Analyzing the *Escherichia coli* Gene Expression Data by a Multilayer Adjusted Tree Organizing Map

Ning Wei
School of Computer Science
The University of Oklahoma
ningwei@ou.edu

Le Gruenwald
School of Computer Science
The University of Oklahoma
ggruenwald@ou.edu

Tyrrell Conway
Department of Botany and Microbiology
The University of Oklahoma
tconway@ou.edu

Abstract

Using the DNA microarray technology, biologists have thousands of array data available. Discovering the function relations between genes and their involvements in biological processes depends on the ability to efficiently process and quantitatively analyze large amounts of array data. Clustering algorithms are among the popular tools that can be used to help biologists achieve their goals. Although some existing research projects employed clustering algorithms on biological data, none of them has examined the Escherichia coli (E. coli) gene expression data. This paper proposes a clustering algorithm called Multilayer Adjusted Tree Organizing Map (MATOM) to analyze the E. coli gene expression data. In a semi-supervised manner, MATOM constructs a multilayer map, and at the same time, removes noise data in the previously trained maps in order to improve the training process. This paper then presents the clustering results produced by MATOM and other existing clustering algorithms using the E. coli gene expression data, and a new evaluation method to assess them. The results show that MATOM performs the best in terms of percentage of genes that are clustered correctly.

1. Introduction

1.1. Problem Statement

Functional genomics aims to reveal the biological functions of an individual gene and its cooperative roles on a genome-wide scale. The DNA microarray [14] is a powerful experimental tool for extracting functional information from the genome. Microarray analysis [4]] is one of the latest breakthroughs in experimental molecular biology, which allows the monitoring of gene expression for tens of thousands of genes in parallel and produces huge amounts of valuable data. With the implementation of the DNA microarray technology, biologists will increasingly depend on the ability to efficiently process and quantitatively analyze large amounts of data to discover functional relations between genes and their involvement in important biological processes.

Clustering algorithms are widely employed in analyzing DNA microarray data ([2] [5] [8]). Clustering gene expression data is to discover unknown genes with already identified genes in the same cluster and also to provide clues to their functions. When genes with similar expression profiles involve in similar biological processes, clustering algorithms can group them together. A gene expression is a value of a gene derived by a biological process. A gene expression profile refers to the set of the expression values for a single gene across many experimental conditions. This paper focuses on clustering the *E. coli* gene expression data in order to identify unknown genes involved in the Acid Tolerance Response (ATR) [13] of *E. coli*, which are regulated by the regulator gene *yhiX*. There are 4290 *E. coli* gene expression data under eight different conditions from the microarray experiments.

The large range of gene expression values has a negative influence on the clustering process; therefore the normalized values, instead of the original values, will be used as data sources in order to discover more accurate clustering results. We normalize a gene expression value by dividing it by the value of the experimental condition $Wt7\text{-}4(2)pct : \frac{OneOfConditions}{\mathrm{Wt7 - 4(2)pct}}$ where *Wt7-4(2)pct* is a natural and original condition.

The normalized gene expression values show the trend of how the gene expression values changed under different conditions. Therefore, the normalized expression values are more meaningful for biological processes. In the rest of this paper, the terms profile and gene expression will be used to mean normalized profile and normalized gene expression, respectively.

The *E. coli* gene expression data have the following special characteristics that must be taken into consideration when clustering them:

- In the *E. coli* gene expression data, there are less than 1.5% genes that we may be interested in. That means that there are a high percentage of genes that are not regulated by *yhiX*, called *noise genes*. The existing clustering algorithms, such as CLICK [12], hierarchical clustering [7] and K-means [1], cannot avoid the negative influence of the high percentage of noise data during the clustering process. Noise data is not what we are interested in. Although some existing algorithms, such as SOM [11], GNG [6], and FLVQ [10], are not very sensitive to noise data based on their theoretical analyses, the exceptionally high percentage of noise data in the *E. coli* gene expression data still has a negative effect on their performance.
- The expression values of noise data have similar values under different experimental conditions, while the expression values of the potential target genes increase or decrease in the biological processes according to the changes in the experimental conditions. In some applications, this kind of data may be treated as outlier data by clustering algorithms. However, genes with this kind of profiles are what a clustering algorithm needs to cluster with target genes.
- There are 13 genes that are identified as ATR genes of *E. coli*. Clustering algorithms need to discover other unknown genes that have the expression profiles similar to those of the target genes.

