

Pattern Recognition Techniques in Microarray Data Analysis

Miao Li, Biao Wang, Zohreh Momeni, and Faramarz Valafar

Department of Computer Science

San Diego State University

San Diego, California, USA

faramarz@sciences.sdsu.edu

Abstract: *The explosion in the production rate of expression level data has highlighted the need for automated techniques that help scientists analyze, understand, and cluster (sequence) the enormous amount of data that is being produced. Example of such problems are analyzing gene expression level data produced by microarray technology on the genomic scale, sequencing genes on the genomic scale, sequencing proteins and amino acids, etc. In this paper we review some of the pattern recognition techniques that have been used in this area.*

Keywords: Pattern recognition, sequencing, microarray, clustering, expression level data

Introduction

Microarray technology relies on the hybridization properties of nucleic acids to monitor DNA or RNA abundance *on a genomic scale*. Microarrays have revolutionized the study of genome by allowing researchers to study the expression of thousands of genes simultaneously for the first time.

The ability to simultaneously study thousands of genes under a host of differing conditions presents an immense challenge in the fields of computational science and data mining. In this survey, we have reviewed methods that help (have been used) in answering both of the above questions. In order to properly comprehend and interpret expression data produced by microarray technology, new computational and data mining techniques need to be developed. The analysis and understanding of microarray data includes a search for genes that have similar or correlated patterns of expression. Among the various frameworks in which pattern recognition has been traditionally formulated, statistical approach, neural network techniques, and methods imported from statistical learning theory are among those that have been applied in microarray data analysis [1]. In the following you will find a surveyed of a number of techniques that have been used in microarray data analysis.

Simple approaches: In order to understand the connection between microarray data and biological function, one could ask the following fundamental question: *how do expression levels of specific genes, or gene-sequences, differ in a control group versus a treatment group?* In other words, in a controlled study, if a specific biological function (condition) is present, how do expression levels change when the function (condition) is turned off (absent) (or visa versa). This line of questioning could be useful in, for instance, determining the effect of a specific genomic treatment of a disease. In such a case, the question would be: has the treatment changed the expression level(s) of specific (target) gene(s)/gene-sequence(s) to noticeably different levels. If so, then have the changes in expression levels resulted in elimination (or alleviating the symptoms) of patient's condition? If the answers to the above questions are "yes", then a correlation between the specific genes/gene-sequences that show changed levels, and the biological function, can be drawn.

One of the simple methods attempting to answer the above question is called the *Fold* approach. In this approach, if the average expression level of a gene has changed by a predetermined number of "folds", that gene expression is declared to have been changed (from on to off, or visa versa) due to the treatment. In many studies, a 2-fold technique is used (rather than 3-fold or 4-fold), in which the average expression level has to change to at least two folds of its initial level in order for it to be classified as changed. The drawback to this method is that it is unlikely that this method reveals the desired correlation between expression data and function, as a predetermined factor of 2 (or 3 or 4) has different significance depending on expression levels of various genes. A further drawback is that this method only compares the

expression level of the gene under question to determine whether it has been turned “on” or “off”. We believe that a better and more biologically relevant method of analysis would be to consider expression patterns of related (or neighboring) genes to determine the “on” or “off” state of the gene currently under observation. The folding technique as it is, does not allow this type of analysis.

Similar to the Fold approach, *T-test* is another simple method applied in gene expression analysis. It manipulates logarithm of expression levels, and requires comparison computation against the means and variances of both treatment and control groups. The fundamental problem with T-test is that it requires repeated treatment and control experiments, which is both tedious and costly. As a result, small number of experimental repetition could affect the reliability of the mean-based approach [2,3].

Karhunen–Loe`ve expansion : Karhunen–Loe`ve expansion [4], as it is known in pattern recognition, is also known as *principal component analysis* (PCA), or *singular value decomposition* (SVD) [5] in statistics. Use of SVD in microarray data analysis has been suggested by various researchers [6-11]. PCA is a linear mathematical technique that finds base vectors that expand the problem space (gene expression space). These vectors are called *principal components* (PCs). In expression data analysis, the vectors are also called *mean expression vectors*, or *eigengenes*. A PC can be thought of as a major pattern in the data set (e.g. gene expression data). The more PCs are used to expand (model) the problem space, the more accurate the representation will be. However, one should also be aware that the lower the significance of a PC, the more noise it represents. So a balance needs to be struck between the need for maximal expansion of the problem space and the need for elimination of noise. In most cases, PCA reduces the dimensionality of the problem space without much loss of generality or information. It is easy to think of each PC as the mean expression vector representing a cluster of expression data (expression pattern). In most studies involving SVD, the technique was used to find underlying patterns or ‘modes’ in expression data with the intention of linking these modes to the action of transcriptional regulators.

