Proceedings of the 2007 IEEE Symposium on
Foundations of Computational Intelligence (FOCI 2007)

Performance Optimization of Adaptive Resonance Neural Networks Using Genetic Algorithms

Hussein T. Al-Natsheh, *Student Member, IEEE* Taisir M. Eldos, *Member, IEEE*
natsheh@ieee.org eldos@ieee.org
Department of Computer Engineering
Jordan University of Science and Technology

***Abstract*— We present a hybrid clustering system that is based on the Adaptive Resonance Theory 1 (ART1) Artificial Neural Network (ANN) with a Genetic Algorithm (GA) optimizer, to improve the ART1 ANN settings. As a case study, we will consider text clustering. The core of our experiments will be the quality of clustering, Multi-dimensional domain space of ART1 design parameters has many possible combinations of values that yield high clustering quality. These design parameters are hard to estimate manually. We proposed GA to find some of these sets. Results show better clustering and simpler quality estimator when compared with the existing techniques. We call this algorithm Genetically Engineered Parameters ART1 or ARTgep.**

I. Introduction

Unsupervised training is defined as self-organizing neural nets that group similar input vectors together, without the use of training data to specify what a typical member of each group looks like or to which group each vector belongs. Unsupervised ANN is mainly used for clustering [1], and ART1 is a typical example. In addition to being unsupervised, ART1 is an online learning ANN, which means that it can adapt to new data sets after being detached from the training algorithm [2]. In contrast to how other ANNs behave regarding new input vectors; the ART1 ANNs do not require the training to restart for the adaptation to take place.

Genetic Algorithms mimic the biological evolution process known as "Survival of the fittest"; improved solutions evolve from previous generations until reaching to a near optimal solution [3]. Designing a genetic algorithm involves:

1. Devising suitable structures to represent the solutions (later called chromosomes or individuals).
2. Defining a set of genetic operators that produce new solutions from the existing ones.
3. Devising an index that can be used as measure of quality for the solutions to drive the evolution process, typically called the fitness function.
4. Selection rules that maintain the population size bound by getting rid of solutions.

GAs have been used as optimization tools in many applications. To use it as ANN ART design parameters optimizer, each set of the ART1 design parameters represents a possible solution; Figure 1 depicts the ART1 architecture and its design parameters: b_{ij} (bottom-up

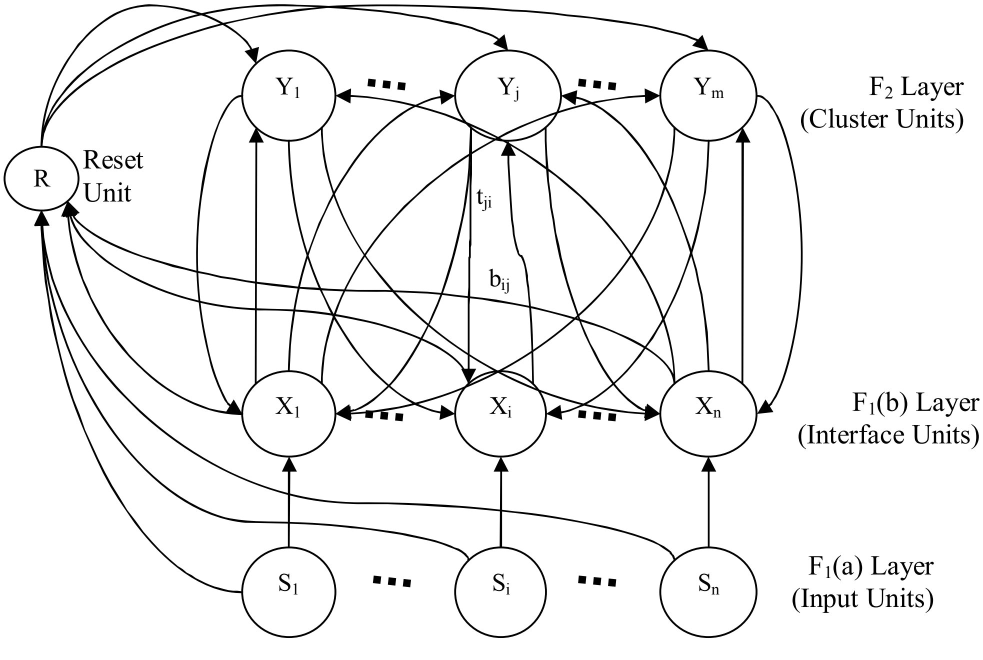


Figure 1 Basic Structure of ART1 [1]

