Multi-SOM: an Algorithm for High-Dimensional, Small Size Datasets

Shen Lu

University of Arkansas at Little Rock,
Department of Computer Science,
Donaghey College of Engineering and Information Technology,
Little Rock, AR 72204 USA.
shenlu2011@gmail.com

Richard S. Segall

Arkansas State University,
Department of Computer & Information Technology,
College of Business,
State University, AR 72467-0130 USA.
rsegall@astate.edu

ABSRACT

Since it takes time to do experiments in bioinformatics, biological datasets are sometimes small but with high dimensionality. From probability theory, in order to discover knowledge from a set of data, we have to have a sufficient number of samples. Otherwise, the error bounds can become too large to be useful. For the SOM (Self-Organizing Map) algorithm, the initial map is based on the training data. In order to avoid the bias caused by the insufficient training data, in this paper we present an algorithm, called Multi-SOM. Multi-SOM builds a number of small self-organizing maps, instead of just one big map. Bayesian decision theory is used to make the final decision among similar neurons on different maps. In this way, we can better ensure that we can get a real random initial weight vector set, the map size is less of consideration and errors tend to average out. In our experiments as applied to microarray datasets which are highly intense data composed of genetic related information, the precision of Multi-SOMs is 10.58% greater than SOMs, and its recall is 11.07% greater than SOMs. Thus, the Multi-SOMs algorithm is practical.

KEYWORDS

Self-Organizing Maps, Weights Vector, Bayesian Decision Theory, Feature Selection, Sample Selection.

1. INTRODUCTION

Self-Organizing Maps (SOM) provides mapping from the input space to the clusters. According to [18], a SOM attempts to organize clusters that are near each other in the grid-space to those seeds that are close in the input space. It differs from k-means clustering because it defines an area around each cluster seed in the grid via a neighborhood function. Clusters that are close in proximity on the grid have similar input variables.

We survey microarray experimental results, in order to gain insight into the data – possibilities and problems – to determine whether the data are sufficient and to select the proper preprocessing and modeling techniques. Several different data sets are considered. For liver cancer [2], there are 17,400 genes and 179 samples, for lung cancer [6], there are 12,600 genes and 245 samples, for NIH cancer dataset [17], 12,196 genes and 240 samples, for prostate cancer [11], there are 26,260 genes and 103 samples. We can make such a conclusion that the majority of Microarray experiments cannot supply enough samples to do classification. The following methods can tell us how many samples are enough to train a learning model with regard to the number of samples, the number of genes and the percentage of errors.

In [13], the inequality shown below provides a general bound on the number of training examples sufficient for any consistent learner, L, to successfully learn any target concept in H, in which m means the number of training examples, |H| means the size of hypothesis space H. L will, with probability (1- δ), output a hypothesis h with $error_D$ (H) < ε, after observing a reasonable number of training examples and performing a reasonable amount of computation.

$$M \geq (1/\varepsilon)(\ln|H| + \ln(1/\delta))$$

In accordance with this inequality, we can find the number of samples in each dataset mentioned above is not sufficient for the learner to learn a target concept. Because limited samples of data might misrepresent the general distribution of data, estimating true accuracy from such samples can be misleading.

If we take a model of the true distribution and train it with a highly skewed distribution, the final classifier accuracy might be unacceptably low. In this paper, first of all, we use proper preprocessing techniques, such as t-test and fold-change, and machine learning algorithms, to investigate Microarray data sets, and then we present a new algorithm, called Multi-SOM, to model Microarray data.

2. RELATED WORK

Previous works by co-author Segall on applications of SOM to data mining for bioinformatics include [19] to [24] as discussed next. [19] presented a chapter on data mining of microarray databases for biotechnology. [20] performed data mining of microarray databases for human lung cancer. [21] performed data visualization and data mining of microarray databases for continuous numerical-valued Abalone fish data and discrete nominal-valued mushroom data using evolutionary algorithms specifically for neural networks and generic algorithms. [22, 23, 24] performed data mining of microarray databases of Leukemia cells using single SOM. This paper extends the methodology used in previous research from using single-SOM to a new algorithm that uses Multi-SOM.