The advantage of SVD is its simplicity and ease of understanding of the algorithm. Among the disadvantages, is the method’s inherent linear nature. Because SVD is a linear technique, it works well in problem spaces that are linearly separable. It is yet to be shown that finding underlying modes in expression data is a linearly separable problem. PCA is a powerful technique for the analysis of gene-expression data when used in combination with another classification technique such as k-means clustering or self organizing maps (SOMs) that requires the user to specify the number of clusters. We will discuss both of these methods in the coming sections.

Bayesian Belief Networks (BBN): A more sophisticated approach, *Bayesian probabilistic* framework has been suggested to analyze expression data [2,12,13]. An alternative approach to the SVD methodology described above is to use prior knowledge of the regulatory network’s architecture to design competing models then use Bayesian belief networks to pick the model that best fits the expression data [14]. Gifford and co-workers have used this approach to distinguish between two competing models for galactose regulation [15]. Friedman and co-workers have used Bayesian networks to analyze genome-wide expression data in order to identify significant interactions between genes in a variety of metabolic and regulatory pathways [16,12].

Baldi et. Al. [2] used BBN’s to model “log-expression values by independent normal distributions, parameterized by corresponding means and variances with hierarchical prior distributions. They derive point estimates for both parameter and hyperparameter, and regularize expressions for the variance of each gene by combining the empirical variance with local background variance associated with neighboring genes”. This approach “compares favorably with simple t-test and fold method. It “can accommodate noise, variability, and low replication often typical of microarray data.” [2]

k-means Clustering (Partitioning): k-means is a divisive clustering approach. It partitions data (genes or experiments) into groups that have similar expression patterns. *k* is the number of clusters (also sometimes called buckets, bins, or classes) into which the user believes the data should fall. The number *k* is an input value given to the algorithm by the user. The k-mean clustering algorithm is a three step process. In the first step, the algorithm randomly assigns all training data to one of the *k* clusters. In the second step the *mean inter- and intra-class distances* are calculated. The *mean inter-class distance* (δ^c) of each cluster is

calculated by first calculating a *mean vector* (μ^c) for each cluster and then averaging the distances between the vectors (data) of a cluster and its mean vector. In expression level data, the mean vector is called the *average expression vector*.¹

$$\mu^c = \frac{1}{n} \sum_{i=1}^n v_i^c \quad \text{and} \quad \delta^c = \frac{1}{n} \sum_{i=1}^n \|v_i^c - \mu^c\|$$

The *mean intra-class distance* between two clusters is the distance between their respective mean vectors. $\Delta^{1,2}$, for instance is the mean intra-class distance between clusters 1 and 2:

$$\Delta^{1,2} = \|\mu^1 - \mu^2\|$$

The third step is an iterative step, and its goal is to minimize the mean inter-class distances (δ^c), maximize intra-class distances ($\Delta^{1,2}$), or both, by moving data from one cluster to another. In each iteration one piece of data is moved to a new cluster where it is closest to a μ^c (the mean vector of the new cluster). After each move, new expression vectors for the two effected classes are recalculated. This process continues, until any further move would increase the mean inter-class means (expression variability for each class) or reduce intra-class distances. There are additional (sometimes optional) steps that can be found in variations of the basic k-means clustering algorithm described above. Quackenbush [17] discusses a few optional steps and variations of this algorithm.