weights) is the weight matrix from the interface layer to the cluster layer, t_{ji} (top-down weights) is the weight matrix from the cluster layer to the interface layer, n (number of input patterns or interface units), m (maximum allowed number of clusters), and ρ (vigilance parameter). The set of parameters b_{ij}, ρ, and m constitute the chromosomes, where bij is initialized by (L and a) from (1), which will be described later. The best set of design parameters is then the fittest amongst the chromosomes [L, a, ρ, m]. In this work, we will show the superiority of this design methodology compared to the conventional methods, which are based on recommended set of parameters.

The vigilance (ρ) and the maximum allowed number of clusters (m) have a big impact on the performance [1]:

1. High ρ and small m results in stable cluster formation after a few epochs of training while some input patterns cannot be placed in clusters.
2. High ρ and large m reduces the sensitivity to order of input with stable cluster formation after a few epochs.
3. Low ρ and small m requires more epochs to stabilize and it results in higher sensitivity to order of input.

The main objective of this work is to define a general and application independent measure for the quality of clustering, without previous knowledge of what the clustering would actually look like, and use it as a fitness function to guide the GA towards finding an optimal or suboptimal set of design parameters for the ART1.

Clustering is used heavily in text mining; applications of clustering in text mining include taxonomy generation, topic extraction, and grouping the hits returned by a search engine. Clustering can also be used to group textual information with other indications from business databases to provide novel insights [4]. Text clustering has a major significance in the search based applications like web-searching, since the fact that searching in clustered text sets instead of one set of documents runs faster. Supervised Text Categorization (TC) is the best method for such applications in terms of quality, but it suffers from some weakness compared with ART1 like expert's intervention, occasional need of retraining, and lack of adaptability [5].

II. Related Work

Research in this area has focused on some major issues like quality, space and time requirement, while only few considered application-independent architecture, learning algorithms, and performance. For example, Adaptive Resonance Theory under Constraints ART-C [6], [7], where dynamic variable value of vigilance parameter is applied, according to an extra constraint reset mechanism to the ART architecture. This concept was applied in ART 2A [7] to produce ART 2A-C. It was examined by clustering of gene expression data application. ART-C shows better performance than K-means, Self-Organizing Map (SOM), and conventional ART.

In [5], the author tests a simple ART1 network implementation and evaluates its text clustering quality on the Reuter data set by standard measures. He employs K-means clustering quality as lower bound and supervised TC as upper bound to publish his results relatively. He also applies incremental search for the best design parameters criteria to find the best setting for ρ and m.

Since ART was published, many approaches have been presented: improved ART1, adaptive Hamming net (AHN) by C. Hung and S. Lin [8], and Fuzzy ART, which are optimized in terms of space and time. In AHN, ART clustering scheme as an optimization problem was solved by finding the best matching unit in time by 4 defined equations.

The symmetric Fuzzy ART (S-Fuzzy ART) network is presented as a possible improvement over Fuzzy ART. Fuzzy ART is the best-known representative of the ART 1-based network group: (in fact, besides being viewed as a standalone system, Fuzzy ART is also known as the basic module of the Fuzzy ARTMAP classifier). However, Fuzzy ART has some weakness that can be summarized into three points: sensitivity to noise and outliers, inefficiency of category structures, and dependence of category structures upon data set input presentation [9].

The Simplified Adaptive Resonance Theory (SART) group of ART algorithms is defined as a generalization of S-Fuzzy ART. Gaussian ART (GART), which is a Gaussian maximum-likelihood (ML) probability density function estimator, is presented as one more instance of class SART. Results of the comparison between Fuzzy ART and S-Fuzzy ART may easily extend to the ARTMAP supervised learning framework in general and, in particular, to the Fuzzy ARTMAP classifier [9].

Projective ART (PART) neural network developed by Cao and Wu recently has been shown to be very effective in clustering data sets in high dimensional spaces [10]. The PART algorithm is based on the assumptions that the model equations of PART (a large scale and singularly perturbed system of differential equations coupled with a reset mechanism) have quite regular computational performance.

