directions, and its eigenvalues correspond to the variances of the data along those principal directions. The eigenvectors of C are now the new basis for the data X. The projection of the data matrix X onto this new basis gives the alternative "PCA view" of the data with mean zero and variance maximized along each principal component direction. A quick decomposition technique to obtain the orthonormal basis is using the Singular Value Decomposition (SVD) [6]. One can center the matrix, calculate the C matrix, and then applying SVD to C. SVD of C gives $C = U\Sigma V^T$ where the matrix V contains the orthonormal basis we sought. We can then project the data to these orthonormal basis with $X * V$; The matrix Σ is a diagonal matrix that contains the eigenvalues of C which are the variances of the orthonormal basis/principal components.

For the H3N2 evolution trajectory analysis, the H3N2 HA nucleotide sequences of the same length were converted to binary strings which yielded a data matrix that can be directly used with PCA algorithm. The first two principal components corresponding to the two largest eigenvalues were then plotted to obtain the trajectory. In section 2.2, H3 and H5 HA protein sequences of the same length were used and converted to binary strings. In section 2.3, all HA nucleotide sequences with the same length were used and converted to binary strings. For both sections 2.2 and 2.3, PCA were then directly applied to the converted binary strings and the first two principal components were selected for plotting and visualization purposes. In section 2.4, influenza genome (consisted of 10 protein sequences) was converted to binary strings with H-value incorporated. PCA algorithm was then used to find the first two principal components for the training data set (the FV1 genome, human flu virus genmoe, and A/Swine/Wisconsin/72 genome). The projection of testing dataset (RV genome, and A/Swine/CO/77 genome) onto

the two principal components were done as outlined above. We perform all the computation using Matlab 7.6 version software. The PCA results were generated by the princomp function from Matlab's Stats toolbox.

5. REFERENCES

[1] Y. Bao, P. Bolotov, D. Dernovoy, B. Kiryutin, L. Zaslavsky, T. Tatusova, J. Ostell, and D. Lipman. The influenza virus resource at the national center for biotechnology information. *J Virol*, 82(2):596–601, Jan 2008.

[2] M. F. Boni. Vaccination and antigenic drift in influenza. *Vaccine*, 26 Suppl 3:C8–14, Jul 2008.

[3] J.-M. Chen, Y.-X. Sun, J.-W. Chen, S. Liu, J.-M. Yu, C.-J. Shen, X.-D. Sun, and D. Peng. Panorama phylogenetic diversity and distribution of type a influenza viruses based on their six internal gene sequences. *Virol J*, 6:137, 2009.

[4] T. Hopp. Computer prediction of protein surface features and antigenic determinants. *Prog Clin Biol Res*, 172B:367–377, 1985.

[5] A. I. Karasin, M. M. Schutten, L. A. Cooper, C. B. Smith, K. Subbarao, G. A. Anderson, S. Carman, and C. W. Olsen. Genetic characterization of h3n2 influenza viruses isolated from pigs in north america, 1977-1999: evidence for wholly human and reassortant virus genotypes. *Virus Res*, 68(1):71–85, Jun 2000.

[6] V. Klema and A. Laub. The singular value decomposition: Its computation and some applications. *IEEE Transactions on Automatic Control*, 25(2):164–176, 1980.

[7] S. Liu, K. Ji, J. Chen, D. Tai, W. Jiang, G. Hou, J. Chen, J. Li, and B. Huang. Panorama phylogenetic diversity and distribution of type a influenza virus. *PLoS One*, 4(3):e5022, 2009.

[8] C. J. Luke and K. Subbarao. Vaccines for pandemic influenza. *Emerg Infect Dis*, 12(1):66–72, Jan 2006.

[9] M. I. Nelson and E. C. Holmes. The evolution of epidemic influenza. *Nat Rev Genet*, 8(3):196–205, Mar 2007.

[10] P. Palese. Making better influenza virus vaccines? *Emerg Infect Dis*, 12(1):61–65, Jan 2006.

[11] A. Rambaut, O. G. Pybus, M. I. Nelson, C. Viboud, J. K. Taubenberger, and E. C. Holmes. The genomic and epidemiological dynamics of human influenza a virus. *Nature*, 453(7195):615–619, May 2008.

[12] S. Ray and T. B. Kepler. Amino acid biophysical properties in the statistical prediction of peptide-mhc class i binding. *Immunome Res*, 3:9, 2007.

[13] C. A. Russell, T. C. Jones, I. G. Barr, N. J. Cox, R. J. Garten, V. Gregory, I. D. Gust, A. W. Hampson, A. J. Hay, A. C. Hurt, J. C. de Jong, A. Kelso, A. I. Klimov, T. Kageyama, N. Komadina, A. S. Lapedes, Y. P. Lin, A. Mosterin, M. Obuchi, T. Odagiri, A. D. M. E. Osterhaus, G. F. Rimmelzwaan, M. W. Shaw, E. Skepner, K. Stohr, M. Tashiro, R. A. M. Fouchier, and D. J. Smith. The global circulation of seasonal influenza a (h3n2) viruses. *Science*, 320(5874):340–346, Apr 2008.

[14] D. J. Smith, A. S. Lapedes, J. C. de Jong, T. M. Bestebroer, G. F. Rimmelzwaan, A. D. M. E. Osterhaus, and R. A. M. Fouchier. Mapping the antigenic and genetic evolution of influenza virus. *Science*, 305(5682):371–376, Jul 2004.

[15] D. A. Steinhauer and J. J. Skehel. Genetics of influenza viruses. *Annu Rev Genet*, 36:305–332, 2002.