The existing data clustering algorithms (CLICK, Hierarchical Clustering, K-means and SOM) do not use the data set similar to ours for testing and do not consider the above characteristics of the *E. coli* gene expression data though they perform well when analyzing other biological data. The objective of this paper is to propose a new clustering algorithm to analyze the *E. coli* gene expression data by taking all their characteristics into consideration. In order to compare the proposed algorithm with the six existing clustering algorithms, CLICK, Hierarchical Clustering, K-means, SOM, GNG, and FLVQ, this paper then proposes an evaluation model to assess their performance on analyzing the *E. coli* gene expression data.

The rest of this paper is organized as follows. Section 2 presents the proposed algorithm called Multilayer Adjusted Tree Organizing Map (MATOM). Section 3 proposes an evaluation model to compare MATOM with the existing clustering algorithms. Finally, Section 4 provides conclusions and future research.

2. The Multilayer Adjusted Tree Organizing Map Algorithm

2.1 Introduction

The Multilayer Adjusted Tree Organizing Map (MATOM) algorithm proposed in this section is a semi-supervised algorithm based on the neural network model. MATOM consists of multi-layers of maps. A map is a particular neural network, which can define a lattice of connections of neural nodes and a shape of a map in the multidimensional space [11]. In the following discussion, the term "map layer" and "map" will be used interchangeably. Using the batch training algorithm [11], MATOM builds a multilayer neural network and a relation tree of the resulting clusters. In a semi-supervised manner, MATOM tracks the target data and deletes the map nodes that contain only noise data in order to save training time on clustering noise data. The target genes for the *E. coli* gene expression data are already known.

Below we identify the existing clustering algorithms' deficiencies and address them in MATOM:

- Hierarchical clustering produces only the relation tree of all data elements in the whole dataset, not of only the resulting clusters. The limitation of the hierarchical clustering algorithm is that it only produces the dendrogram tree, not the final resulting clusters directly even though they can be produced by cutting off the tree. The relations of clusters are more helpful than those of single data elements because they can be used for merging or splitting the resulting clusters. A dendrogram tree produced by the hierarchical clustering algorithm does not provide such information. MATOM produces the final clustering results directly while providing the relation tree of clusters.
- K-means requires the number of clusters, K, to be predefined. With different values of K, K-means produces different clustering results. The algorithm does not provide any information of how to choose a correct value for K. If a correct value of K is not available, it is not easy to achieve the desired clusters. The same problem also exists in FLVQ. On the contrary, MATOM does not require the number of clusters to be specified in advance. The relation tree of clusters provided by MATOM can be employed to adjust the algorithm parameters for use in its next executions.
- GNG adapts the outlier data to the growing map structure effectively. Therefore, GNG has a high percentage of accuracy of clustering results and a low percentage of false genes in the clusters. However, GNG uses a huge amount of execution time because it inserts only one new node into the map during each training epoch. MATOM, with the multilayer neural network model, avoids the influence of noise data to save time on training the map nodes of noise data.
- The advantage of SOM is the use of the Kohonen updating rule [11], which trains a map efficiently and produces accurate clustering results. However, SOM wastes time to train noise data resulting in a negative impact on the correct training process. It is not easy to adjust the map size in SOM because SOM

does not provide the relations of the map nodes directly. The termination of SOM is not based on any optimization process.

- The existing clustering algorithms do not allow users to determine the size of the final clusters. This size is useful for biologists because they need to screen a limited number of target genes from huge amounts of data to study them in the next step of biological analysis. MATOM provides a function to allow users to determine the size of the final clusters and uses this to decide when it should terminate its execution.

2.2. The MATOM Algorithm

2.2.1. Basic Ideas

During the training process, MATOM trains the map nodes that contain the target genes, employing the batch training algorithm [11]. MATOM is terminated when the desired clusters are achieved by testing the size of the final clusters. Using Best Match Unit, MATOM finds the centroids of the map nodes and uses them to construct the relation tree's leaves. In other words, MATOM constructs each new layer of the maps on the nodes that contain the target genes by tracking the sample genes.