Genetic algorithms were used to optimize some types of ANN like the back-propagation [12], [13], [14], [15], and [16]. In this work, we design a genetic algorithm to search for the best design parameters for the ART1 ANN, with another challenge since the genetic algorithm itself uses a set of parameters that needs to be optimized as well [11].

III. ARTGEP

We design a fitness function that measures the performance of the ART1 and use it in the genetic search process as a guide towards the best set of parameters for a given data set. The test data set will consist of a group of web pages with features extracted as input to the proposed system.

We consider the Sensitivity of Order (*SoO*) of units to be clustered, a parameter to measure the fitness of the clustering. Usually, the clustering results depend on the order by which the data sets are presented to the ART network. Genetically engineering the parameters of the ART decouples the quality from the presentation order.

Could Not Cluster (*CNC*) units stand for the data sets that the clustering algorithm does not allocate to any cluster. The cardinality of the *CNC* can be used as a parameter to measure the quality of clustering. Usually designers update the ART parameters to overcome this problem. Employing a combination of ART1 and GA is expected to improve the clustering efficiency and quality, by tuning ART1 design parameters to minimize *SoO* and *CNC* problems.

The following stages summarize our approach:

A. Designing ART1

Basically, two weight matrices, described in Figure 1, form ART1 net: top-down weights matrix (t_{ji}), and bottom-up weights matrix (b_{ij}), while tji is initialized by 1's, b_{ij} is initialized by the following equation:

$$b(i,j) = a \times \frac{L}{L-1+n} \qquad (1)$$

Where: i = 1: n, j = 1: m, n is the number of input units, m is the max number of clusters, L and a are design parameters for initializing weight matrix.

To measure *SoO* parameter, we run ART1 twice: in-order and reverse-order, then apply the following equation:

$$SoO = \frac{match}{m} \qquad (2)$$

Where: *match* is the number of equal clusters from the two runs, we reverse the meaning of sensitivity to avoid zero in the denominator of the equation.

CNC is a counter that increments whenever the norm of input layer is zero; $\|s\|=0$, or when the winning cluster unit value = -1. Which means either the input pattern is all zeros (no features) or all cluster nodes are inhibited due to the reset activation caused by low vigilance ($\|x\|/\|s\| < \rho$). Both cases indicate an input unit that could not be clustered.

B. Designing GA

We propose the following fitness function:

$$F = \frac{1}{2} \times \left(SoO + \frac{m}{CNC + m} \right) \qquad (3)$$

We apply GA according to [3], [16] using a random population, all chromosomes enter into scaling function, so that every parameter is bounded by maximum and minimum values according to [17], [18], and [19] the values resolution reaches to 0.0001. The pseudo-code of the scaling function which updates design parameters is described as follows:

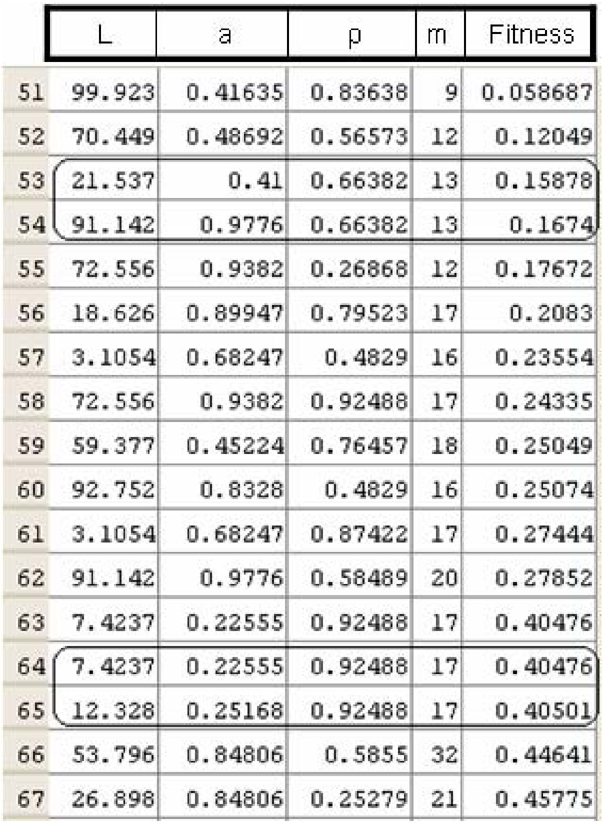
```
While ($L<1$) $L= L*10$;
While ($a>1$) $a= a/10$;
While ($\rho>1$) $\rho = \rho /10$;
While ($m<1$) $m= m*10$; integer ($m$);
```