A Lung Cancer Outcome Calculator Using Ensemble Data Mining on SEER Data

Ankit Agrawal, Sanchit Misra, Ramanathan Narayanan, Lalith Polepeddi, Alok Choudhary
Dept. of Electrical Engg. and Computer Science
Northwestern University
2145 Sheridan Rd
Evanston, IL 60201
USA
{ankitag,smi539,ran310,choudhar}@eecs.northwestern.edu, lpolepeddi@u.northwestern.edu

ABSTRACT

We analyze the lung cancer data available from the SEER program with the aim of developing accurate survival prediction models for lung cancer using data mining techniques. Carefully designed preprocessing steps resulted in removal/ modification/splitting of several attributes, and 2 of the 11 derived attributes were found to have significant predictive power. Several data mining classification techniques were used on the preprocessed data along with various data mining optimizations and validations. In our experiments, ensemble voting of five decision tree based classifiers and meta-classifiers was found to result in the best prediction performance in terms of accuracy and area under the ROC curve. Further, we have developed an on-line lung cancer outcome calculator for estimating risk of mortality after 6 months, 9 months, 1 year, 2 year, and 5 years of diagnosis, for which a smaller non-redundant subset of 13 attributes was carefully selected using attribute selection techniques, while trying to retain the predictive power of the original set of attributes. The on-line lung cancer outcome calculator developed as a result of this study is available at http://info.eecs.northwestern.edu:8080/ LungCancerOutcomeCalculator/

Categories and Subject Descriptors

J.3 [**Life and Medical Sciences**]: Medical information systems

General Terms

Measurement

Keywords

Ensemble data mining, Lung cancer, Predictive modeling, Outcome calculator

BIOKDD 2011, August 2011, San Diego, CA, USA

1. INTRODUCTION

Respiratory (lung) cancer is the second most common cancer [1], and the leading cause of cancer-related deaths among men and women in the USA [2]. Survival rate for lung cancer is estimated to be 15% after 5 years of diagnosis [28].

The Surveillance, Epidemiology, and End Results (SEER) Program [4] of the National Cancer Institute is an authoritative repository of cancer statistics in the United States [3]. It is a population-based cancer registry which covers about 26% of the US population across several geographic regions and is the largest publicly available domestic cancer dataset. The data includes patient demographics, cancer type and site, stage, first course of treatment, and follow-up vital status. The SEER program collects cancer data for all invasive and in situ cancers, except basal and squamous cell carcinomas of the skin and in situ carcinomas of the uterine cervix [28]. The 'SEER limited-use data' is available from the SEER website on submitting a SEER limited-use data agreement form. [20] presents an overview study of the cancer data at all sites combined and on selected, frequently occurring cancers from the SEER data. The SEER data attributes can be broadly classified as demographic attributes (e.g. age, gender, location), diagnosis attributes (e.g. primary site, histology, grade, tumor size), treatment attributes (e.g. surgical procedure, radiation therapy), and outcome attributes (e.g. survival time, cause of death), which makes the SEER data ideal for performing outcome analysis studies.

There have been numerous statistical studies using the SEER data like demographic and epidemiological studies of rare cancers [35], assessing susceptibility to secondary cancers that emerge after a primary diagnosis [29], performing survival analysis [28], studying the impact of a certain type of treatment on overall survival [12], studying conditional survival (measuring prognosis of patients who have already survived a period of time after diagnosis) [31, 32, 11], amongst many others. There also have been scattered applications of data mining using SEER data for breast cancer survival prediction [25, 14, 6, 16] and a few studying lung cancer survival [10, 13].

Applying data mining techniques to cancer data is useful to rank and link cancer attributes to the survival outcome. Further, accurate outcome prediction can be extremely useful for doctors and patients to not only estimate survivability, but also aid in decision making to determine the best course of treatment for a patient, based on patient-

specific attributes, rather than relying on personal experiences, anecdotes, or population-wide risk assessments. Here we use data mining techniques to predict survival of respiratory cancer patients, at the end of 6 months, 9 months, 1 year, 2 years, and 5 years of diagnosis. Experiments with several classifiers were conducted to find that many meta classifiers used with decision trees can give impressive results, which can be further improved by combining the resulting prediction probabilities from several classifiers using an ensemble voting scheme. Further, we have developed an an on-line lung cancer outcome calculator to estimate the patient-specific risk for mortality due to lung cancer at the end of 6 months, 9 months, 1 year, 2 years, and 5 years.

The rest of the paper is organized as follows: Section 2 summarizes the recent research relevant to the problem, followed by a description of the major classification schemes used in this study in Section 3. The survival prediction system is presented in Section 4, and Experiments and results are presented in Section 5. The lung cancer outcome calculator is described in Section 6, and the conclusion and future work is presented in Section 7.

2. RELATED WORK

With SEER data being available in the public domain, there is a mature literature on the statistics of SEER data [35, 29, 28, 12, 31, 32, 11], many of them using the the SEERStat software provided by SEER itself.

In addition, there also have been a few data mining applications, which has become a very significant component of cancer research and survivability analysis. A number of techniques based on data mining have been proposed for the survivability analysis of various cancers. [36] uses decision trees and artificial neural networks for survivability analysis of breast cancer, diabetes and hepatitis. [25] uses artificial neural networks on SEER data to predict breast cancer survival. [14] empirically compared three data mining techniques: neural networks, decision trees and logistic regression for the task of predicting 60 months breast cancer survival. They applied these techniques on 2000 version of SEER data. They found that decision trees performed the best with 93.6% accuracy, followed by neural networks. [6] found that the pre-classification process used by [14] was not accurate in determining the records of the 'not survived' class. The authors of [6] corrected this and investigated Naive bayes, the back-propagated neural networks, and the C4.5 decision tree algorithm using the data mining tool WEKA. Decision Trees and Neural networks performed the best with 86.7% and 86.5% accuracy respectively. According to the authors, the difference in results reported by [14] and those obtained by them is due to the facts that they used a newer database (2000 vs. 2002), a different class-distribution ($109,659$ and $93,273$ vs. $35,148$ and $116,738$) and different toolkits (industrial grade tools vs. WEKA).

The authors in [16] studied 5-year survival of follow-up patients in SEER data in 2002 who were diagnosed as breast cancer from $1992-1997$. They compared seven data mining algorithms (artificial neural network, naive bayes, bayes net, decision trees with naive bayes, decision trees (ID3), decision trees(J48)) and logistic regression model. The conclusion was that logistic regression (accuracy 85.8%) and decision trees (accuracy 85.6%) were the best ones with high accuracies and high sensitivities. [6, 16] also showed that there is a significant imbalance between survived and not-survived classes for the five year survival problem: 80% survived, 20% not-survived. This imbalance in data can potentially affect the accuracy of the developed model. [34] addressed this problem and used under-sampling to balance the two classes. The conclusion was that the performance of the models is best while the distribution of data is approximately equal.

