Comparison of Two Clustering Methods on Yeast Data

Paula Silvonen
VTT Information Technology
P.O.Box 1201
02044 VTT, FINLAND
paula.silvonen@vtt.fi

ABSTRACT
There is a huge amount of data to be processed in result of microarray analysis. A common way for finding descriptive characteristics from the data is clustering. However, numerous algorithms have been proposed for the task without specific instructions for their usage.

We experimented on yeast genome data with a number of algorithm implementations and selected two different approaches for further evaluation. We evaluated the significance of proper preprocessing of the data and the goodness of the clustering solutions in the light of the cluster similarities.

Categories and Subject Descriptors
I.5.3 [**Pattern Recognition**]: Clustering; J.3 [**Computer Applications**]: Life and Medical Sciences

General Terms
Clustering

Keywords
Clustering, Microarray Analysis

1. INTRODUCTION
Microarray technology has made it possible to simultaneously interrogate the activity of thousands of genes. The production and use of microarrays is still extremely demanding: producing them involves many steps which influence the outcome of the process; the computation and statistical analysis of the massive amount of data requires efficient methods to avoid combinatorial explosion. Data mining techniques have become widely used also on this field.

Clustering is the unsupervised classification of patterns (observations, data items, or feature vectors) into groups (clusters) [3]. The clustering problem has been found useful in many disciplines as one of the steps in exploratory data analysis. However, clustering is a difficult problem combinatorially, the results are not easily validated, and the variability of approaches furthermore complicates the comparison of the solutions.

In this experiment we evaluated some clustering methods to find out what kind of differences there are in the results they produce. The data involved was the whole yeast genome, which had been stress induced by salt. The data was gathered and used originally by Causton et al. [1]

We evaluated two hierarchical methods, one top-down and one bottom-up method, which also had different approaches to determine the goodness of the clustering solution. We analysed and compared the results with cluster similarity measures.

2. DATA AND NORMALIZATION
Yeast (Saccharomyces cerevisiae) has been the source of many microarray experiments. The data we used was a part of an experiment where genome-wide expression analysis was carried out to explore how gene expression in Saccharomyces cerevisiae is remodelled in response to various changes in extracellular environment, including changes in temperature, oxidation, nutrients, pH, and osmolarity [1]. Our data was stress induced by salt.

An overnight culture of strain ATCC-201388 was grown in YPD and used to inoculate seven (2 l) flasks containing 200 ml of YPD. These were grown to $OD_600 = 0.6 - 0.8(\sim 1.8 - 2.410^7 cell/ml)$ at 30°C. Sodium chloride was added to a final concentration of 1.0 M. Cells were harvested by centrifugation 15, 30, 45, 60, 90, and 120 min after the addition of sodium chloride and from duplicate cultures immediately before the addition of sodium chloride (time 0 min). The chip type in this experiment was Affymetrix (Santa Clara, CA) YE6100 gene chips.

The data had been prenormalized so that the expression profiles corresponding to individual time points were scaled to a common reference profile based on fluorescent intensities of 5 DNA controls that were added after the preparation of total RNA. A scaling ratio between the sample chip and the reference chip was calculated by using the following formulas:

$$r = e\frac{\sum_{i=1}^{n} \ln \alpha_i - \ln \alpha}{n}$$

where n is the total number of controls, α is the reference sample and a_i is the sample to be normalized. The rescaled value for sample i was then calculated as:

$$\alpha_i^2 = a_i / r$$

where a_i is the sample to be normalized, and α_i^2 is the rescaled value. In addition, values below 0.01 were set to 0.01.

No replicates exist for this data set, so we did not need to calculate means, medians or any other representations for data groups.

For the experiment we chose to use two values for each time point per gene: the normalized value, and the relative change in value per time point. If the gene was absent in the time point the relative change was set to 1. Genes absent in each time point were discarded. This left us with 5316 genes. In the original paper only genes with fold three change were kept. We, however, chose not to make that restriction because of the relatively reliable nature of the Affymetrix chips.