2.2.2. The Details of the MATOM Algorithm

STEP 1: Require users to set the map size and the final clusters' size (the map size is the number of nodes in the map and the final clusters' size is the number of genes in the final clusters). Initialize the weight vectors for the map nodes. Each map node has a weight vector. The weights of the nodes in the first map (or also called the first map layer) are randomly initialized by choosing some data elements from the data set, m_i , as $<m_{i1}, m_{i2}, \ldots m_{in}>$, where m_i is the initial weight vector of map node i, and n is the number of attributes of the input data elements. Usually, the map size is chosen as 2x2. The number of training times (epochs) is usually set to *max {1, 4*munit/dlength},* where *munit* is the number of map nodes and *dlength* is the length of the data set [Juha 99]. MATOM chooses Gaussian Function as the neighborhood function, and hexagonal lattice as the map lattice [Juha 99].

STEP 2: Enter data and related information from the data set to the algorithm. Import the whole input data set into the training data set and sample data set into the reference data set.

STEP 3: Determine the winner node or best match unit (BMU). BMU has the Euclidian shortest (minimum) distance to its data elements x: $||x-m_b||=min\ \{||x-m_i||\}$, where m_b is the BMU, m_i is a weight vector of map node i, x is a vector representing a data element, the distance between x and m_i is calculated using the Euclidean distance function, $\|x-m_i\| = \sqrt{\sum_{j=1}^{n}(x_j - m_{ij})^2}$.

STEP 4: Update the weights within the neighborhood of the map nodes using the batch training rule. Each BMU (also called the centroid of a map node) is updated using the following batch training rule with the predefined training time t, $m_j(t+1) = \frac{\sum_{i=1}^{n} h_{im(j)}(t)x_i}{\sum_{i=1}^{n} h_{im(j)}(t)}$, where h_{im} is the neighborhood function centered on the winner units, $h_{im(j)}(t) = \exp\left(-\frac{\|m_j - m_i\|^2}{2\sigma^2(t)}\right)$, where $\sigma(t)$ is the neighborhood radius rate at the training time t (a neighborhood radius rate is a range of a radius between a winner node and its neighbors), m_j and m_i are the positions of BMU b and its neighbor node i on this map layer. $\sigma(t)$ is the neighborhood radius rate *[radius_ini, radius_fin]* at time t in which the initial radius (an optional parameter) is *radius_ini*= $\max\{1, mapsize/2\}$ (*radius_ini* can also be defined by users) and the final radius, *radius_fin*, is usually set as 1. Record the centroids of this map layer for constructing the relation tree later.

STEP 5: Build a new layer of the map and delete the noise data in the previously trained map. Track *critical nodes* which are nodes containing the sample data elements. Construct a new layer of the map from the previous critical nodes.

STEP 6: Construct a relation tree of the map nodes using the centroids of the maps. Build this tree structure for each layer of the multilayer maps using the single-link algorithm as follows: for each layer of the map, find the two nodes x_i and x_j that have the shortest distance $d_s = \min_{i,j}\{\|x_i - x_j\|\}$, store them as the two leaf nodes in the relation tree whose distance in the tree is corresponding to their distance in the map, then merge x_i and x_j into one node x_k representing their average value, add x_k to the nodes to be examined, and remove x_i and x_j from the nodes to be examined, then repeat the entire process until no new nodes are added to the relation tree. Each level of the tree is drawn from the nodes in one layer of the multilayer maps. The leaves of the tree represent the nodes of the maps.

STEP 7: Terminate the algorithm if the size of the critical nodes reaches the one set in STEP 1; otherwise, repeat STEPs 3-7.