K-means clustering is easy to implement. However, the major disadvantage of this method is that the number k is often not known in advance. Another potential problem with this method is that because each gene is uniquely assigned to some cluster, it is difficult for the method to accommodate a large number of stray data points, intermediates or outliers. Further concerns about the algorithm have to do with the algorithms' biological interpretation (in the case of expression data) of the final clustered data. In this regard, Tamayo et al. explain that k-means clustering is "a completely unstructured approach, which proceeds in an entirely local fashion and produces an unorganized collection of clusters that is not conducive to interpretation". [18]

The most recent variant of the k-means clustering algorithm (at the time of this survey) designed specifically for the assessment of gene spots (on the array images) is the work of Bozinov et al [19]. The technique is based on clustering pixels of a target area into foreground and background. The authors report: "results from the analysis of real gene spots indicate that our approach performs superior to other existing analytical methods". [19]

Hierarchical Clustering: There are various hierarchical clustering algorithms that can be applied to microarray data analysis. These include single-linkage clustering, complete-linkage clustering, average-linkage clustering, weighted pair-group averaging, and within pair-group averaging [17,20-22]. These algorithms only differ in the manner in which distances are calculated between the growing clusters and the remaining members of the data set. Hierarchical clustering algorithms usually generate a gene similarity score for all gene combinations, place the scores in a matrix, join those genes that have the highest score, and then continue to join progressively less similar pairs. In the clustering process, after similarity score calculations, the most closely related pairs are identified in an above-diagonal scoring matrix. In this process, a node in the hierarchy is created for the highest-scoring pair, the gene expressed profilers of the two genes are averaged, and the joined elements are weighted by the number of elements they contain. The matrix is then updated replacing the two joined elements by the node. For n genes, the process is repeated $n-1$ times until a single element (that contains all genes) remains.

¹ In the following formulas, we assume a Euclidian measure for distance, and arithmetic averaging for calculating the means. In various forms of the algorithm, various measures of distance and averaging techniques have been used. A popular and more representative measure of distance has been the Mahalanobis distance.

The first report by Wen et al. [20] uses clustering and data-mining techniques to analyze large-scale gene expression data. This report is significant in that it shows how integrating results that were obtained by using various distance metrics can reveal different but meaningful patterns in the data. Eisen et al. [21] also make an elegant demonstration of the power of hierarchical clustering in the analysis of microarray data.

Similar to the k-means algorithm, the advantage of hierarchical clustering lies in its simplicity. A further advantage of hierarchical technique versus the k-means method is that the results from hierarchical clustering methods can easily be visualized. Although hierarchical cluster analysis is a powerful technique and possesses clear advantages for expression data analysis, it also presents researchers with two major drawbacks. The first problem arises from the “greedy” nature of the algorithm. Hierarchical clustering is essentially a greedy algorithm, and like other such algorithms, it suffers from sensitivity to early mistakes in the greedy process. Because, by definition greedy algorithms cannot go back (backtrack) to redo the step that was taken by mistake, “small errors in cluster assignment in early stages of the algorithm can be drastically amplified” [23]. Therefore, the dependence on the results produced by certain arbitrarily imposed clustering structures (that do not correspond to reality) can give rise to misleading results. For instance, in time-course gene expression studies, hierarchical clustering has received mixed reviews. These algorithms often fail to discriminate between different patterns of variation in time. For instance, a gene expression pattern for which a high value is found at an intermediate time point will be clustered with another gene for which a high value is found at a later point in time. These variations have to be separated in a subsequent step.

The second drawback of hierarchical clustering is best described by Quackenbush [17]: “one potential problem with many hierarchical clustering methods is that, as clusters grow in size, the expression vector that represents the cluster might no longer represent any of the genes in the cluster. Consequently, as clustering progresses, the actual expression patterns of the genes themselves become less relevant”. As a result, an active area of research in hierarchical cluster analysis is in detecting when to stop the hierarchical merging of elements. In this direction new hybrid techniques are emerging that combine hierarchical methodology with k-means techniques.

Mixture models and EM (expectation maximization): Mixture models are probabilistic models built by using positive convex combination of distributions taken from a given family of distributions [24,25]. EM algorithm is an iterative algorithm that proceeds in two alternating steps, the E (expectation) step and the M (maximization) step. Applying EM algorithm to the corresponding mixture model can serve as a complementary analysis to standard hierarchical clustering. “An attractive feature of the mixture modeling approach is that the strength of evidence measure for the number of true clusters in the data is computed”. This assessment of reliability addresses the primary deficiency of hierarchical clustering and is often an important question to biologists considering data from microarray studies. A disadvantage of the technique is that variance parameters are difficult to estimate for the mixture model in the case of clusters with small number of samples. “One way to address it is to use a fully Bayesian estimation procedure. It might not work with data having temporal structure, say gene expression of the same population of cells measured at a different number of time points” [24].