In [26] is presented a Multi-layer neural network. This network consists of two types of elements: CU is Clustering Units, which distinguish some clusters in the input data, and DCB is Data Completion Blocks which are between input and CUs to prepare data for CUs. The aim of the CU elements is to independently assign label Y to each corresponding input vector X. The aim of the DCB is to prepare input vectors X basing on the outputs from the previous layer. The DCB can effectively describe the distribution of the input, but, in order to build DCB layer, the entire dataset has to be handled first. If the dataset is large, it takes time to do so. Moreover, the advantage of SOMs is to reduce the dimension of the data. So, using clustering in pre-processing step is against the original property of SOMs. An algorithm, called Multi-layer Kohonen Self-Organizing Feature Map (MLKSFM) is

given in [27]. It has a hierarchical structure in which the pre-classification if performed at the lower level MLKSFM and the final language identification if performed at the top level. This approach is to use k-means to retune SOMs which have already been built.

Some improvements of multi-layer SOMs algorithms are because of the requirements of the applications. [26][8] The additional layer is for the evaluation of domain features. Although they belong to Multi-layer SOMs, they cannot be used to improve the hypothesis of SOMs.

Rauber et al. [16] developed the Growing Hierarchical Self-Organizing Map (GHSOM) that is an Artificial Neural Network (ANN) model with hierarchical architecture composed of independent growing self-organized maps. The motivation was to provide a model that adapts its architecture during its architecture during its unsupervised training process according to the particular requirements of the input data. Rauber et al. [16] applied their GHSOM model to text data only with data sets of 420 and 10,000 articles respectively with good results, but did not apply to microarray data sets as this articles addresses.

3. SELF ORGANIZING MAPS

3.1 SOMs Introduction

Self-Organizing Maps belong to competitive neural networks. Competitive learning is an adaptive process in which neurons in a neural network are sensitive to different input categories, sets of samples in a specific domain of the input space. ([1], [3], [4], [7], [9], [10], [12], [14], [15], [25])

A Self-Organizing Map consists of two layers as shown in figure 1. Suppose that we have a set of n-dimensional vectors. The first layer of SOMs is the input data which transfer to the second layer. The second layer has a number of neurons which are chosen arbitrarily and can be used to representing the feature space.

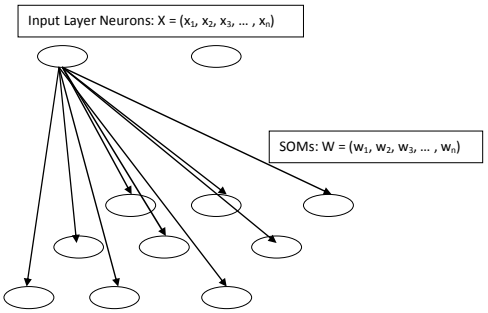

Figure 1. SOMs Architecture

On the second layer, each neuron has the same dimension as the input neuron from the first layer. First of all, weights of the neurons on the second layer are set randomly. During the training process, they have their own weights vector and update those during the training process. When an input x arrives from the first layer to the second layer, the neuron that is best able to represent it wins the competition and is allowed to learn it even better. Moreover, not only the winning neuron but also its neighbors on the lattice are allowed to learn. All neighbors m_k of m_j can be updated in this way:

$$m_k \leftarrow m_k + \alpha(x_i - m_k)$$

The neighbors of m_j are defined to be all m_k, such that the distance between l_j and l_k is small. The effect of the update is to move the prototype closer to the input data, but also to maintain a smooth two-dimensional special relationship between the prototypes.

3.2 SOMs Learning Process

At the beginning, all weights of the second layer's neurons are set to random values. The training process starts with selection of the input neuron. In the training set, we randomly select one of the neurons as the input of SOMs. The difference between input neuron and all neurons of SOMs are calculated as follows:

$$D_{ij} = |X^l - W_{ij}| = \text{sqrt}((x_1 - w_{ij1})^2 + ... + (x_n - w_{ijn})^2)$$

where i and j are the index of neurons in the output layer, l is the index of input neurons, n is the index of the dimension on the input neuron vector l. After that, the SOMs choose the winning neuron, the one whose weights vector is the most similar or closest to the input neuron.