Modeling survival for lung cancer is not as developed as for breast cancer. [27] performs a statistical analysis of the SEER data and computes survival percentage based on gender, race, geographic area, cancer stage, etc. [10] used SEER data containing records of lung cancer patients diagnosed from 1988 through 1998. They examined the following attributes: AJCC stage, grade, histological type and gender. For each of the first three attributes, they considered four popular values that are generally used in lung cancer studies. The attribute gender had two values: male and female. This gave them 128 ($4 \times 4 \times 4 \times 2$) possible combinations of values. They applied ensemble clustering on those combinations to get seven clusters and studied survival patterns of those clusters. [13] used SEER data for patients diagnosed of cancer of lung or bronchus from the year 1988 through 2001. They studied 8 months survivability of lung cancer. They compared penalized logistic regression and SVM for survival prediction of lung cancer, and found that logistic regression resulted in better prediction performance (in terms of <sensitivity, specificity> pair). They also note that SVM-modeling is significantly slow, taking hours to train.

3. CLASSIFICATION SCHEMES

We used several classification schemes resulting in identification of top 5 classification schemes, plus ensemble voting scheme to combine the prediction probabilities from the top 5 (details presented in Experiments and Results section). This section presents a brief description of the classifiers and meta-classifiers used in the experiments reported in this paper.

1. **Support vector machines**: SVMs are based on the Structural Risk Minimization(SRM) principle from statistical learning theory. A detailed description of SVMs and SRM is available in [30]. In their basic form, SVMs attempt to perform classification by constructing hyperplanes in a multidimensional space that separates the cases of different class labels. It supports both classification and regression tasks and can handle multiple continuous and nominal variables. Different types of kernels can be used in SVM models, like linear, polynomial, radial basis function (RBF), and sigmoid. Of these, the RBF kernel is the most recommended and popularly used, since it has finite response across the entire range of the real x-axis.

2. **Artificial neural networks**: ANNs are networks of interconnected artificial neurons, and are commonly used for non-linear statistical data modeling to model complex relationships between inputs and outputs. The network includes a hidden layer of multiple artificial neurons connected to the inputs and outputs with different edge weights. The internal edge weights are 'learnt' during the training process using techniques like back propagation. Several good descriptions of neural networks are available [7, 17].

3. **J48 decision tree**: In a decision tree classifier, the internal nodes denote the different attributes whose values would be used to decide on the classification path, and the branches denote the split depending on the attribute values, while the leaf nodes denote the final value (classification) of the dependent variable. While constructing the decision tree, the J48 algorithm [26] identifies the attribute that must be used to split the tree further based on the notion of information gain/gini impurity.

4. **Random forest**: The Random Forest [8] classifier consists of multiple decision trees. The final class of an instance in a Random Forest is assigned by outputting the class that is the mode of the outputs of individual trees, which can produce robust and accurate classification, and ability to handle a very large number of input variables. It is relatively robust to overfitting and can handle datasets with highly imbalance class distributions.

5. **LogitBoost**: Boosting is a technique that can dramatically improve the performance of several classification techniques by sequentially applying them repeatedly to re-weighted versions of the input data, and taking a weighted majority vote of the sequence of classifiers thereby produced. In [19], the authors explain the theoretical connection between Boosting and additive models. The LogitBoost algorithm is an implementation of additive logistic regression which performs classification using a regression scheme as the base learner, and can handle multi-class problems.

6. **Decision stump**: A decision stump [33] is a weak tree-based machine learning model consisting of a single-level decision tree with a categorical or numeric class label. Decision stumps are usually used in ensemble machine learning techniques.

7. **Random subspace**: The Random Subspace classifier [23] constructs a decision tree based classifier consisting of multiple trees, which are constructed systematically by pseudo-randomly selecting subsets of features, trying to achieve a balance between overfitting and achieving maximum accuracy. It maintains highest accuracy on training data and improves on generalization accuracy as it grows in complexity.

8. **Reduced error pruning tree**: Commonly known as REPTree [33], it is a implementation of a fast decision tree learner, which builds a decision/regression tree using information gain/variance and prunes it using reduced-error pruning.

9. **Alternating decision tree**: ADTree [18] is decision tree classifier which supports only binary classification. It consists of two types of nodes: decision nodes (specifying a predicate condition, like 'age' $>$ 45) and prediction nodes (containing a single real-value number). An instance is classified by following all paths for which all decision nodes are true and summing the values of any prediction nodes that are traversed. This is different from the J48 decision tree algorithm in which an instance follows only one path through the tree.

10. **Voting**: Voting is a popular ensemble technique for combining multiple classifiers. It has been shown that ensemble classifiers using voting may outperform the individual classifiers in certain cases [24]. Here we combine multiple classifiers by using the average of probabilities generated by each classifier. The base classifiers used for the voting scheme were LogitBoost (with DecisionStump), RandomSubSpace (with REPTree), J48 decision tree, Random Forests, and ADTree.

4. SURVIVAL PREDICTION SYSTEM

Understanding and cleaning data to prepare it for a data mining analysis is one of the most important steps in the data mining approaches. Appropriate preprocessing, therefore, becomes extremely crucial in any kind of predictive modeling, including that of cancer survival, as also widely accepted by numerous other related studies. The proposed respiratory cancer survival prediction system consists of four stages:

1. **SEER-related preprocessing**: This is the first stage preprocessing designed according to the way SEER program records, codes, and releases the data. There are three principle steps in this stage:

   (a) Convert apparently numeric attributes to nominal, e.g. marital status, sex.

   (b) Split appropriate numeric attributes into numeric and nominal parts, e.g. tumor size. ('CS TUMOR SIZE' gives the exact size of the tumor in mm, if it is known. But in some cases, the doctor notes may say 'less than 2cm', in which case the coder assigns a value of 992 to the field, which, if used as a numeric value, would correspond to 992mm, which is incorrect)

   (c) Construct survival time in months (numeric) from SEER format of YYMM.

2. **Problem-specific preprocessing**: This is the second stage preprocessing which is specific to the problem of survival prediction. The following are the steps in this stage:

   (a) Select data records for a particular time period of interest.

   (b) Filter the attributes that vary too much or too little, since they do not have significant predictive power.

   (c) For cancer-specific survival analysis, remove records where the patient died because of something other than the cancer in study.