2.3. The Advantages and Disadvantages of MATOM

2.3.1. Advantages

- Through the multilayer neural network structure, MATOM deletes noise data by building a new layer from the critical nodes as well as reducing the percentage of noise data in new map layers. MATOM is robust and not sensitive to noise data because the structure of the multilayer maps allows MATOM to delete noise data at the beginning of the training process. MATOM can recognize the correct noise data by tracking critical nodes when the percentage of noise data is much higher than that of target data in the data set. There is no existing clustering algorithm that deletes noise data in this way. In a semi-supervised manner, MATOM would not miss the potentially interested data elements by tracking the critical nodes.
- MATOM produces a relation tree of clusters that users can employ to adjust the algorithm parameters for use in its next executions, such as the size of the final clusters and the initial map size. This prevents users from repeatedly running MATOM blindly. The hierarchical clustering algorithm [9] draws a dendrogram tree to provide the relations of data elements in the whole data set. However, the relations of clusters provided by MATOM help users adjust the parameters of clustering training processes directly. Users obtain the degree of similarities among clusters through the distance of the clusters provided by the relation tree. This can be used to determine whether clusters should be merged or split. If there are two clusters that have a short distance in the relation tree, they can be merged together as one cluster for studying the profiles of genes in them. This helps biologists identify those genes' behaviors in similar biological processes. On the other hand, users can adjust the size of the final clusters for the next clustering processes. For instance, if some critical nodes in the same level of the relation tree are far away from each other (i.e. the distance between them is long), the size of the final clusters can be adjusted to be smaller. By doing that, users can know the size of the final clusters that they should use in the next executions of MATOM.
- Employing the batch training rule, the weights of the map nodes in MATOM are independent of the order of the input sequence, which leads to more accurate clustering results. MATOM also reduces the percentage of false genes in the final results by avoiding the negative effect of the order of the input data sequence on the training process. In contrast, classical SOM using the sequential training rule [11] provides the final clusters that contain a high percentage of false genes because the sequential training process is sensitive to the order of the input data sequence.
- MATOM provides a function to allow users to determine the size of the final clusters directly. This allows users to terminate MATOM when they desire. The existing clustering algorithms do not provide this kind of terminating functions.

2.3.2. Disadvantages

- MATOM requires users to predefine the size of the final clusters. Before analyzing data, users usually do not have any knowledge about this parameter. However, users can determine and adjust this parameter based on the relation tree of clusters provided by MATOM after the first time running the algorithm. Although K-means provides an error function determining the distance between the centroids of the final clusters and the data elements in the final clusters, which can be used to terminate K-means, the error value cannot produce any direct clue to adjust the value of K, which is a critical parameter of K-means. FLVQ has the same problem. Based on the neural network model, SOM and GNG cannot present the relations of the map nodes directly. Because of this limitation of SOM and GNG, it is difficult to adjust the map size in order to obtain better clustering results.
- The initial map size is a critical parameter for MATOM. If users do not have sufficient knowledge about their target data elements, it is difficult for them to choose the correct map size. MATOM starting with a small map size may require a long time to finish the clustering process when target data elements are distributed in different map nodes. This is because MATOM needs to build many layers of maps to cluster target genes into different nodes. In another situation where target data elements should be clustered into one cluster, MATOM starting with a big map size may lead to false clustering results because it groups target data elements into different nodes of the initial big map at the beginning. In the situation where target data elements are distributed in different nodes, MATOM starting with a small initial map size also finds the correct clustering results through the multi-step construction of the new layers. Therefore, a small initial map size is recommended.
- Users have to provide some target data elements that allow MATOM to cluster the related data elements correctly in a semi-supervised manner. If there are some data elements that users may be interested in but are not in the target data elements set, called *hidden target data elements*, MATOM could miss them totally in the clustering results. The clusters that contain hidden target data elements are called *hidden target clusters*. Although other clustering algorithms that do not delete noise data do not miss hidden target data elements in the final clustering results, users may also ignore them because users usually do not explore every cluster but focus on only the clusters containing the already-identified data

elements. MATOM sacrifices missing hidden target data elements to get a more efficient training process.

3. Experimental Results

3.1. The Proposed Evaluation Model

From the research work on *E. coli* genes by the University of Oklahoma's Microarray Facility [13], there are thirteen genes regulated by *yhiX* and containing the putative *yhiX* binding motif, called *target genes regulated by yhiX with high confidence*. Another eleven genes regulated by *yhiX* but lacking the binding motif are called *target genes regulated by yhiX with low confidence*. The thirteen genes regulated by *yhiX* with high confidence should be clustered into one cluster. The eleven genes regulated by *yhiX* with low confidence should be clustered into another cluster. The profiles of these two clusters should be similar because the genes in those two groups are both regulated by *yhiX*, but the genes of the first group contain the putative *yhiX* binding motif and the genes of the second group lack the *yhiX* binding motif. In other words, the distance between the centroids of these two clusters is short.