$$D(k_1, k_2) = \min_{ij} D_{i,j}$$

Here, k_1 and k_2 are the index of the winning neuron. After finding the winning neuron, we need to update the weights of the winner and all the adjacent neurons, as follows,

$$h(\rho, t) = \exp(\rho^2 / (2*\sigma^2(t))$$

$$\rho = \text{sqrt}((k_1 - i)^2 + (k_2 - j)^2)$$

After calculating the topological neighborhood function for each neuron, the weights of all the neurons are updated, as follows,

$$W_{ij}(t+1) = W_{ij}(t) + \alpha(t)\, h(\rho, t)\, (X^l(t) - W_{ij}(t))$$

$\alpha(t)$ is a learning rate function that also decrease with time. If a neuron is a winner or adjacent to the winner, then its weight vector is updated or remains unchanged otherwise. On each step, the SOMs determines the neuron whose weights vector is the most similar to the input vector, and correct it and its neighbor's weights vector to make them closer to the input vector.

4.MULTI-SOM

4.1 Introduction

Self-Organizing Map uses two-dimensional topology to present high-dimensional data. Like other Neural Network algorithms, SOM algorithm can significantly discrete the error, nearly to the level of Kmeans. So, two dimensional network structure used by SOM is reasonable for high-dimensional datasets. The structure of SOM gives rise to discussions about neighborhood preservation or violation. From input space to output space, self organizing maps are built up with randomly chosen training data. Therefore, the size of the map and the association of neighborhood can affect the quality of the SOM model. According to experimental results, it is hard to say that the bigger the size of the map, the better.

Moreover, SOMs are based on a number of plausible heuristics, such the initial weights, the size of the map, learning rates, and so on. All these can lead to very slow convergence, poor performance or other unsatisfactory results.

Therefore, in this paper, a Multi-SOMs algorithm is presented. In this algorithm, we choose a number of small maps. Since the size of the map is not big, it is fast to train a SOM model. Totally, we have a large number of neurons with randomly assigned weights. Since the size of neurons is large enough that we can consider that the distribution of the weights on Multi-SOMs is close to the true distribution and its bias is statistically small.

Other than the improvement of the distribution, we also use the Bayes decision theory to make a prediction. Consider each map is a training set. For each input testing data, based on the probability distribution of the training set, according to Bayes decision theory, we can find the maximum probability of the class that the input data can be classified.

4.2 Description of Multi-SOMs Algorithm

Algorithm: Multi-SOMs

Step 1: train a number of SOMs

Input: training samples, samples; the learning rate, l; the map size, k; maximum iteration times, MaxIteration.

Output: A number of Self-Organizing Maps trained with classify samples.

Method:

1. generate a number of SOM.
2. for each SOM{

   2.1. do{

   2.2. for each Input neuron {

   2.3. check the similarity of the sample X^l to the weight vector W_{ij} on the SOM

   2.4. record the most similar weight vector

   2.5. update the neighborhood weight vectors.

   2.6.}

3.} while (t < MaxIteration)

}

Step 2: Classify the input neuron

4.Calculate $P(c_k)$, the probability of class k = 1, …, m

5.For each neuron, X, in the training set{

   5.1 Find the similar neuron, S_{ij} , from the n^{th} SOM

   5.2 Record the class of the similar neuron

   5.3$P(c_m \mid X^l) = P(X^l \mid c_m)*P(c_m) / \Sigma^m_{k=1} P(X^l \mid c_k)*P(c_k)$

   5.4 $C(X^l) = Max^m_{k=1} (P(c_k \mid X^l))$

}

5. EXPERIMENTS

We compare the performance of Multi-SOM with SOM algorithm in decision making. We finished experiments in the following steps. First of all, we generated a high quality data set through sample selection and feature selection. And then, we used the data set with the best quality to test Multi-SOMs and SOMs.

5.1 Sample Selection and Feature Selection

Statistical process for microarray expression data includes the following steps:

1. pre-processing: because of experimental errors, some values of expression data are missing. We use KNN algorithm to automatically impute missing values first.
2. sample selection: since microarray expression data set is not very big, we can use total data for any experiments and applications. However, regarding to the different number of treated samples and untreated samples, we randomly generate data sets in which both treated and untreated classes have the same number of samples.
3. feature selection: even if data mining analysis can be performed, it is still extremely useful to reduce the data set to those genes that are best distinguish between the sample classes.