   (d) For cancer-specific survival analysis, remove attributes apart from survival time, which directly or indirectly specify the outcome, e.g. cause of death, whether the patient is still alive.

   (e) For binary class prediction, derive appropriate binary attributes for survival, e.g. 5-year survival.

3. **Predictive modeling**: This is where data mining classifiers are employed to construct predictive models for cancer-specific survival, on the preprocessed data. The two straightforward steps of this stage are:

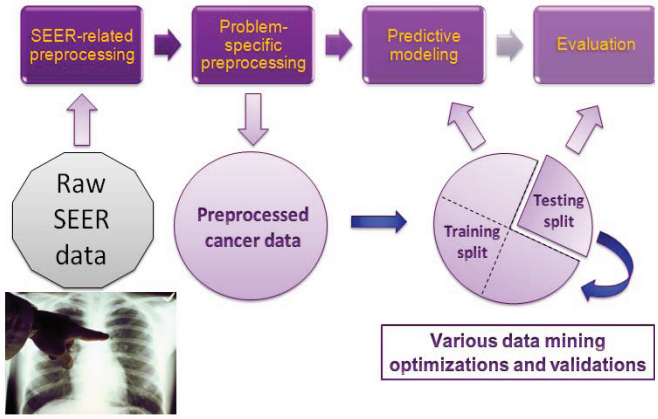


Figure 1: Block-diagram of the survival prediction system

(a) Split the preprocessed data in training and testing sets (or use cross validation)

(b) Construct a model on the training data using data mining classifiers, e.g. Naive bayes, logistic regression, decision trees, etc., including an ensemble of different classifiers.

4. **Evaluation**: In this stage, the predictive model is evaluated on the testing data.

   (a) Compare the survival predictions from the predictive model on unseen data (testing set) against known survival.

   (b) Calculate performance metrics like accuracy (percentage of predictions that are correct), precision (percentage of positive predictions that are correct), recall/sensitivity (percentage of positive labeled records that were predicted as positive), specificity (percentage of negatively labeled records that were predicted as negative), area under the ROC curve (a measure of discriminative power of the model), etc.

Fig. 1 presents the block diagram of the survival prediction system with carefully designed preprocessing steps followed by modeling and evaluation with different data mining optimizations and validations.

5. EXPERIMENTS AND RESULTS

In this study, we used the data in the SEER November 2008 Limited-Use Data files [4] (released in April 2009) from nine SEER registries (Atlanta, Connecticut, Detroit, Hawaii, Iowa, New Mexico, San Francisco-Oakland, Seattle-Puget Sound, and Utah). The SEER data used in this study had a follow-up cutoff date of December 31, 2006, i.e., the patients were diagnosed and followed-up upto this date. In our experiments, we used the WEKA toolkit for data mining [22].

The SEER-related preprocessing resulted in modification and splitting of several attributes, many of which were found to have significant predictive power. In particular, 2 out of 11 newly created (derived) attributes were within the top 13 attributes that were selected to be used in the lung cancer outcome calculator. These were a) the number of regional lymph nodes that were removed and examined by the pathologist; and b) number of malignant/in-situ tumors. These attributes were derived from 'Regional Nodes Examined' and 'Sequence Number-Central' respectively from raw SEER data, both of which had nominal values encoded within the same attribute, with the latter also encoding non-malignant tumors.

Subsequently, we selected the data for the patients diagnosed between 1998 and 2001. This choice was made because of the following: Since we wanted to do a survival prediction for upto 5-years, and the follow-up cutoff date for the SEER data in study was December 31, 2006, we used the data for cancer patients with year of diagnosis as 2001 or before. Moreover, since several important attributes were introduced to the SEER data in 1998 (like RX Summ-Surg Site 98-02, RX Summ-Scope Reg 98-02, RX Summ-Surg Oth 98-02, Summary stage 2000 (1998+)), we further restricted the patient data with year of diagnosis as 1998 or after. Thus, we selected the data of all cases of respiratory cancer patients in the above mentioned nine SEER registries diagnosed between 1998 and 2001. There were a total of 70132 such instances. After removing the attributes which varied too much or too little (and hence did not have significant predictive power), we were left with a total of 68 attributes. We further removed all instances where the patient died because of something other than respiratory cancer, reducing the number of instances to 57254. After removing cause of death and related attributes, we were left with 64 attributes (including survival time in months). Since the survival rate of respiratory cancer is extremely low, we derived binary attributes for 6-month, 9-month, 1-year, 2-year, and 5-year survival. The number of attributes were thus reduced from 118 in the initial dataset to 64, i.e., 63 predictor attributes and 1 outcome attribute (which can be 6-month/9-month/1-year/2-year/5-year survival).

Table 1 presents the distribution of not-survived and survived patients at the end of 6 months, 9 months, 1 year, 2 years, and 5 years of diagnosis. It clearly shows that the distribution can be quite lopsided for some classes.

For classification, we built predictive models using more than 30 different classification schemes, and of those which completed execution in reasonable time, the top 5 were selected:

1. J48 decision tree
2. Random forest
3. LogitBoost (with Decision Stump as the underlying classifier)
4. Random subspace (with REPTree as the underlying classifier)
5. Alternating decision tree

Because these 5 classification schemes gave good performance, we also decided to use the ensemble voting technique for combining the results from these classifiers. Voting can combine the probabilities generated by each classifier in different ways, like average, product, majority, maximum, minimum, median. After some initial experiments with the

Table 1: Class distribution

Fraction/Survival class	6-month	9-month	1-year	2-year	5-year
Not-survived	38.85%	49.12%	57.04%	72.79%	83.23%
Survived	61.15%	50.88%	42.96%	27.21%	16.77%

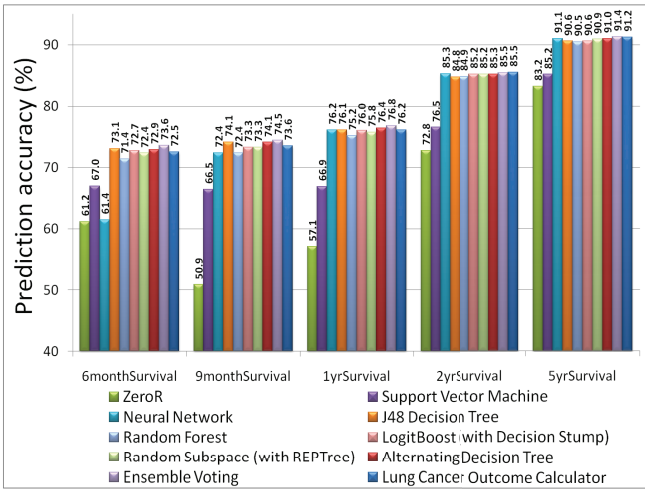


Figure 2: Prediction accuracy comparison amongst different classification techniques. The lung cancer outcome calculator uses ensemble voting scheme using just 13 predictor attributes.