Later in discussion we will point out different approaches for preprocessing, which might have improved the clustering results.

3. CLUSTERING METHODS

Six different algorithms with some variations were tested, and afterwards two of them were chosen for further evaluations. The methods we tested were different types of hierarchical clustering, k-means, Expectation Maximization, and graph theoretic approaches. We chose hierarchical partitional clustering and hierarchical agglomerative clustering, both implemented in gCLUTO [4]. We selected these methods in part because the usage of the same tool in both experiments made the comparison of the results somewhat easier. Also our familiarity with the tool played a role in the selection. The reason for picking the two methods from gCLUTO's selection was that the writers claimed them to perform best in their own experiments.

In gCLUTO there were also different similarity measures to choose from. The results and methods described here are produced with cosine similarity $\cos(g_i, g_j) = \frac{g_i^t g_j}{|g_i||g_j|}$.

3.1 Hierarchical Partitional (top-down) Clustering

In hierarchical partitional clustering a random pair of genes is first selected from the collection to act as the seeds of two clusters [6]. For each point, its similarity to seeds is computed, and it is assigned to the cluster corresponding to its most similar seed. This leads to the initial two-way clustering. Clustering is then repeatedly refined so that it optimizes the desired clustering criterion function. This kind of algorithm is often called incremental. During the execution, each move directly optimizes the particular criterion function, so the refinement strategy always converges to a local minimum. The greedy refinement algorithm does not guarantee convergence to a global minimum. The local minimum solution depends on the particular set of seed points that were selected during the initial clustering. The overall process is repeated a number of times. In this approach N different clustering solutions (i.e., initial clustering followed by cluster refinement) are computed, and the one that achieves the best value for the particular criterion function is kept.

Complexity of the algorithm is $O(n \log n)$, where n is the number of genes in the dataset. The complexity can be regarded as relatively small compared to many other approaches.

gCLUTO provides with eight different criterion functions to choose from [5]. The criterion function we decided to evaluate was I_2. In the criterion function each cluster is represented by its centroid vector

$$C_A = \frac{G_A}{|A|}$$

where A is a set of genes and G_A is the composite vector

$$G_A = \sum_{g \in A} g$$

The goal is to find the clustering solution that maximizes the similarity between each gene and the centroid of the cluster that it is assigned to. With cosine function to measure the similarity between a gene and a centroid, the criterion function becomes the following:

$$maximize I_2 = \sum_{r=1}^{k} \sum_{g_i \in S_r} \cos(g_i, C_r)$$

which can be rewritten as follows

$$I_2 = \sum_{r=1}^{k} \sum_{g_i \in S_r} \frac{g_i^t C_r}{|C_r|} = \sum_{r=1}^{k} \frac{G_r^t C_r}{|C_r|} = \sum_{r=1}^{k} \frac{G_r^t G_r}{|G_r|} = \sum_{r=1}^{k} |G_r|$$

where S is the set of genes, C is a centroid vector, and G is a composite vector. The term G_r is the square-root of the pairwise similarity between all the genes in S_r. The criterion function tends to emphasize the importance of clusters whose genes have smaller pairwise similarities compared to clusters with higher pair-wise similarities. Among the current k clusters, the algorithm selects the cluster which leads to the k + 1 clustering solution that optimizes the value of the particular criterion function (among the different k choices). The criterion function makes this approach quite similar to k-means.

3.2 Hierarchical Agglomerative (bottom-up) Clustering

In this approach each gene is initially assigned to its own cluster [6]. After that pairs of clusters are repeatedly selected and merged until there is the desired number of clusters. The clusters to be merged are selected according to their similarity.

The complexity of the algorithm is $O(n^2 \log n)$, where n is the number of genes in the dataset. The complexity is not good, and leads to lengthy execution times. However, our data set was quite small and did not thus present problems. Zhao and Karypis [6] proposed a new approach in which they introduced intermediate clusters obtained by partitional clustering algorithms to constrain the space over which agglomeration decisions are made. This kind of method seemed to lead to better solutions than the traditional agglomerative algorithms.