Out of range values are rejected by setting fitness value to 0, as follows:

```
Fitness =0; If ($L=1$)
  Or ($r>1$) or ($r<= (1/n)$)
  Or ($a<=0$) or ($a>=1$)
  Or ($m<$min accepted value defined by user, for example
    $m= 4$) or ($m>$max accepted value defined by user,
    for example $m= d/3$, where $d$ is the number of input
    patterns).
```

Table I shows a sample of chromosomes from a random experiment. Fitness value calculated for all chromosomes by applying these chromosomes to the training set which equals to about 70% of the input data set [20]. After that, the population is sorted based on the fitness value. As we can see from Table I, tuning the values of L and a -while ρ & m are fixed- affects the fitness value for each chromosome. This explains how (L and a) impact the performance of the ART clustering.

TABLE I
SAMPLE OF CHROMOSOMES FROM RANDOM EXPEREMENT



	L	a	p	m	Fitness
51	99.923	0.41635	0.83638	9	0.058687
52	70.449	0.48692	0.56573	12	0.12049
53	21.537	0.41	0.66382	13	0.15878
54	91.142	0.9776	0.66382	13	0.1674
55	72.556	0.9382	0.26868	12	0.17672
56	18.626	0.89947	0.79523	17	0.2083
57	3.1054	0.68247	0.4829	16	0.23554
58	72.556	0.9382	0.92488	17	0.24335
59	59.377	0.45224	0.76457	18	0.25049
60	92.752	0.8328	0.4829	16	0.25074
61	3.1054	0.68247	0.87422	17	0.27444
62	91.142	0.9776	0.58489	20	0.27852
63	7.4237	0.22555	0.92488	17	0.40476
64	7.4237	0.22555	0.92488	17	0.40476
65	12.328	0.25168	0.92488	17	0.40501
66	53.796	0.84806	0.5855	32	0.44641
67	26.898	0.84806	0.25279	21	0.45775

Highlighted chromosomes shows the impact of L & a to the fitness value (clustering performance) when ρ & m are fixed

The GA is applied as the following steps:

1. Generate random population of chromosomes.
2. Each chromosome enters to the scaling function.
3. Calculate the fitness value of each chromosome.
4. Sort descending the population by the fitness value.
5. Select 4 parents (2 pairs) randomly from the top 10 chromosomes. Another pair is selected from the rest 10 of the top 20 chromosomes.
6. Each pair of parents enters to 2X2 crossover function to produce a new child (each child has the first half of the first parent, and the second half of the second parent).
7. The mutation rate defined by user (for example probability = 0.1) is applied to each produced child. Two kinds of mutation function are applied: either to add a small value to a random component of the child or to subtract a small value.

8. Each child enters to the scaling function, and the fitness value of them is calculated. This step makes the new child chromosomes ready to join the population.
9. The new 3 chromosomes are replaced with the smallest 3 chromosomes from the bottom of the population.
10. The population is resorted by the fitness value.
11. Check the stopping condition to stop or go to step 5.

The stopping condition is either to reach a threshold fitness value (defined by user for example 0.7) of the top chromosomes of the population or to reach a state where the top 10 chromosomes have the same fitness values. The top chromosome of the last population obtained by the algorithm is our target. This set of design parameters (*L, a, ρ, m*) will be applied for testing the rest 30% of input data set, and from now on, it will be applied for all new data sets, since ART1 has the on-line property which means that the network can continuously be trained once the network is detached from the training algorithm [2].

C. Input Data Set

The test data consists of a hundred of random web pages, with a feature extraction procedure similar to those in [21, 22 and 23] applied to prepare the data set, where only nouns and verbs are filtered and stored in a set without repeat, to produce a group of sets (D1, D2, …, Di, …, D100) where every set represents a web page. A vocabulary set (U) is generated from the union of all sets without repeating any word. This set is indexed from 1 to N, where N is the number of words. A binary matrix is generated by presenting all items in U as headers of columns, and all sets (D1, D2, …, Di, …, D100) as headers of rows. Now for every row Di, every matched word from Di with a word from U will be represented by a binary value 1. All other un-matched words will be filled with 0's. This binary matrix will be the input data set of the ARTgep algorithm.