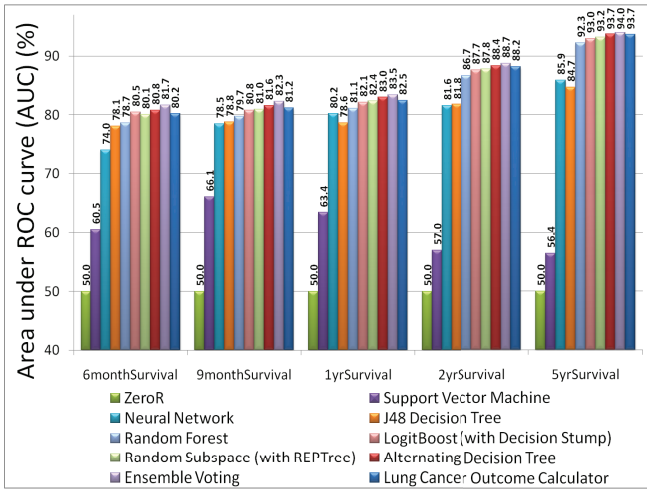


Figure 3: Prediction performance comparison in terms of area under the ROC curve (AUC). The lung cancer outcome calculator uses ensemble voting scheme using just 13 predictor attributes.

different ways of combining the probabilities (which gave similar results), we chose to calculate the resulting probability by taking the average of the probabilities generated by each classifier.

We conducted experiments with the above mentioned 6 (=5+1) classification schemes, on each of the 5 datasets (with class variable as 6-month, 9-month, 1-year, 2-year, and 5-year survival). 10-fold cross-validation was used for training and testing, and 10 runs of each <dataset, algorithm> were conducted (with different cross-validation folds) for statistical analysis of the performance comparison. Thus, there were a total of $5 \times 6 \times 10 \times 10 = 3000$ runs. Next, we present the results.

The ZeroR classifier is commonly used as a baseline classifier to measure the improvement in prediction performance due to modeling over simply going by statistical majority, i.e., always predicting the majority class. Fig. 2 presents the overall prediction accuracy of the above-mentioned 6 classification schemes, along with ZeroR classifier, on each of the five datasets. Since, accuracy results can be often misleading due to imbalanced classes, the area under the ROC curve (AUC) is considered a better metric to measure the ability of the model to discriminate between the different class values. Fig. 3 presents the area under the ROC curve (AUC) for the same. For completeness, Fig. 2 and Fig. 3 also present the classification results obtained by using support vector machines (with RBF kernel) [15, 9] and neural networks, although their results were found to be less accurate and inconsistent as compared to other classifiers. Moreover, the execution time for constructing SVM and neural network models was significantly larger as compared to other models. Therefore, instead of multiple runs of cross-validation, a single run of training-testing split (training on 66% data, and testing on 33%) was conducted to measure the accuracy of these models. The SVM models required around 15 CPU hours for construction (slow training of SVM models is also acknowledged in [13]), and the neural network model construction did not complete after more than 400 CPU hours of execution time. The results for neural network reported in this paper were obtained on the dataset with a reduced attribute set (from 63 attributes to 13 attributes, used for the tool as described later), which, for the ensemble voting scheme was found to give similar prediction accuracy as with using all 63 attributes. Neural network modeling on this smaller dataset took about 80 CPU hours. Since for this data, better prediction quality was obtained by other models that could be constructed faster than SVM and neural network models, these models were not investigated further. Ability to construct the models in reasonable time is crucial to enable regular model updates by incorporating new data as and when it becomes available.

From Fig. 2 and Fig. 3, it is clear that ensemble voting classification scheme gives the best prediction performance, both in terms of prediction accuracy and AUC, which was also found to be (statistically) significantly better than the J48 decision tree as the base learner, at 5% significance level. Some important observations from these figures are as follows. For 5-year survival prediction, the baseline classifier (ZeroR) classifies all records as 'not survived' (majority class), achieving a prediction accuracy of 83.2% because of the imbalanced class distribution, which seems quite impressive, but is clearly uninformative and not useful in practice. Model-driven prediction for the same 5-year prediction boosts the prediction accuracy up to 91.4%, which means an

effective reduction of error rate from 16.8% to 8.6%, thereby reducing the error rate almost by a factor of 2. Apart from prediction accuracy, an excellent discriminative power (discrimination between death and survival) of 5-year survival prediction model was also obtained with a high AUC of 0.94.

In general, it is not straightforward to compare prediction results on different datasets with different class distributions. The work in [10] had applied ensemble clustering to study survival patterns of obtained clusters, but no test results were reported. Moreover, the study used only 4 attributes with popular values of those attributes. The predictive models used in the current study are more general using all available attributes. The work in [13] studied 8 months survivability of lung cancer using variations of logistic regression and SVM techniques, and reported results in terms of sensitivity and specificity. Again, their results are not directly comparable to ours, since both the dataset and the target class are different. More specifically, we use a more recent release of the SEER database with newer attributes, and a different time period of the diagnosed cases, as compared to [13]. They report sensitivity and specificity as measures of the quality of prediction. Some of the best <sensitivity, specificity> combinations in their experiments were: <74.62, 70.57>, <74.84, 68.26>, <75.44, 63.27>. We had conducted experiments for 9-month survival, and the <sensitivity, specificity> combination with ensemble voting scheme was <78.90, 70.15>.