Unlike the traditional algorithm evaluation model, which measures execution time and memory space, our evaluation model focuses on the accuracy of the clustering results. There are seven parameters in our evaluation model: n denoting the number of genes in a cluster; n_h the number of genes regulated by *yhiX* with high confidence; n_l the number of genes regulated by *yhiX* with low confidence; n_{fh} the number of false genes in the cluster that should contain only genes regulated by *yhiX* with high confidence; n_{fl} the number of false genes in the cluster that should contain only genes regulated by *yhiX* with low confidence; and n_A the number of target genes with high and low confidence in the right clusters where target genes should be clustered. The performance measurements are defined as follows:

- The percentage of genes regulated by *yhiX* with high confidence in the clusters: $C_{Th} = \frac{n_h}{n}\%$.
- The percentage of genes regulated by *yhiX* with low confidence in the clusters: $C_{Tl} = \frac{n_l}{n}\%$.
- The percentage of false genes in the cluster that should contain only target genes regulated by *yhiX* with high confidence: $C_{Fh} = \frac{n_{fh}}{n}\%$.
- The percentage of false genes in the cluster that should contain only target genes regulated by *yhiX* with low confidence: $C_{Fl} = \frac{n_{fl}}{n}\%$.
- The accuracy of clusters is the percentage of target genes with high and low confidence that are grouped into the right clusters where the target genes should be clustered: $C_A = \frac{n_A}{n}\%$.
- The average percentage of the first four measurements: $C_{ia} = \frac{\sum C_i}{N}$, where C_i is C_{Th}, C_{Tl}, C_{Fh} or C_{Fl}, and N is the total number of all target clusters that contain the target genes.

An ideal algorithm should give a zero value of C_{Fh} and C_{Fl}. A clustering algorithm may cluster some false genes in its final clusters. Therefore, it is important to measure the percentage of false genes. Moreover, the five performance measurements listed above can present the whole picture of the performance of a clustering algorithm. In the proposed performance evaluation model, C_{Tha}, C_{Tla}, C_{Fha} and C_{Fla} are used. Although the target genes with high confidence and with low confidence should be grouped into two separate clusters by an ideal clustering algorithm, they may be clustered into many clusters. This is the reason why C_A is a key measurement to evaluate the performance of a clustering algorithm. C_A measures the accuracy of the clustering results produced by the algorithm. In the evaluation model, an algorithm that has the best performance should yield higher values of C_{Tha}, C_{Tla} and C_A and lower values of C_{Fha} and C_{Fla}.

3.2. Performance Comparisons

In this section, a performance comparison of the six existing algorithms (CLICK, Hierarchical Clustering, K-means, SOM, GNG, and FLVQ) and MATOM using the proposed evaluation model is presented. The values of each measurement and the critical parameters of the algorithms are listed in Table 1. Figure 1 shows the percentages of target genes that each clustering algorithm can cluster in the right clusters. Figures 2 and 3 show the capabilities of the algorithms to cluster the target genes with high confidence and low confidence, respectively.

A good algorithm should have a high percentage of clustering accuracy and a low percentage of false genes in the final clusters. The average percentage of false genes in the final clusters of genes with high confidence, C_{Fha}, is recorded on the y-axis and the accuracy of clusters, $C_{A,}$ is on the x-axis. The accuracy of clusters only considers how many percents of the target genes with high and low confidence can be clustered into the right clusters, but

ignores the false genes in the right clusters. Therefore, we also need to consider the percentage of false genes in the same performance measurement figure. For instance, in the experiments, K-means_2 clusters all the target genes into the right clusters, but it has more than 90% false genes in those clusters. Giving an extreme example, if there were a cluster that contained 4290 genes (the whole dataset), all targets would be in the right cluster but then there would be more than 99% of false genes in the same cluster. This kind of clustering results does not provide any valuable information.