Gene Shaving: Gene shaving is a recently developed and popular statistical method for discovering patterns in gene expression data. The original algorithm uses the PCA algorithm and was proposed by Hastie, Tibshirani et al. [26]. Later variations are under development, some of which include the replacement of the PCA step with a nonlinear variety. Gene shaving is designed to identify clusters of genes with coherent expression patterns and large variation across the samples. Using this method, the authors have successfully analyzed gene expression measurements made on samples from patients with diffuse large B-cell lymphoma and identified a small cluster of genes whose expression is highly predictive of survival. [26] The shaving algorithm can be summarized in the following steps: 1) Build the expression matrix Ξ for all genes, and center each row of Ξ to zero mean. 2) Compute the leading principal component of rows of Ξ . 3) “Shave off” the proportion (typically 10%) of the genes having smallest absolute inner product with leading principal component. 4) Repeat the second and third steps until only one remains. 5) Estimate the optimal cluster size k (for maximum gap). 6) Orthogonalize each row of Ξ with respect to the average gene in the size-optimized gene cluster. 7) Repeat the first five steps with the

orthogonalized data to find the second optimal cluster. This process is continued until a maximum of M clusters are reached (M is chosen a priori).

There are two varieties to the original shaving algorithm: supervised (partially supervised), or unsupervised. In supervised and partially supervised shaving, available information about genes and samples (outcome measure, known properties of genes or sample, or any other a priori information) can be used to label their data as a means to supervise the clustering process and ensuring meaningful groupings. Gene shaving also allows genes to belong to more than one cluster. These two properties (the ability to supervise and multiple groupings for the same gene) make gene shaving different from most hierarchical clustering and other widely used methods for analyzing gene expression data.

The most prominent advantage of the shaving methods proposed by Hastie, Tibshirani et al. were best expressed by the authors themselves: “(shaving methods) search for clusters of genes showing both high variation across the samples, and coherence (correlation) across the genes. Both of these aspects are important and cannot be captured by simple clustering of genes or setting threshold of individual genes based on the variation over samples”. [26]

The drawback of this method is the computational intensity of the algorithm. The shaving process requires repeated computation of the largest principal component of a large set of variables. Some variant approaches should consider replacement algorithms for SVD that are less computationally intensive.

Support Vector Machine (SVM): SVM is a supervised machine learning technique in the sense that vectors are classified with respect to known reference vectors. “SVM solve the problem by mapping the gene-expression vectors from expression space into a higher-dimensional feature space, in which distance is measured using a mathematical function known as a kernel function, and the data can then be separated into two classes”. [17] Gene expression vectors can be thought of as points in an n -dimensional space. For microarray analysis, sets of genes are identified that represent a target pattern of gene expression. The SVM is then trained to discriminate between the data points for that pattern (positive points in the feature space) and other data points that do not show that pattern (negative points in the feature space). “With an appropriately chosen feature space of sufficient dimensionality, any consistent training set can be made separable”. [27] SVM is a linear technique in that it uses hyperplanes as separating surfaces between positive and negative points in the feature space. Specifically, SVM chooses the hyperplane that provides maximum margin between the plane surface and the positive and negative points. This feature provides a mechanism to avoid over fitting. Once the separating hyperplanes have been selected, “the decision function for classifying points with respect to the hyperplane only involves dot product between points in the feature space”, which carries a low computational burden. [27]

Because the linear boundary in the feature space maps to a nonlinear boundary in the gene expression space, SVM can be considered as a nonlinear separation technique. The important advantage of SVM is that it offers a possibility to train generalizable, nonlinear classifiers in high-dimensional space using a small training set. For large training sets, SVM typically selects a small support set that is necessary for designing the classifier, thereby, minimizing the computational requirements during testing. Furey, et al. [28] praises SVM as follows: “It (SVM) has demonstrated the ability to not only correctly separate entities into appropriate classes, but also identify instances whose established classification is not supported by the data. It performs well in multiple biological analyses, having remarkable robust performance with respect to sparse and noisy data”.