6. ON-LINE LUNG CANCER OUTCOME CALCULATOR

Further, for the purpose of building an on-line tool for lung cancer outcome prediction, we used correlation-based feature subset selection technique [21] to identify a smaller non-redundant subset of attributes which were highly correlated with the outcome variable while having low intercorrelation amongst themselves. The goal here was to make the tool convenient to use by reducing the number of attributes, while trying to retain the predictive power of the original set of attributes in the preprocessed data. The attribute subsets obtained for the five different outcome variables were combined, and clearly redundant attributes were manually removed. SEER-specific attributes were further removed to make the calculator more easily applicable to new patients. The calculator uses the resulting 13 input variables as shown in Fig. 4 (with relative predictive power) to estimate lung-cancer-specific mortality risk using the ensemble voting scheme. Following is a brief description of these attributes. The original SEER names of the attributes are also mentioned wherever significantly different from the names used in the calculator.

1. **Age at diagnosis**: Numeric age of the patient at the time of diagnosis for lung cancer.

2. **Birth place**: The place of birth of the patient. There are 198 options available to select for this attribute (based on the values observed in the SEER database).

3. **Cancer grade**: A descriptor of how the cancer cells appear and how fast they may grow and spread. Available options are - well-differentiated, moderately differentiated, poorly differentiated, undifferentiated, and undetermined.

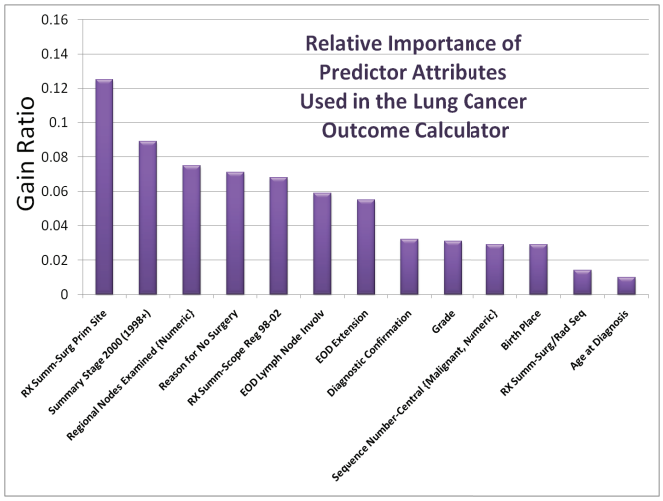


Figure 4: The attributes used in the lung cancer outcome calculator along with their relative predictive power.

4. **Diagnostic confirmation**: The best method used to confirm the presence of lung cancer. Available options are - positive histology, positive cytology, positive microscopic confirmation (method unspecified), positive laboratory test/marker study, direct visualization, radiology, other clinical diagnosis, and unknown if microscopically confirmed.

5. **Farthest extension of tumor**: The farthest documented extension of tumor away from the lung, either by contiguous extension (regional growth) or distant metastases (cancer spreading to other organs far from primary site through bloodstream or lymphatic system). There are 20 options available to select for this attribute. The original SEER name for this attribute is 'EOD extension'.

6. **Lymph node involvement**: The highest specific lymph node chain that is involved by the tumor. Cancer cells can spread to lymph nodes near the lung, which are part of the lymphatic system (the system that produces, stores, and carries the infection-fighting-cells. This can often lead to metastases. There are 8 options available for this attribute. The original SEER name for this attribute is 'EOD Lymph Node Involv'.

7. **Type of surgery performed**: The surgical procedure that removes and/or destroys cancerous tissue of the lung, performed as part of the initial work-up or first course of therapy. There are 25 options available for this attribute, like cyrosurgery, fulguration, wedge resection, laser excision, pneumonectomy, etc. The original SEER name for this attribute is 'RX Summ-Surg Prim Site'.

8. **Reason for no surgery**: The reason why surgery was not performed (if not). Available options are - surgery performed, surgery not recommended, contraindicated due to other conditions, unknown reason, patient or patient's guardian refused, recommended but unknown if done, and unknown if surgery performed.

9. **Order of surgery and radiation therapy**: The order in which surgery and radiation therapies were administered for those patients who had both surgery and radiation. Available options are - no radiation and/or surgery, radiation before surgery, radiation after surgery, radiation both before and after surgery, intraoperative radiation therapy, intraoperative radiation with other radiation given before/after surgery, and sequence unknown but both surgery and radiation were given. The original SEER name for this attribute is 'RX Summ-Surg/Rad Seq'.

10. **Scope of regional lymph node surgery**: It describes the removal, biopsy, or aspiration of regional lymph node(s) at the time of surgery of the primary site or during a separate surgical event. There are 8 options available for this attribute. The original SEER name for this attribute is 'RX Summ-Scope Reg 98-02'.

11. **Cancer stage**: A descriptor of the extent the cancer has spread, taking into account the size of the tumor, depth of penetration, metastasis, etc. Available options are - in situ (noninvasive neoplasm), localized (invasive neoplasm confined to the lung), regional (extended neoplasm), distant (spread neoplasm), and unstaged/unknown. The original SEER name for this attribute is 'Summary Stage 2000 (1998+)'.

12. **Number of malignant tumors in the past**: An integer denoting the number of malignant tumors in the patient's lifetime so far. This attribute is derived from the SEER attribute 'Sequence Number-Central', which encodes both numeric and categorical values for both malignant and benign tumors within a single attribute. As part of the preprocessing, the original SEER attribute was split into numeric and nominal parts, and the numeric part was further split into 2 attributes representing number of malignant and benign tumors respectively.

13. **Total regional lymph nodes examined**: An integer denoting the total number of regional lymph nodes that were removed and examined by the pathologist. This attributed was derived by extracting the numeric part of the SEER attribute 'Regional Nodes Examined'.

Prediction performance with just 13 attributes used in the calculator is also presented in Fig. 2 and Fig. 3, which shows only marginal decrease in prediction performance as compared to using all 63 variables. A careful selection of attributes for the calculator has therefore resulted in a decrease in the number of attributes from 63 to 13, while incurring only a marginal cost on prediction accuracy (Prediction accuracy = 91.2% for 5-year survival prediction with 13 attributes, as compared to 91.4% with 63 attributes; AUC = 0.937 for 5-year survival prediction with 13 attributes, as compared to 0.94 with 63 attributes). It seems that these 13 attributes were able to reasonably encode the information available in the previously used 63 attributes, which prevents any significant drop in accuracy. It is also interesting that the birth place shows up as a significant attribute in the set of 13 attributes. Fig. 5 shows a screenshot of the lung cancer outcome calculator. A preliminary version of the calculator was reported in a recent poster abstract [5].