After generating different data sets, as the output of the process, we use data mining analysis to evaluate them. Precision model building includes two steps: model building and model validation. Model building involves in training data selection. Model validation involves in testing the built model with testing samples and measuring the precision and recall of the output of the generated model.

We use KNN, Random Forest, Multipass-Lqv, and SOM algorithms to calculate the precision and recall on different data sets. KNN is based on the direct comparison of the distance between two neighbors. This algorithm is good for high dimensional vectors. Random Forest is based on decision tree theory. Since the best features are selected to build decision trees, the significance of different features are considered in this algorithm. Multipass-Lqv and SOM belong to neural network algorithm. Since samples can be randomly selected as input for many times, these algorithms are good for high-dimensional small size data sets, such as microarray expression data.

We complete the experiments with the original dataset, dataset generated randomly, and dataset generated by statistical approaches, which can be found in table 1. For each dataset, four algorithms, such as K Nearest Neighbor (KNN), Random Forest, SOM, Multipass-Lvq, are used to test the quality of the output data. For each data set, among those algorithms, the one with the best performance on all data sets can be used to evaluate the quantity of the data set. The measurements of the performance of each algorithm on the different datasets can be used to evaluate the quantity of the different datasets. These measurements include accuracy, sensitivity, specificity, precision and recall.

According to the experimental result, we can see that the dataset, with the total data, fold-change set to 2.0 and pValue set to 0.01, performs better than others.

Table 1. Description of Datasets

	dataset	samples	genes	removing perc	total order	sub order
1	total	179	19536			
2	total_pvlaue005	179	6660	0.659090909	24	8
3	total_pvlaue001	179	4383	0.775644963	18	6
4	total_fc20	179	2717	0.860923423	12	4
5	total_fc20_pvalue005	179	1270	0.93499181	6	2
6	total_fc20_pvalue001	179	772	0.96048321	3	1
7	total_fc15	179	5795	0.703368141	19	7
8	total_fc15_pvalue005	179	2768	0.858312858	14	5
9	total_fc15_pvalue001	179	2181	0.888359951	9	3
10	balance3	50	19536			
11	balance3_pvlaue005	50	5864	0.6998362	21	7
12	balance3_pvlaue001	50	3939	0.798372236	16	6
13	balance3_fc20	50	2785	0.85744267	15	5
14	balance3_fc20_pvalue005	50	886	0.95464783	4	2
15	balance3_fc20_pvalue001	50	743	0.961967649	2	1
16	balance3_fc15	50	5900	0.697993448	22	8
17	balance3_fc15_pvalue005	50	2403	0.876996314	10	4
18	balance3_fc15_pvalue001	50	2135	0.890714578	7	3
19	balance2	50	19536			
20	balance2_pvlaue005	50	5948	0.695536446	23	8
21	balance2_pvlaue001	50	4006	0.79494267	17	6
22	balance2_fc20	50	2739	0.859797297	13	5
23	balance2_fc20_pvalue005	50	898	0.954033579	5	2
24	balance2_fc20_pvalue001	50	739	0.9621724	1	1
25	balance2_fc15	50	5851	0.700501638	20	7
26	balance2_fc15_pvalue005	50	2422	0.876023751	11	4
27	balance2_fc15_pvalue001	50	2148	0.89004914	8	3

Table 2. Rank the order of datasets by accuracy, sensitivity, specificity for original and random datasets.