In Figure 1, MATOM has the best balance point of the accuracy of the clustering results and the percentage of false genes. This figure also verifies that the performance of the classical SOM using the sequential training rule is worse than that of the batch training version of SOM, a result that has been stated in a number of existing works [3]. The two versions of SOM have average performances in terms of clustering accuracy and percentage of false genes. As also shown in this figure, with different values of K, K-means provides totally different results. CLICK has an average performance in terms of the accuracy of the clustering results but gives a very high percentage of false genes. Therefore, it does not produce valuable clustering results. FLVQ has a very low clustering accuracy and is not suitable for gene expression analysis. GNG has an average performance in terms of clustering accuracy and percentage of false genes. By cutting the dendrogram tree to find the final clusters, the hierarchical clustering algorithm still yields only an average performance.

Figure 2 reports the capabilities of the algorithms in clustering the target genes with high confidence. The average percentage of false genes in the clusters of genes with high confidence, C_{Fha}, is recorded on the y-axis and the average percentage of target genes with high confidence, C_{Tha}, is on the x-axis. MATOM is a semi-supervised clustering algorithm, which tracks the thirteen target genes with high confidence during the clustering process. MATOM will not miss the target genes with high confidence. Therefore, for MATOM, in the clusters of target genes that have high confidence, the percentage of false genes is more important than that of target genes. In Figure 2, MATOM has the highest percentage of target genes with high confidence and the lowest percentage of false genes. In Figure 2, K-means_3, K-means_1, hierarchical clustering, FLVQ and GNG also have better performances than K-means_2, CLICK, SOM_1, and SOM_2. The batch version of SOM has fewer false genes than the sequential version of SOM, and thus is more suitable for the *E. coli* gene expression data analysis. CLICK and K-means_2 have almost 100% of false genes

Figure 3 assesses the capabilities of the algorithms in clustering target genes with low confidence. The average percentage of false genes in the clusters of genes with low confidence, C_{Fla}, is recorded on the y-axis and the average percentage of target genes with low confidence, C_{Tla}, is on the x-axis. In Figure 3, SOM_1 performs better than MATOM and other unsupervised manner clustering algorithms. But examining Figure 1, the batch version of SOM does not cluster more target genes in the right clusters. In other words, the batch version of SOM has lower accuracy of clustering results than MATOM. Combining the performances recorded in Figures 1 and 3, MATOM performs the best because of its highest accuracy of clusters and its second-highest percentage of the target genes with low confidence as well as its lowest percentage of false genes. K-means_1, K-means_2 and CLICK do not find more target genes with low confidence, but have more than 80% of false genes. Hierarchical clustering is poor at clustering the target genes with low confidence. FLVQ, with the worst performance in Figure 3, does not find any target genes with low confidence. GNG, K-means_3, and the sequential version of SOM cluster a small number of target genes with low confidence but have more than 80% of false genes in the final clusters.

Table 1. The Performances Results of the Clustering Algorithms

Algorithm	Critical Parameter	*Ctha*	*Ctla*	*Cfha*	*Cfla*	*Ca*
Hierarchical Clustering	cutoff=1.54	52.00%	4.00%	44.00%	44.00%	62.50%
K-means_1	K=100	52.93%	17.88%	47.07%	82.12%	41.66%
K-means_2	K=3	1.46%	1.24%	98.54%	98.76%	100.00%
K-means_3	K=200	54.63%	13.88%	45.37%	86.11%	41.67%
GNG	22 map nodes	57.14%	9.52%	42.86%	90.48%	66.67%
SOM_1	Map size 4x20; batch training rule	44.44%	33.33%	55.56%	66.67%	54.17%
SOM_2	Map size 4x20; sequential training rule	33.33%	12.82%	66.67%	87.18%	75.00%
CLICK	EdgeThreshold=6	7.63%	2.66%	92.37%	97.34%	62.50%
FLVQ	Initial 50 clusters	62.94%	0.00%	37.05%	0.00%	16.67%
MATOM	Initial map size 2x2, size of final cluster 18	79.41%	25.80%	17.65%	74.19%	79.17%