7. CONCLUSION AND FUTURE WORK

In this paper, we used different meta classification schemes with underlying decision tree classifiers to construct models for survival prediction for respiratory cancer patients. Prediction accuracies of 73.61%, 74.45%, 76.80%, 85.45%, and 91.35% was obtained for the 6-month, 9-month, 1-year, 2-year, and 5-year respiratory cancer survival prediction using the ensemble voting classification scheme. Further, a lung cancer outcome calculator was developed using carefully selected 13 attributes, while retaining the prediction quality.

Given the prediction quality, we believe that the calculator can be very useful to not only accurately estimate survivability of a lung cancer patient, but also aid doctors in decision making and improve informed patient consent by providing a better understanding of the risks involved in a particular treatment procedure, based on patient-specific attributes. Accurate risk prediction can potentially also save valuable resources by avoiding high risk procedures that may not be necessary for a particular patient.

Future work includes developing models for conditional survival prediction (e.g. 5-year prediction, given that the patient has already survived for 1 year), and exploring the use of undersampling/oversampling to deal with unbalanced data. We also plan to do similar analysis for other cancers, and developing on-line cancer outcome calculators for them.

8. ACKNOWLEDGMENTS

This work was supported in part by NSF grants CNS-0551639, IIS-0536994, NSF HECURA CCF-0621443, and NSF SDCI OCI-0724599, NSF IIS-0905205, DOE FASTOS award number DE-FG02-08ER25848 and DOE SCIDAC-2: Scientific Data Management Center for Enabling Technologies (CET) grant DE-FC02-07ER25808.

9. REFERENCES

[1] Introduction to lung cancer. National Cancer Institute, SEER training modules, URL: http://training.seer.cancer.gov/lung/intro/ accessed: April 29, 2010.

[2] Lung cancer statistics. Centers for Disease Control and Prevention, URL: http://www.cdc.gov/cancer/lung/statistics/ accessed: April 29, 2010.

[3] Overview of the seer program. Surveillance Epidemiology and End Results, URL: http://seer.cancer.gov/about/ accessed: April 29, 2010.

[4] Surveillance, epidemiology, and end results (seer) program (www.seer.cancer.gov) limited-use data (1973-2006). National Cancer Institute, DCCPS, Surveillance Research Program, Cancer Statistics Branch, 2008. released April 2009, based on the November 2008 submission.

[5] A. Agrawal, S. Misra, R. Narayanan, L. Polepeddi, and A. Choudhary. Poster: A lung cancer mortality risk calculator based on seer data. In *Proc. of IEEE 1st International Conference on Computational Advances in Bio and Medical Sciences (ICCABS)*, pages 233–233, 2011.

[6] A. Bellaachia and E. Guven. Predicting breast cancer survivability using data mining techniques. In *Proceedings of Ninth Workshop on Mining Scientific*

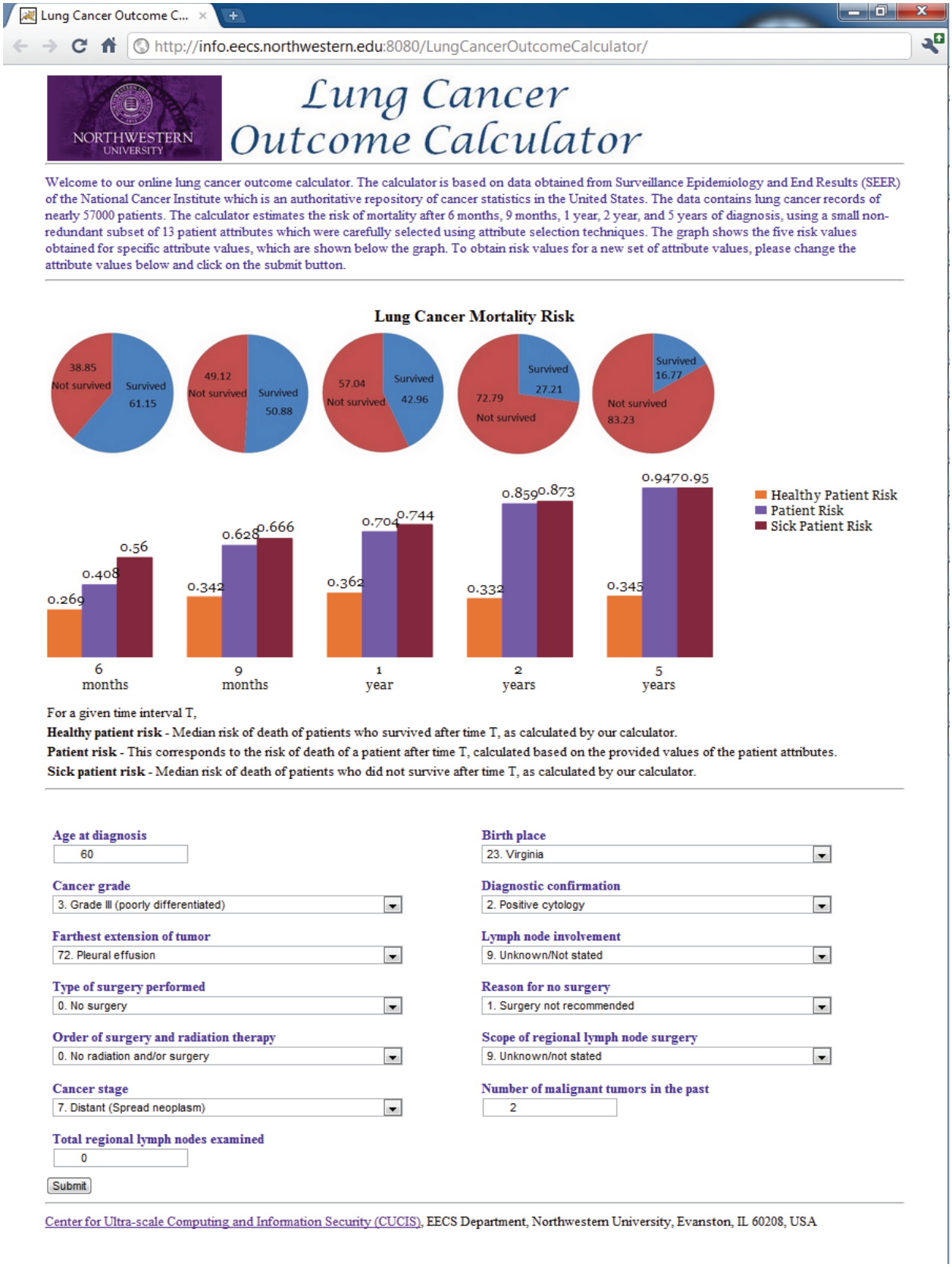


Figure 5: Screenshot of the Lung Cancer Outcome Calculator.

and Engineering Datasets in conjunction with the Sixth SIAM International Conference on Data Mining (SDM 2006), 2006.

