Genetic Algorithm Based K-Means Fast Learning Artificial Neural Network

Yin Xiang, Alex Tay Leng Phuan[1]

Nanyang Technological University
[1] Contact e-mail: aslptay@ntu.edu.sg

Abstract. The K-means Fast Learning Artificial Neural Network (KFLANN) is a small neural network bearing two types of parameters, the tolerance, δ and the vigilance, μ. In previous papers, it was shown that the KFLANN was capable of fast and accurate assimilation of data [12]. However, it was still an unsolved issue to determine the suitable values for δ and μ in [12]. This paper continues to follows-up by introducing Genetic Algorithms as a possible solution for searching through the parameter space to effectively and efficiently extract suitable values to δ and μ. It is also able to determine significant factors that help achieve accurate clustering. Experimental results are presented to illustrate the hybrid GA-KFLANN ability using available test data.

1 Introduction

K-Means Fast Learning Artificial Neural Network (KFLANN) has the ability to cluster effectively, with consistent centroids, regardless of variations in the data presentation sequence [6], [7], [12]. However, its search time on parameters δ and μ for clustering increases exponentially compared to the linear increase in the input dimension. A Genetic Algorithm (GA) was used to efficiently orchestrate the search for suitable δ and μ values, thus removing the need for guesswork. The hybrid model, GA-KFLANN, shows that the technique indeed has merit in fast completion as well as accurate clustering. Although the δ and μ values obtained provided sub-optimal clustering results, these results were still within acceptable clustering tolerance. This paper also provides an introduction to the K-means Fast Learning Artificial Neural Network (KFLANN) and a description of how the Genetic Algorithm was weaved into the algorithm to support the effective search for the required parameters.

1.1 The KFLANN Algorithm

The basic architecture of the KFLANN is shown in Figure 1 [3], [4], [5]. It has 2 layers, the input and output layer, and a set of weight vectors connecting the 2 layers. The KFLANN is a fully connected network.

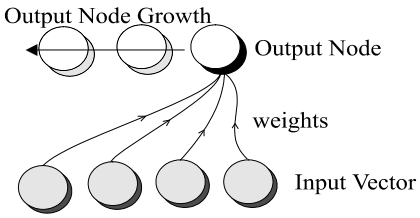

Figure 1. The Architecture of KFLANN

The number of output nodes can increase according to the classification requirements, determined indirectly by the δ and μ parameters. As each new cluster is formed, a new output node is created and the weight vectors of the new output node are assimilated with the exemplar values. The algorithm of the KFLANN follows.

1.1.1 Algorithm of KFLANN

Notation
- μ: vigilance value
- δ_i: tolerance value of the i^{th} attribute
- n: the number of input attributes
- I_i: the i^{th} input node
- W_{ji}: weight connecting the i^{th} input node and the j^{th} output neuron

D[a] = 1 if a > 0. Otherwise D[a] = 0.

1 Initialize network with μ between 0 and 1. Determine and set δ_i for i = 1, 2, 3, …, n. The values of μ and δ affect the behaviors of the classification and learning process.

2 Present the next pattern to the input nodes. If there are no output clusters present, GOTO 6.

3 Determine the set of clusters that are possible matches using equation (1). If there are no output clusters GOTO 6.

$$\frac{\sum_{i=1}^{n} D\left[\delta_i^2 - (W_{ji} - I_i)^2\right]}{n} \geq \mu \qquad (1)$$

4 Using criteria in equation (2) determine the winning cluster from the match set from Step 3. Normalize W_{ji} and I_i. The following distance is calculated between the normalized versions.

$$winner = \arg\min_j \left[\sum_{i=0}^{n} (W_{ji} - I_i)^2\right] \qquad (2)$$

5 When the Winner is found. Add vector to the winning cluster. If there are no more patterns, GOTO 7. Else GOTO 2.

6	No match found. Create a new output cluster and perform direct mapping from input vector into weight vector of new output cluster. If there are no more patterns, GOTO 7. Else GOTO 2.
7	Re-compute cluster center using K-means algorithm. Find the nearest vector to the cluster center in each cluster using equation (2). Place the nearest vector in each cluster to the top of the training data and GOTO 2.