IV. Experimental Results

Since the literature has shown that the ART is superior to both the K-means and the SOM [5], [6], it suffices to compare the ARTgep performance with that of the conventional ART. The training set has 2000 features (according to our text feature extraction software), we compare our set of design parameters with the recommended ones (L=2, a=0.5, ρ =1, m=d/3 = 23) from previous work [5], [17], [1]8, and 19], where ρ is estimated from Figure 2, *d* is the number of web pages and *m* is the maximum allowed number of clusters. We assume that this set is the best conventional ART1 for comparison purpose.

By running both: the conventional ART1 and ARTgep with the same training data set we obtained results in Table II. As in [5] we used Jaccard (JAC) [24] and Fowlkes-Mallows (FM) [25] quality measures equations:

$$JAC = \frac{A}{\sqrt{A+B+C}} \qquad (4)$$

$$FM = \frac{A}{\sqrt{(A+C)\times(A+C)}} \qquad (5)$$

Where [5]:

A: is the pair-wise number of true positives, i.e. the total number of document pairs grouped together in the expected solution and are indeed clustered together by the clustering algorithm.

B: is the pair-wise number of false positives, i.e. the number of document pairs not expected to be grouped together but are clustered together by the clustering algorithm

C: is the pair-wise number of false negatives, i.e. the number of document pairs expected to be grouped together but are not clustered together by the clustering algorithm.

Table II shows better results of ARTgep, where higher value of fitness function is better. Also, higher values of JAC and FW are reflecting the higher value of fitness function of ARTgep. This concludes that our fitness function from (1), is an indicator of clustering quality. This indicator does not need a prior knowledge of the text's real topic (cluster) to measure the quality of the clustering. That is very helpful for text mining applications, because of the need for unsurprised learning of such applications.

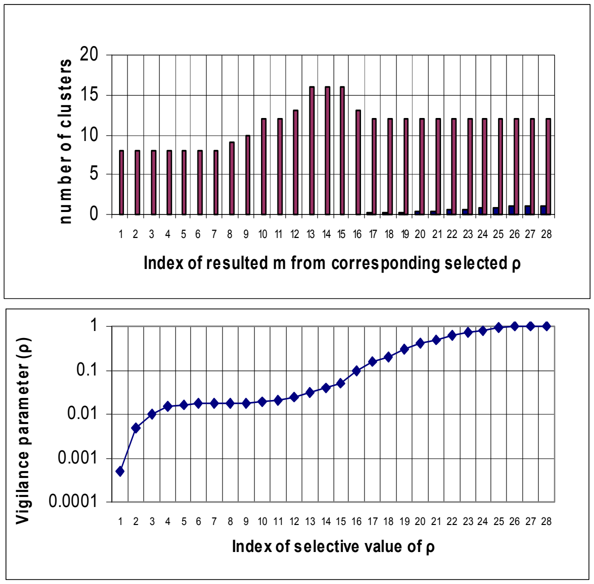


Figure 2: Select higher ρ with a corresponding acceptable number of clusters.

After adding the testing data set and running both generated ART design sets from previous experiment, we obtained results in Table III, which shows that ARTgep is superior, and we conclude that reducing the effect of *SoO* increases the quality and the generalization of ART1, since [5] states that *SoO* is case dependant.

If we run the conventional ART1 and ARTgep for all input data sets, we will get the results listed in Table IV. The fitness values of the last population of ARTgep after running all input data set using two population sizes; 30 and 100, are shown in Figures 3 and 4. The stopping condition for both experiments was repeating the GA 20 times. We noticed that using population size 30 results has a bit higher fitness value and 70% time saving. Also, the intervention of GA to find the targeted design parameters set is higher in the population size 30 than size 100; because when we use population size 100, the random initial generated chromosomes take bigger role than the randomness from the GA. So we recommend using small population in GA to obtain faster and better results.