One of the drawbacks of SVM is its sensitivity to the choice of a kernel function, parameters, and penalties. For instance, if the kernel function is not appropriately chosen for the training data, SVM may not be able to find a separating hyperplane in feature space. [27] Choosing the best kernel function, parameters and penalties for the SVM algorithm can be difficult in many cases. Because of the sensitivity of the algorithm to these choices, different choices often yield completely different classifications. As a result, “it is necessary to successively increase kernel complexity until an appropriate (biologically sound) classification is achieved”. [17] The ad hoc nature of the penalty term (error penalty), the computational complexity of the training procedure (a quadratic minimization problem), and risk of over fitting when using larger hidden layers are further drawbacks of this method.

Other Techniques in Microarray Data Analysis: Sasik et al. [23] have presented the *percolation clustering* approach to clustering of gene expression patterns base on the mutual connectivity of the patterns. Unlike SOM or *k*-means which force gene expression data into a fixed number of predetermined clustering structures, this approach is to reveal the natural tendency of the data to cluster, in analogy to the physical phenomenon of percolation.

GA/KNN is another algorithm described by Li, et al [29]. “This approach combines a *genetic algorithm* (GA) and the *k*-Nearest Neighbor (KNN) method to identify genes that can jointly discriminate between different classes of samples. The GA/KNN is a supervised stochastic pattern recognition method. It is capable of selecting a subset of predictive genes from a set of large noisy data for sample classification.” [29]

A large body of research has been conducted on the application of various types of artificial neural networks applied to the problems at hand. These techniques have been reviewed in [30] and will not be repeated in this paper.

Conclusion

A number of pattern recognition techniques have been applied to analyze microarray expression data [2,17,1]. The simplest category of these techniques is that of individual gene-based analysis such as fold approach, t-test rule, and Bayesian framework. More sophisticated techniques include (but are not limited) cluster analysis methods and SVD technique. The hypothesis (hope) behind using clustering methods is that genes in a cluster share some common function or regulatory elements. Although, this may be true in biologically sound division of data, it only holds partially true (or not at all) in the clustering produced by a purely mathematical technique. The success of these algorithms should be evaluated based on how biologically sound (or relevant) the clustering is. On this road, we believe that algorithms that allow the inclusion of a priori biological knowledge show higher potential.

This review represents only a small part of the research being conducted in the area, and only is meant as a complementary/continuation of the survey that others have conducted in this area [17,1]. It should in no way be taken as a complete survey of all algorithms. For the reason of limited space, some significant developments in the area had to be left out. Furthermore, new techniques and algorithm are being proposed for microarray data analysis on a daily basis making survey articles such as this highly time-dependent.

References

1. Szabo A. *et al* Variable selection and pattern recognition with gene expression data generated by the microarray technology. *Math Biosci* (2002) 176(1), 71-98
2. Baldi, P. *et al*. A Bayesian framework for the analysis of microarray expression data: regularized t-test and statistical inference of gene changes (2001) *Bioinformatics*, 17(6) , 509-519
3. Pan, W. A comparative review of statistical methods for discovering differentially expressed genes in replicated microarray experiments (2002) *Bioinformatics* 18(4), 546-54
4. Mallat, S. G. (1999) *A Wavelet Tour of Signal Processing* (Academic, San Diego), second edition
5. Anderson, T. W. (1984) *Introduction to Multivariate Statistical Analysis* (Wiley, NY) second edition
6. Alter, O. *et al*. Singular Value Decomposition for Genome-wide Expression Data Processing and Modeling (2000) *Proc. Natl. Acad. Sci. USA* 97, 10101-10106
7. O. Alter, P.O. Brown and D. Botstein , Singular value decomposition for genome-wide expression data processing and modeling. *Proc Natl Acad Sci USA* 97 (2000), pp. 10101-10106
8. N.S. Holter, A. Maritan, M. Cieplak, N.V. Fedoroff and J.R. Banavar , Dynamic modeling of gene expression data. *Proc Natl Acad Sci USA* 98 (2001), pp. 1693-1698
9. N.S. Holter, M. Mitra, A. Maritan, M. Cieplak, J.R. Banavar and N.V. Fedoroff , Fundamental patterns underlying gene expression profiles: simplicity from complexity. *Proc Natl Acad Sci USA* 97 (2000), pp. 8409-8414