After each cycle of clustering with all exemplars, the cluster centers are updated using K-means algorithm. This is *Step 7* of the KFLANN algorithm. A comparison between each cluster center and patterns in respective cluster is then conducted to determine the nearest point to each cluster center. The algorithm then assigns this point as the new centroid.

1.1.2 Parameter Search for δ and μ

The KFLANN algorithm is able to cluster effectively only if the correct δ and μ values are used [7]. As illustrated in Figure 2, the δ values indirectly determine the clustering behaviour of the algorithm. A larger δ provides lesser clusters (a), while a smaller δ provides more clusters (b). Since the characteristic spread of data is sometimes unknown, the δ values are still very much a guessing game. The results of a brute-force combinatorial exhaustive search [12] are used to compare with the results obtained from the GA search presented in this paper. This original brute-force algorithm tries all possible combinations of tolerance and vigilance values. For example, if there are n attributes in an exemplar and each tolerance has m steps on its value range, m^n modifications have to be made on tolerance values totally. The high price of the exhaustive search provides the motivation for better alternatives.

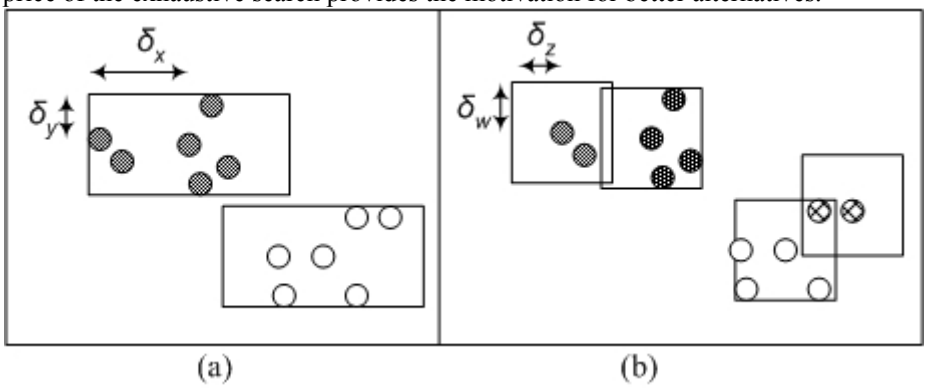

Figure 2. Different clusters are formed for the same data set when δ is varied

1.2 The Genetic Algorithm

Genetic Algorithms are guided, yet random search algorithms for complex optimization problems and are based on principles from natural evolutionary theory.

GAs are computationally simple yet powerful and do not require the search space to be continuous, differentiable, unimodal or of a functional form.

The GA process is illustrated in Figure 3. To obtain solutions, the problem space is initially encoded into a relevant format, suitable for evolutionary computation. The parameters of the search space are encoded in the form known as *chromosomes* and each indivisible parameter in a chromosome is called a *gene*. A collection of such strings is called a *population*. Initially, a random population is created, which represents different points in the search space. An *objective* and *fitness* function is associated with each string that represents the degree of *goodness* of the chromosome. Based on the principle of survival of the fittest, a few of the strings are reproduced and each is assigned a number of copies that go into the mating pool. Biologically inspired operators like *crossover* and *mutation* are applied on these strings to yield a new generation of chromosomes. The process of reproduction, crossover and mutation continues for a fixed number of generations or till a termination condition is satisfied. [10]

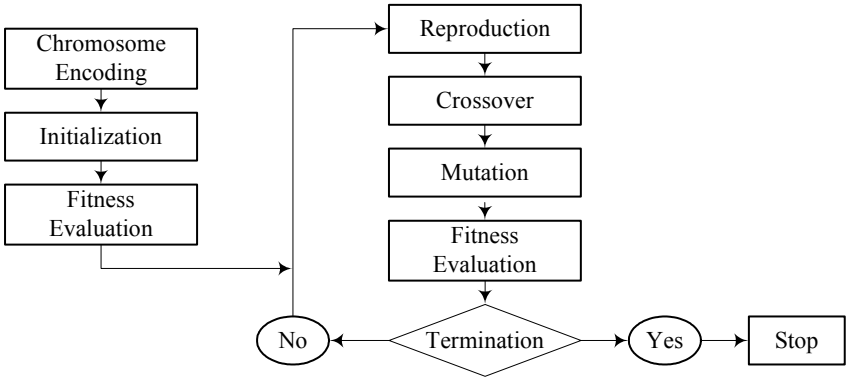

Figure 3. The GA process

GA is useful when a sub-optimal solution is sufficient. The self-evolving nature and likeliness to reach a near-optimal condition regardless of dimensionality, is the strong motivation for introducing GA into KFLANN.