TABLE II
RESULTS OF RUNNING TRAINING SET

Table 1	*L*	*a*	*ρ*	*m*	No. of clusters	fitness function	*A*	*B*	*C*	JAC	FM
Conventional ART Set	2	0.5	1	23	12	**0.5214**	87	391	493	**0.089598**	**0.165231**
ARTgep Set	95.7991	0.1923	0.8807	19*	14	**0.6579**	66	164	481	**0.092827**	**0.186074**

* Generated by ARTgep which allows m to be up-to d/3 = 23, fitness function is the new quality indicator, JAC & FM traditional indicators

TABLE III
RESULTS OF RUNNING TESTING SET

Table 2	*L*	*a*	*ρ*	*m*	No. of clusters	fitness function
Test Conventional ART	2	0.5	1	33	18	0.52
Test ARTgep	95.7991	0.1923	0.8807	33	18	0.5758

TABLE IV
RESULTS OF RUNNING ALL INPUT DATA SET

Table 3	*L*	*a*	*ρ*	*m*	No. of clusters	fitness function
Conventional ART for 100 web	2	0.5	1	33	18	0.52
ARTgep for 100 web	1.5274	0.4983	0.1556	33	14	0.6212

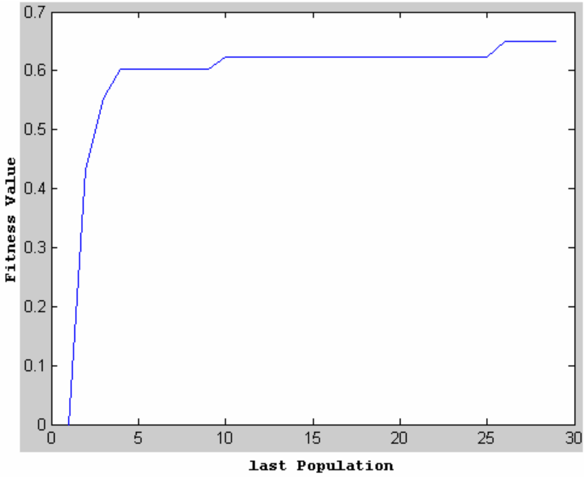


Figure 3: fitness values of the last population of ARTgep for all input data set using population size = 30 individuals

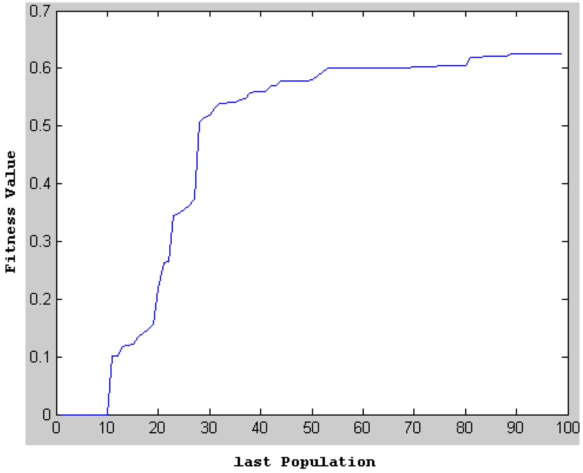


Figure 4: fitness values of the last population of ARTgep for all input data set using population size =100 individuals

V. CONCLUSION

Optimizing the ART1 ANN design parameters using genetic algorithms yields better performance in terms of the clustering quality. We develop a novel fitness function which used as a new clustering quality estimator. This quality measure is simpler of calculation, tracing time and complexity. The sensitivity to order of inputs gets minimized which is a big plus in the ART design in the quest for a more generalized system. Besides, the new quality estimator is application independent. Much more, it can be used for optimizing the ART1 design parameters during the design phase, since it doesn't need a proper knowledge of what the targeted clusters would look like. The experiments also, show the impact of initial bottom-up weight matrix of ART1 to the performance of its clustering.

Future work will concentrate on optimizing the genetic algorithms themselves before being used to optimize the ART ANN and apply the same concept to the ART2 ANN design parameters optimization; real numbers instead of just binary values of ART1.

REFERENCES

[1] Laurence Fausett, "Fundamentals of neural networks: architecture, algorithms, and applications", Prentice Hall, pp. (15, 16, 218–242), 1994.

[2] Imagination Engines Inc. "IEI's patented self-training artificial neural network object", 2005.