10. Raychaudhuri S, Stuart JM, Altman RB. Principal components analysis to summarize microarray experiments: application to sporulation time series. *Pac Symp Biocomput* 2000, 455-466.
11. Wu, C., Berry, M., Shivakumar, S. and Mcarty, J. 1995. Neural networks for full-scale protein sequence classification: sequence encoding with singular value decomposition. *Machine Learning*. 21: 177-193.
12. N. Friedman, M. Linial, I. Nachman and D. Pe'er , Using Bayesian networks to analyze expression data. *J Comput Biol* 7 (2000), pp. 601-620
13. A. Drawid and M. Gerstein , A Bayesian system integrating expression data with sequence patterns for localizing proteins: comprehensive application to the yeast genome. *J Mol Biol* 301 (2000), pp. 1059-1075
14. D.K. Gifford , Blazing pathways through genetic mountains. *Science* 293 (2001), pp. 2049-2051
15. Hartemink AJ, Gifford DK, Jaakkola TS, Young RA. Using graphical models and genomic expression data to statistically validate models of genetic regulatory networks. *Pac Symp Biocomput* 2001, 422-433
16. D. Pe'er, A. Regev, G. Elidan and N. Friedman , Inferring subnetworks from perturbed expression profiles. *Bioinformatics* 17 Suppl 1 (2001), pp. S215-S224
17. Quackenbush, J. Computational Analysis of MicroArray Data (2001) *Nature Genetics* 2, 418-427
18. Tamayo, P. *et al.* Interpreting patterns of gene expression with self-organizing maps: methods and application to hematopoietic differentiation (1999) *Proc. Natl. Acad. Sci. USA* 96, 2907-2912
19. Bozinov D Unsupervised technique for robust target separation and analysis of DNA microarray spots through adaptive pixel clustering (2002) *Bioinformatics* 18(5), 747-56.
20. Wen, X.*et al.* Large-scale Temporal Gene Expression Mapping of Central Nervous System Development (1998) *Proc. Natl. Acad. Sci USA* 95, 334-339
21. Eisen, P. T. *et al* Cluster Analysis and Display of Genome-Wide Expression *Patterns* (1998) *Proc. Natl. Acad. Sci. USA* 95, 14863-14868
22. Alon, U.*et al.* Broad Patterns of gene Expression Revealed by Clustering Analysis of Tumor and Normal Colon Tissues Probed by Oligonucleotide Arrays (1999) *Proc. Natl. Acad. Sci USA* 96, 6745-6750
23. Sasik, R. *et al.* Percolation Clustering: A Novel Approach to the Clustering of Gene Expression Patterns in Dictyostelium Development (2001) *PSB Proceedings* 6, 335-347
24. Baldi, P. On the convergence of a clustering algorithm for protein-coding regions in microbial genomes (2000) *Bioinformatics* 16, 367-371
25. Ghosh, D. Mixture modeling of gene expression data from microarray experiments *Bioinformatics* (2001) 18(2), 275-286
26. Hastie, T. *et al.* "Gene Shaving" as a Method for Identifying Distinct Sets of Genes with Similar Expression Pattern (2000) *Genome Biology* 1, 1-21
27. Brown, M. P. S. *et al* Knowledge-based analysis of microarray gene expression data by using support vector machines (2000) *Proc. Natl. Acad. Sci. USA* 97, 262-267
28. Furey, T.S *et al.*, Support vector machine classification and validation of cancer tissue samples using microarray expression data (2000) *Bioinformatics*, 16, 906-914
29. Li, *et al* Gene selection for sample classification based on gene expression data: study of sensitivity to choice of parameters of the GA/KNN method (2001) *Bioinformatics* 17(12), 1131-1142
30. F. Valafar. Neural Network Applications in Biological Sequencing. *Proceedings of the 2003 International Conference on Mathematics and Engineering Techniques in Medicine and Biological Sciences.* (METMBS'03) June 24-27, 2002. In print.