Precision-Recall Measurement for the Sampling		
feature selection	algorithms	the good sample
no	knn	total
	random forest	balance2
	multipasslvq	balance2
	som	total
pValue = 0.05	knn	total
	random forest	balance2
	multipasslvq	total
	som	total
pValue = 0.01	knn	total
	random forest	total
	multipasslvq	total
	som	total
fc = 2.0	knn	balance3
	random forest	total
	multipasslvq	balance2
	som	balance3
fc = 2.0 pValue = 0.05	knn	balance3
	random forest	balance3
	multipasslvq	balance2
	som	total
fc = 2.0 pValue = 0.01	knn	balance3
	random forest	total
	multipasslvq	total
	som	total
fc = 1.5	knn	total
	random forest	balance2
	multipasslvq	balance3
	som	balance3
fc = 1.5 pValue = 0.05	knn	total
	random forest	balance3
	multipasslvq	total
	som	total
fc = 1.5 pValue = 0.01	knn	total
	random forest	total
	multipasslvq	total
	som	total

Table 3. Rank the order of datasets by precision-recall measurement for original and random datasets

Accuray-sensitivity-specificity Measurement for the Sampling					
feature selection	measurement	algorithms	rank1	rank2	rank3
no	accuracy	som	total	balance2	balance3
	sensitivity	knn	balance3	balance2	total
	specificity	MultipasssLvq	balance3	balance2	total
		random forest	total	balance3	balance2
pValue = 0.05	accuracy	random forest	total	balance2	balance3
		som	total	balance2	balance3
	sensitivity	knn	total	balance2	balance3
	specificity	random forest	balance3	total	balance2
		MultipasssLvq	balance2	balance3	total
pValue = 0.01	accuracy	som	total	balance2	balance3
	sensitivity	knn			
	specificity	random forest	balance3	total	balance2
		SOM	balance2	total	balance3
		MultipasssLvq	balance2	balance3	total
fc = 2.0	accuracy	random forest	total	balance2	balance3
		som	total	balance2	balance3
	sensitivity	knn	total	balance2	balance3
	specificity	knn	balance3	total	balance2
		random forest	total	balance2	balance3
		MultipasssLvq	balance2	total	balance3
fc = 2.0 pValue = 0.05	accuracy	som	total	balance2	balance3
	sensitivity	knn	total	balance2	balance3
	specificity	random forest	total	balance3	balance2
fc = 2.0 pValue = 0.01	accuracy	som	total	balance2	balance3
	sensitivity	knn	total	balance2	balance3
	specificity	random forest	total	balance2	balance3
		MultipasssLvq	balance3	balance2	total
fc = 1.5	accuracy	som	total	balance2	balance3
	sensitivity	knn	total	balance2	balance3
	specificity	random forest	balance3	balance2	total
fc = 1.5 pValue = 0.05	accuracy	som	total	balance2	balance3
	sensitivity	knn	total	balance2	balance3
	specificity	random forest	total	balance3	balance2
		MultipasssLvq	balance3	balance2	total
fc = 1.5 pValue = 0.01	accuracy	som	total	balance2	balance3
	sensitivity	knn	total	balance2	balance3
	specificity	random forest	total	balance2	balance3

5.2 Performance

Figure 2 is a 100*100 self organizing map. Each position is an N dimensional neuron vector. Initially, in this map, all of the values of vectors are randomly generated. Before we train this map, the training data can be visualized in figure 2.

For the best data set we choose, we visualize its distribution on the self-organizing map as follows, after feature selection and sample selection.

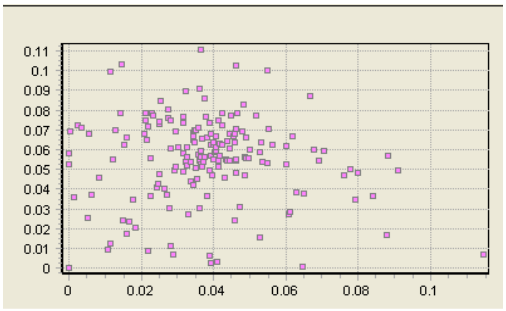

Figure 2. The distribution of the training set after sample selection and feature selection

We followed the following steps to evaluate the two algorithms.

1. We build input vectors.
2. We build a n*n map with a random weight vector on each position.
3. We use input vectors to train SOM map.
4. We use the trained map to classify input vectors.

After training with Multi-SOMs model, the distribution of the weights vectors can be visualized in figure 3.