In our experiment we used the UPGMA scheme (group average) to measure the similarity of two clusters as the average of the pairwise similarity of the genes from each cluster [5]:

$$sim_{UPGMA}(S_i, S_j) = \frac{1}{n_i n_j} \sum_{g_i \in S_i, g_j \in S_j} \cos(g_i, g_j) = \frac{G_i^t G_j}{n_i n_j}$$

The UPGMA scheme should overcome some problems that are present in single- and complete-link approaches, which either base their decisions on limited amount of information or assume that all the genes in the cluster are very similar to each other [5]. In our experiment, however, the latter problem was evident.

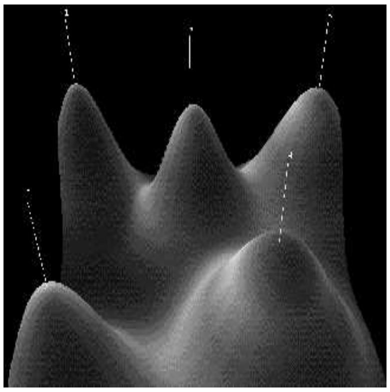

Figure 1: Mountain Visualization of partitional clustering solution

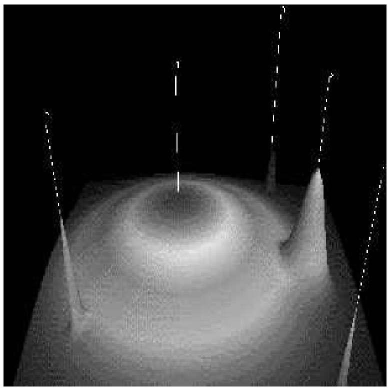

Figure 2: Mountain Visualization of agglomerative clustering solution

4. RESULTS

Both approaches were evaluated with number of clusters being five, because with the partitional method that seemed to give best results. The results of the two methods were very different both in their characteristics and cluster sizes. Tables 1 and 2 present the solution statistics of the approaches.

The agglomerative approach seemed to give better results in terms of internal cluster similarity and external cluster dissimilarity, but the clusters with good similarity measures were only a few genes in size - maybe too small to be interesting. It seemed that the UPGMA similarity measure was very conservative; if some genes with highly similar characteristics were found, they were clustered together, and no different ones would be added.

The partitional method gave clusters of quite similar sizes. The cluster similarity measures were quite good, although not as good as the ones produced with the agglomerative algorithm. According to the cluster statistics, with this method all the clusters were quite equally good compared to the variation of the clusters obtained with the agglomerative method.

The interesting cluster characteristics for the agglomerative method was that there was a cluster of genes where from time point 30 to time point 45 there was a huge growth in value. Interestingly enough, the partitional method found clusters from all other time points, where there was almost as huge a growth in value.

Figures 1 and 2 present mountain visualizations of the clustering solutions. The visualization uses Multidimensional Scaling (MDS), which maps the original high-dimesional dataset and the displayed two-dimensional data so that the data's distortion is minimized as it is mapped to the plane [4]. MDS is an algorithm that takes as input a list of high-dimensional vertices and outputs a list of lower-dimensional vertices. In gCLUTOs implementation, the cluster midpoints are used as input and the output consists of two-dimensional points, which are used to place peaks on the plane of the visualization. MDS evaluates a particular mapping using a stress function that calculates the mappings distortion using a sum-of-squared-error calculation. The implementation was made so that relationships found between clusters were based on the same method of comparison as used for deriving the internal contents of the clusters.

In figure 3 are the matrix visualizations of the two solutions, the partitional clustering on the left and the agglomerative clustering on the right. Each row represents a gene, and black lines divide the clusters. Dark colour indicates high values.