[7] C. Bishop. *Neural Networks for Pattern Recognition*. Oxford: University Press, 1995.

[8] L. Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[9] C.-C. Chang and C.-J. Lin. *LibSVM - A Library for Support Vector Machines*, 2001. Software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm.

[10] D. Chen, K. Xing, D. Henson, L. Sheng, A. Schwartz, and X. Cheng. Developing prognostic systems of cancer patients by ensemble clustering. *Journal of Biomedicine and Biotechnology*, 2009, 2009.

[11] M. Choi, C. D. Fuller, C. R. T. Jr., and S. J. Wang. Conditional survival in ovarian cancer: Results from the seer dataset 1988-2001. *Gynecologic Oncology*, 109(2):203 – 209, 2008.

[12] N. Coburn, A. Govindarajan, C. Law, U. Guller, A. Kiss, J. Ringash, C. Swallow, and N. Baxter. Stage-specific effect of adjuvant therapy following gastric cancer resection: a population-based analysis of 4,041 patients. *Annals of Surgical Oncology*, 15:500–507, 2008.

[13] F. D. Machine learning methods in the analysis of lung cancer survival data. *DIMACS Technical Report 2005-35 February 2006*.

[14] D. Delen, G. Walker, and A. Kadam. Predicting breast cancer survivability: a comparison of three data mining methods. *Artificial Intelligence in Medicine*, 34(2):113 – 127, 2005.

[15] Y. EL-Manzalawy and V. Honavar. *WLSVM: Integrating LibSVM into Weka Environment*, 2005. Software available at http://www.cs.iastate.edu/ yasser/wlsvm.

[16] A. Endo, T. Shibata, and H. Tanaka. Comparison of seven algorithms to predict breast cancer survival. *Biomedical Soft Computing and Human Sciences*, 13(2):11–16, 2008.

[17] L. Fausett. *Fundamentals of Neural Networks*. New York, Prentice Hall, 1994.

[18] Y. Freund and L. Mason. The alternating decision tree learning algorithm. In *Proceeding of the Sixteenth International Conference on Machine Learning*, pages 124–133. Citeseer, 1999.

[19] J. Friedman, T. Hastie, and R. Tibshirani. Special invited paper. additive logistic regression: A statistical view of boosting. *Annals of statistics*, 28(2):337–374, 2000.

[20] L. A. Gloeckler Ries, M. E. Reichman, D. R. Lewis, B. F. Hankey, and B. K. Edwards. Cancer Survival and Incidence from the Surveillance, Epidemiology, and End Results (SEER) Program. *Oncologist*, 8(6):541–552, 2003.

[21] M. Hall. *Correlation-based feature selection for machine learning*. PhD thesis, Citeseer, 1999.

[22] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The weka data mining software: An update. *SIGKDD Explorations*, 11(1), 2009.

[23] T. Ho. The random subspace method for constructing decision forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(8):832–844, 1998.

[24] J. Kittler. Combining classifiers: A theoretical framework. *Pattern Analysis & Applications*, 1(1):18–27, 1998.

[25] L. M, L. J, B. HB, T. S, P. L, and J. H. Artificial neural networks applied to survival prediction in breast cancer. *Oncology*, 57:281–286, 1999.

[26] J. Quinlan. *C4. 5: programs for machine learning*. Morgan Kaufmann, 1993.

[27] L. A. G. Ries and M. P. Eisner. Cancer of the lung. *SEER Survival Monograph: Cancer Survival Among Adults: US SEER Program, 1988-2001, Patient and Tumor Characteristics*.

[28] L. A. G. Ries and M. P. Eisner. *Cancer of the lung*, chapter 9. National Cancer Institute, SEER Program, 2007.

[29] K. Rusthoven, T. Flaig, D. Raben, et al. High incidence of lung cancer after non-muscle-invasive transitional cell carcinoma of the bladder: Implications for screening trials. *Clin Lung Cancer*, 9:106–111, 2008.

[30] V. N. Vapnik. The nature of statistical learning theory. *Springer*, 1995.

[31] S. J. Wang, R. Emery, C. D. Fuller, J.-S. Kim, D. F. Sittig, and C. R. Thomas. Conditional survival in gastric cancer: a seer database analysis. *Gastric Cancer*, 10:153–158, 2007. 10.1007/s10120-007-0424-9.

[32] S. J. Wang, C. D. Fuller, R. Emery, and C. R. Thomas. Conditional survival in rectal cancer: A seer database analysis. *Gastrointest Cancer Res*, 1:84–89, 2007.

[33] I. Witten and E. Frank. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann Pub, 2005.

[34] L. Ya-Qin, W. Cheng, and Z. Lu. Decision tree based predictive models for breast cancer survivability on imbalanced data. In *proceedings of third International Conference on Bioinformatics and Biomedical Engineering (ICBBE)*, 2009.

[35] J. C. Yao, M. Hassan, A. Phan, C. Dagohoy, C. Leary, J. E. Mares, E. K. Abdalla, J. B. Fleming, J.-N. Vauthey, A. Rashid, and D. B. Evans. One Hundred Years After "Carcinoid": Epidemiology of and Prognostic Factors for Neuroendocrine Tumors in 35,825 Cases in the United States. *Journal of Clinical Oncology*, 26(18):3063–3072, 2008.

[36] Z.-H. Zhou and Y. Jiang. Medical diagnosis with c4.5 rule preceded by artificial neural network ensemble. *IEEE Transactions on Information Technology in Biomedicine*, 7(1):37–42, 2003.