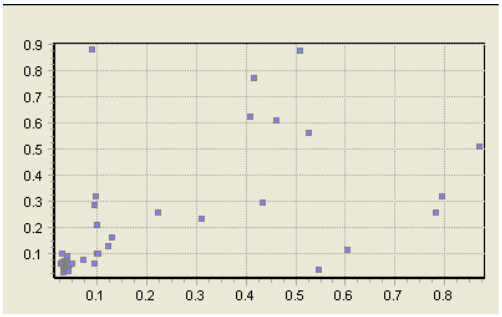

Figure 3. The distribution of the weights vectors after training.

We compare the performance of SOM and Multi-SOM on 657KB liver cancer Microarray gene expression data set with 179 samples and 772 genes. Experimental results are showed in figure 4. The precision of SOM is 83.65% and Multi-SOM 94.23%. The recall of SOM is 81.52%, and Multi-SOM 92.59%. Therefore, the precision of Multi-SOMs is 10.58% greater than SOMs, and its recall is 11.07% greater than SOMs.

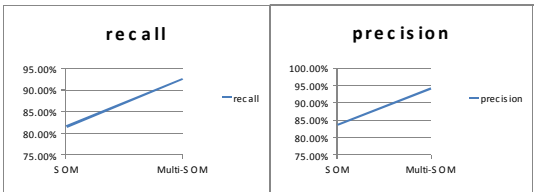

Figure 4. Precision and Recall for SOM and Multi-SOM Algorithms

6. CONCLUSION

In this paper, the open issue in bioinformatics is presented that the size of the sample set and the dimension of the sample set is critical to computational efficiency and accuracy. The theory of SOMs is based on some hypothesis about initial weights, the size of the map, the learning rate and so on. However, these points can still affect the quality of the SOMs model for classification or clustering. In order to solve this problem, an algorithm, called Multi-SOMs is proposed. This algorithm splits SOM into a number of small maps and makes a final prediction with Bayes decision theory. The time and space complexities are not issues as the Multi-SOM algorithm was very fast for these sizes of problems.

Moreover, for experiments, we present the pre-processing of the dataset and the classification process. The experimental results show us that Multi-SOMs is better than SOMs as applied to microarray datasets that by their design are highly dense. It is noted that Multi-SOMs are assigned the size of the map before building the model. In future, another kind of SOMs technique, called Growing SOMs (GSOM) [28] can be combined to dynamically build the model.

7. REFERENCES

[1] S.-I. Amari, Topographic organization of nerve fields. *Bulletin of Mathematical Biology*. v42, n. 3, 1980, pp.339-364.

[2] X Chen, S.T. Cheung, S. So, S.T. Fan, C. Barry, J. Higgins, K.M. Lai, J. Ji, S. Dudoit, I.O. Ng, et al. Gene expression patterns in human liver cancers. *Mol. Biol. Cell*. Vol. 13, Issue 6, pp. 1929-1939, June 2002.

[3] R. L. Didday, (1970) *The Simulation and Modeling of Distributed Information Processing in the Frog Visual System*. PhD thesis, Stanford University.

[4] R. L. Didday, (1976) A model of visuomotor mechanisms in the frog optic tectum. *Mathematical Biosciences*, 30:169-180.

[5] S. Dudoit, J. Fridly and, T. P. Speed, Comparison of discrimination methods for the classification of tumors using gene expression data, *Journal of the American Statistical Association*. 2002. Vol. 97, No. 457. pp.77-88.

[6] G. J. Gordon et al., Translation of microarray data into clinically relevant cancer diagnostic tests Using Gene Expression Ratios in Lung Cancer and Mesothelioma, *Cancer Res*. Vol. 62, No. 17, pp. 4963–4967, September 2002.

[7] S. Grossberg, On the development of feature detectors in the visual cortex with applications to learning and reaction-diffusion system. *Biological Cybernetics*. v21, 1976, pp. 145-159.

[8] Z. Huang. A combined self-organizing feature map and multilayer perception for isolated word recognition. *IEEE Transactions on Signal Processing*, Vol. 40, Issue 11, pp. 2651–2657, November 1992.

[9] T. Kohonen, K. Mkisara and T. Saramki, Phonotopic maps insightful representation of phonological features for speech recognition. *Proceedings of the Seventh International Conference on Pattern Recognition*, 1984, pp. 182-185.