[3] Stuart Russell and Peter Norvig, "Artificial Intelligence A Modern Approach- 2nd edition", Local Search Algorithms and Optimization Problems- Genetic algorithms, Prentice Hall, pp. 116-119, 2003.

[4] Text Mining Using Oracle Data Mining, 10g Release 1, Oracle Corporation, 2003.

[5] L. Massey, "Evaluating Quality of Text Clustering with ART1", Royal Military College of Canada, *Advances in neural networks research, IJCNN'03, Pages: 771-778*, 2003

[6] J. He, A. Tan, C. Tan. ART-C: A neural architecture for self-organization under constraints. In *Proceedings of International Joint Conference on Neural Networks (IJCNN)*, pages 2550-2555, 2002.

[7] J. He, A. Tan, C. Tan. Self-organizing Neural Networks for Efficient Clustering of Gene Expression Data. In the *Proceedings of International Joint Conference on Neural Networks (IJCNN)*. July 2003. Portland, OR, USA. p1684-1689.

[8] C. Hung and S. Lin, "Adaptive Hamming Net: A fast-learning ART 1 model without searching," *Neural Networks, vol. 8, no. 4, pp. 605--618*, 1995.

[9] Andrea Baraldi and Ethem Alpaydın, "Constructive Feedforward ART Clustering Networks", *IEEE transactions on neural networks, vol. 13, no. 3*, May 2002.

[10] Yongqiang Cao and Jianhong Wu, " Dynamics of Projective Adaptive Resonance Theory Model: The Foundation of PART Algorithm", *IEEE Transactions On Neural Networks, Vol. 15, No. 2*, March 2004.

[11] Daniel W. Boeringer, Douglas H. Werner, Senior Member, IEEE, and David W. Machuga, Member, IEEE, " A Simultaneous Parameter Adaptation Scheme for Genetic Algorithms With Application to Phased Array Synthesis*", IEEE Transactions On Antennas And Propagation, Vol. 53, No. 1*, January 2005.

[12] G. Rovithakis, M. Maniadakis, M. Zervakis, "A geneticaly optimized Artificial Neural Network structure for Feature Extraction and Classification of vascular tissue fluorescence spectrums", *Proceedings of the Fifth IEEE International Workshop on Computer Architectures for Machine Perception*, 2000.

[13] Seung-Soo Han and Gary S. May, "Optimization of Neural Network Structure and Learning Parameters Using Genetic Algorithms", IEEE, 1996.

[14] Belinda Choi and Kevin Bluff, "Genetic Optimization of Control Parameters of a Neural Network", IEEE, 1995.

[15] Joerg W, Peter P., and Heribert P., "Genetically Optimized Neural Network Classifiers for Bankruptcy Prediction", *proceedings of 29th annual Hawaii international Conference on system sciences*, 1996.

[16] Xin Yao, "Evolving Artificial Neural Networks", *proceedings of the IEEE, 87(9):1423-1447*, September, 1999.

[17] Carpenter, G., and S. Grossberg. 1987. "A massively parallel architecture for a self-organizing neural pattern recognition machine". Computer Vision, Graphics and Image processing.

[18] Lippmann, R. 1987. An introduction to computing with neural nets. IEEE ASSP Magazine.

[19] L. Massey. "Determination of clustering tendency with art neural networks", In: *Proc. Of Recent Advances in Soft-Computing (RASC02)*, Nottingham, UK, Dec. 2002.

[20] David D. Lewis 1997. Reuters-21578 text categorization test collection, Distribution 1.0 README file (v 1.2), section VIII.A.

[21] The University of Waikato, "Weka 3: Data Mining Software in Java", open source software, version 3.4, 2005.

[22] Shay Cohen and Eytan Ruppin and Gideon Dror, "Feature Selection Based on the Shapley Value", 2005.

[23] Dmitry Davidov and Evgeniy Gabrilovich and Shaul Markovitch, "Parameterized Generation of Labeled Datasets for Text Categorization Based on a Hierarchical Directory", 2005.

[24] M. Downton and T. Brennan. "Comparing classifications: an evaluation of several coefficient of partition agreement", In*: Proc. Meeting of the Classification Soc.*, Boulder, CO, June 1980.

[25] E. Fowlkes and C. Mallows. "A method for comparing two hierarchical clusterings". *Journal of American Statistical Association, 78, pp. 553-569*, 1983.