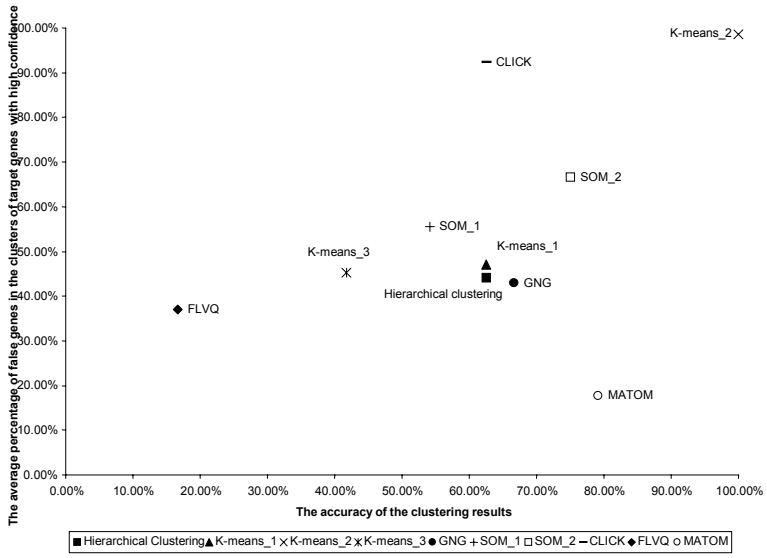

Figure 1. The Accuracy of the Clustering Results vs. Percentage of False Genes in the Final Clusters of Target Genes with High Confidence

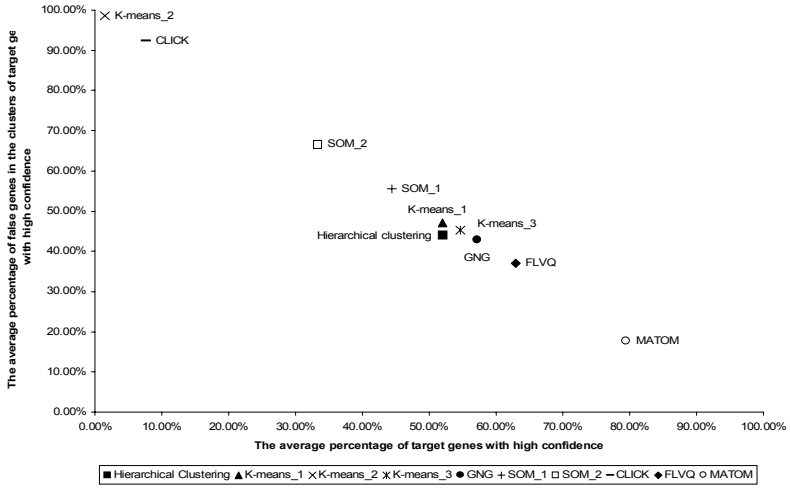

Figure 2. The Target Genes with High Confidence vs. the Percentage of False Genes in the Clusters of Target Genes with High Confidence

3.3 Comparison Conclusions

From the analysis above, it can be concluded that MATOM performs the best among the studied clustering algorithms in analyzing the *E. coli* gene expression data. MATOM has the best balance points in Figures 1 and 3. Although the batch version of SOM performs a little bit better than MATOM in Figure 3, the batch version of SOM has much lower accuracy of clusters than MATOM in Figure 1 does.

Comparing with other clustering algorithms, MATOM provides several functions to analyze gene expression data, such as the relation tree of clusters and the function to allow users to determine the size of the final clusters. In a semi-supervised manner, MATOM avoids the negative influence of noise data on the clustering process. As seen in the performance comparison, MATOM has more advantages and useful features for clustering *E. coli* gene expression data than other algorithms do.