[10] T. Kohonen, Self-organized formation of topologically correct feature maps. *Biological Cybernetics*. v43. 1982, pp.59-69.

[11] J. Lapointe, C. Li, M. van de Rijn, J.P. Huggins, E. Bair, et al. Gene expression profiling identifies clinically relevant subtypes of prostate cancer. *Proceeding of the National Academy of Sciences*. Vol. 101, No. pp. 3811-816, January 2002.

[12] C. v. d. Malsburg, Self-organization of orientation sensitive cells in the striate cortex. *Kybernetik*. v14, 1973, pp.85-100.

[13] T. Mitchell. *Machine Learning*. Ohio, McGraw-Hill, 1997.

[14] M. Nass and L. N. Cooper, A theory for the development of feature detecting cells in visual cortex. *Biological Cybernetics*. v19, n. 1, 1975, pp. 1-18.

[15] R. Pérez, L. Glass and R. J. Shlaer, Development of specificity in cat visual cortex. *Journal of Mathematical Biology*. v1. 1975, pp. 275-288.

[16] A. Rauber, D. Merkl, and M. Dittenbach, *The growing hierarchical self-organizing map: Exploratory analysis of high-dimensional data,* IEEE Transactions on Neural Networks, 2002, No. 6, pp. 1331-1341.

[17] A. Rosenwald, G. Wright, W.C. Chan, et al. The use of molecular profiling to predict survival after chemotherapy for diffuse large-B-cell lymphoma. *The New England Journal of Medicine*. Vol. 346, pp. 1937-1947, June 2002.

[18] *Applying Data Mining Techniques Using Enterprise Miner™ 4*, Course Notes, SAS Press, 2002, SAS Institute, Cary, NC.

[19] R. S. Segall, "Data mining of microarray databases for biotechnology," *Encyclopedia of Data Warehousing and Mining*, Edited by John Wang, Montclair State University, USA; Idea Group Inc., 2006, ISBN 1-59140-557-2, pp.734-739.

[20] R. S. Segall and Q. Zhang, "Data mining of microarray databases for human lung cancer, *Proceedings of the Thirty-eighth Annual Conference of the Southwest Decision Sciences Institute,* vol. 38, no. 1, March 15-17, 2007, San Diego, CA.

[21] R. S. Segall, and Q. Zhang, "Data visualization and data mining of continuous numerical and discrete nominal-valued microarray databases for bioinformatics," *Kybernetes: The International Journal of Systems & Cybernetics,* Volume 35, Number 10, 2006, pp. 1538-1566.

[22] R.S. Segall and R.M. Pierce, "Data mining of leukemia cells using self-organized maps," *Proceedings of 2009 Conference on Applied Research in Information Technology*, sponsored by Acxiom Laboratory of Applied Research

(ALAR), University of Central Arkansas (UCA), Conway, AR, February 13, 2009, pp. 92-98.

[23] R. S. Segall and R. M. Pierce, "Advanced data mining of leukemia cell micro-arrays", *Proceedings of 13th World Multi-Conference on Systemics, Cybernetics and Informatics: WMSCI 2009*, Orlando, FL, July 10-13, 2009.

[24] R. S. Segall, and R. M. Pierce, "Advanced data mining of leukemia cell micro-arrays", *Journal of Systemics, Cybernetics and Informatics (JSCI)*, Volume 7, Number 6, 2009, pp.60-66.

[25] N. V. Swindale, A model for the formation of ocular dominance stripes, *Proceedings of the Royal Society of London*, B, 215, pp. 211-230, 1980.

[26] A. Tomczyk, P. S. Szczepaniak, and B. Lis. Generalized multi-layer Kohonen network and its application to texture recognition. *International Conference on Artificial Intelligence and Soft Computing*, June 2004. Zakopane, Poland.

[27] L. Wang, E. Ambikairajah, E. H. C. Choi. A comparison study of the multi-layer Kohonen self-organizing feature maps for spoken language identification. *IEEE Workshop on Automatic Speech Recognition and Understanding*, December, 2007. Kyoto, Japan.

[28] Wikipedia, Growing self-organizing map, 2012 http://en.wikipedia.org/wiki/Growing_self-organizing_map