Table 1: Clustering results of the partitional method

Cluster	Size	ISim	ISdev	ESim	ESdev	Descriptive	Descriminating
0	793	0.87	0.04	0.69	0.10	Time-45-60	Time-45-60
1	656	0.87	0.04	0.71	0.10	Time-60-90	Time-60-90
2	1288	0.88	0.05	0.73	0.10	Time-15-30	Time-15-30
3	745	0.87	0.05	0.74	0.09	Time-0-15	Time-0-15
4	1834	0.88	0.06	0.78	0.07	Time-120	Time-15-30

Table 2: Clustering results of the agglomerative method

Cluster	Size	ISim	ISdev	ESim	ESdev	Descriptive	Descriminating
0	3	1.00	0.00	0.26	0.00	Time-120	Time-120
1	4	0.91	0.05	0.38	0.03	Time-15	Time-15
2	4	0.90	0.05	0.44	0.14	Time-45	Time-45
3	83	0.87	0.05	0.588	0.10	Time-30-45	Time-30-45
4	5222	0.777	0.09	0.56	0.11	Time-15-30	Time-30-45

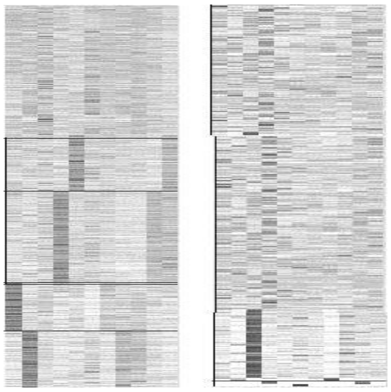

Figure 3: Matrix Visualizations of both clustering solutions

5. DISCUSSION

Our preprocessing scheme was probably not the best possible one. The normalized values of absent genes should have been replaced with some indication of gene's absence to avoid their influencing the results. Replacing the relative growth in value with a log ratio might also have been a good option.

The cosine similarity measure is somewhat problematic, as it does not consider the length of the vectors in determining their distance. The approach is quite susceptible to noise.

To further evaluate the solutions, different cluster evaluation schemes could be implemented. Useful would be validity indices such as the modified Hubert Γ statistic, Dunn and Dunn-like indices, or Davies-Bouldin index [2]. Also getting more background knowledge on the genes in each cluster would yield more information on the clusters.

Even though the data was only a part of the Causton set [1], the comparison between their clustering results and ours would be interesting.

6. CONCLUSIONS

We evaluated two different clustering methods on yeast data. The data was whole yeast genome, which had been stress induced by salt. The clustering algorithms evaluated were two hierarchical methods with different criterion functions for cluster similarity. The results differed both in cluster sizes and cluster characteristics. The results were compared to each other and validated with internal cluster similarity and external dissimilarity. More or different preprocessing might have given better results.

7. REFERENCES

[1] H. Causton, B. Ren, S. Koh, C. Harbison, E. Kanin, E. Jennings, T. Lee, H. True, E. Lander, and R. Young. Remodeling of yeast genome expression in response to environmental changes. *Mol Biol Cell*, 12(2):323–337, February 2001.

[2] M. Halkidi, Y. Batistakis, and M. Vazirgiannis. On clustering validation techniques. *Journal of Intelligent Information Systems*, 17(2/3):107–145, 2001.

[3] A. Jain, M. Murty, and P. Flynn. Data clustering: A review. *ACM Computing Surveys*, 31(3):264–323, September 1999.

[4] M. Rasmussen and G. Karypis. Gcluto: An interactive clustering, visualization, and analysis system. Technical report, University of Minnesota, Department of Computer Science and Engineering, 2004.

[5] Y. Zhao and G. Karypis. Criterion functions for document clustering - experiments and analysis. Technical report, University of Minnesota, Department of Computer Science and Engineering / Army HPC Research Center, 2001.

[6] Y. Zhao and G. Karypis. Evaluation of hierarchical clustering algorithms for document datasets. Technical report, University of Minnesota, Department of Computer Science and Engineering, 2002.