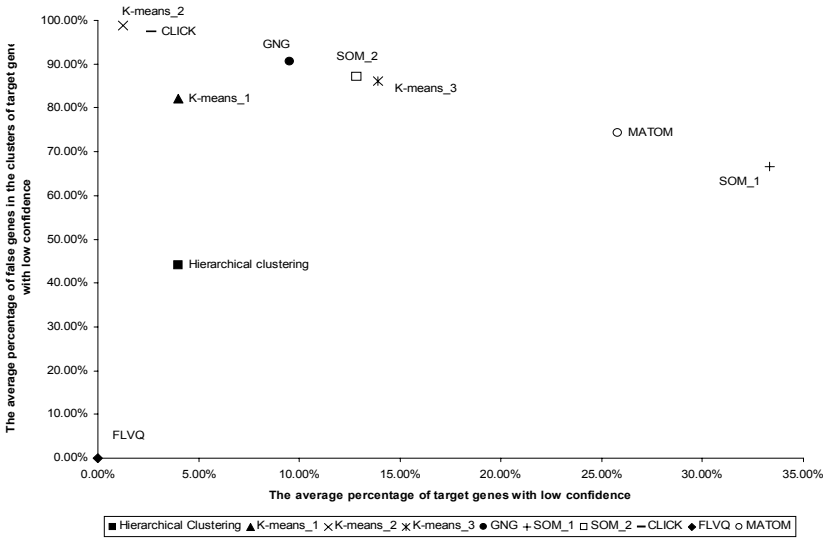

Figure 3. The Target Genes with Low Confidence vs. the Percentage of False Genes in the Clusters of Target Genes with Low Confidence

4. Conclusions and Future Research

This paper proposed a novel clustering algorithm called Multilayer Adjusted Tree Organizing Map (MATOM) to analyze the *E. coli* gene expression data. The paper then presented the comparison results for clustering the *E. coli* gene expression data and showed that MATOM performs the best. MATOM was developed to take the characteristics of the *E. coli* gene expression data into consideration. For other biological analysis applications that also share the properties similar to those of the *E. coli* gene expression data, MATOM should yield a good performance. However, to verify this, our future work includes running MATOM on other biological data sets.

References

[1] K. Alsabti; S. Ranka; V. Singh, *An efficient k-means clustering algorithm* In Proceedings of IPPS/SPDP Workshop on High Performance Data Mining, 1998.

[2] Alvis Brazma; Jaak Vilo, *Gene Expression Data Analysis* FEBS Letters FEBS Letters Volume 480, Issue 1, Page(s): 17-24, 2000.

[3] A. Baraldi; P. Blonda, *a survey of fuzzy clustering algorithms for pattern recognition. I and II* Systems, Man and Cybernetics, Part B, IEEE Transactions, Volume 29 Issue: 6, Page(s): 778 –801, Dec. 1999.

[4] C.A. Harrington, et al., *Monitoring gene expression using DNA microarrays* Current Opinion in Microbiology 3(3), Page(s):285-291, 2000.

[5] M. Eisen; P. T. Spellman; D. Botstein; and P. O. Brown, *Cluster analysis and display of genome-wide expression patterns* Proceedings of National Academy of Science USA, Page(s): 14863-14867. 1998.

[6] B. Fritzke, *A growing neural gas network learns topologies* In G. Tasauro, D. Tourestzky and T. Leen, editors, Advances in Neural Information Processing Systems, volume 7, MIT Press, Cambridge MA, Page(s): 625-632, 1995.

[7] R.W. Johnson; D.W. Wichern, *Applied Multivariate Statistical Analysis* Prentice Hall, Upper Saddle River, New Jersey, 1998.

[8] G.S. Michaels; D.B. Carr; M. Askenazi; S. Fuhrman, X. Wen; R. Somogyi, *Cluster Analysis and Data Visualization of Large-Scale Gene Expression Data* Pac. Symposium on BioComputing, Page(s): 42-53, 1998.

[9] B. Mirkin, *Mathematical Classification and Clustering* Kluwer Academic Publishers, Dordrecht, Boston, London, 1996.

[10] N. B. Karayiannis; J. C. Bezdek, *An integrated approach to fuzzy learning vector quantization and fuzzy c-means clustering*, IEEE Trans. Fuzzy Syst., vol.5, no.4, Page(s): 622-628, 1997.

[11] T. Kohonen, *The self--organizing map*. 2nd, Springer Press, 1995.

[12] Roded Sharan; Ron Shamir, *CLICK: A Clustering Algorithm with Applications to Gene Expression Analysis.* Proc. 8th International Conference on Intelligent Systems for Molecular Biology, Page(s): 307-316, 2000.

[13] Don L. Tucker; Nancy Tucker; Tyrrell Conway, *Gene expression profiling of the pH response in E. coli* 2002

[14] A. Watson; A. Mazumder; M. Stewart; S. Balasubramanian, *Technology for microarray analysis of gene expression* Current Opinion in Biotechnology, Page(s): 9(6):609-614, 1998.