1.2.1 Chromosome Encoding

The chromosomal encoding for the GA-KFLANN consists of two parts: the control genes (ConG) and the coefficient genes (CoeG). The ConG are a string of binary numbers, which are used to turn on or off the corresponding features to achieve the goal of feature selection. Whereas, the CoeG are a sequence of real numbers representing tolerance and vigilance values to control the behaviour of KFLANN. The ConG may not be used when all features are fully utilized in a clustering problem. For the purpose of discussion, the Iris flower dataset is now used as an example to illustrate the encoding required for GA computations. The δ and μ are first converted into chromosomal equivalents as shown in Figure 4. CoeG shown in shaded pattern represent those turned off by their corresponding ConG.

1.2.2 Population Initialization

In the 1st generation of the GA, ConG (if used) are randomly assigned to the value '1' or '0', while CoeG are randomly initialized to values between the upper bound and the lower bound of tolerance or vigilance of features from the input data. Since tolerance value (δ) is the maximum distance of how far a pattern in a cluster can be

Population									
	Control Genes				Coefficient Genes				
					Tolerance δ				Vigilance μ
					Sepal length	Sepal width	Petal length	Petal width	
1	1	1	1	0	0.88	0.56	0.40	0.39	0.75
2	0	0	1	1	0.54	0.60	0.17	0.47	0.5
⋮									
n	1	1	1	1	0.73	0.65	0.34	0.12	1.0

Figure 4. Sample Chromosome Encoding for the Iris Dataset

from the cluster center in each attribute [12] as shown in Figure 2, the upper bound of tolerance for each attribute can be set to half of the maximum distance among data points in that attribute, while the lower bound can be assigned to the minimum distance. For example, assume there are 5 data points: (0, 0), (1, 1), (3, 4), (5, 0), and (2, 6). The upper bound for the 1st dimension is (5-0)/2 = 2.5, while the lower bound is 1, which is the minimum distance among data points. Therefore, tolerance δ_1 shall be initialized in the range [1, 2.5]. Similarly tolerance δ_2 can be found in the range [1, 3].

1.2.3 Fitness Evaluation

Fitness evaluation of the clustering results is the key issue for GA. A good fitness evaluation can make GA produce proper tolerance and vigilance values and lead KFLANN to the optimal clustering, while a poor one can cause GA to converge towards a wrong direction and lead to inaccurate clustering.

Within-group variance $\sigma_{W_i}^2$ for each cluster i and between-group variance σ_B^2 can be easily computed from the output of KFLANN for each clustering result. And the two types of variance satisfy the following equation:

$$\sigma_T^2 = \sigma_W^2 + \sigma_B^2 \qquad \sigma_W^2 = \sum_{i=1}^{k} \sigma_{W_i}^2 \tag{3}$$

where σ_T^2 is the total variance. Since σ_T^2 is fixed for a data set, a natural criterion for grouping is to minimize σ_W^2, or, equivalently, maximize σ_B^2 [1]. Moreover, the clustering with the maximum between-group variance and minimum within-group

variance means highly dense clusters and good data compression. Thus, a possible evaluation criterion can be formed as maximizing the term: σ_B^2 / σ_W^2.

It works reasonably well for data sets without overlapping patterns, but not so well as expected with overlapping clusters. An additional term used in fitness evaluation is a Boolean variable, convergence, representing whether a clustering converges. The whole term is expressed as follows:

$$fitness = (convergence + 1) \times \sigma_B^2 / \sigma_W^2 \tag{4}$$

This is to ensure that converged clustering has much higher fitness value and force the GA to produce tolerance and vigilance that can make KFLANN converge and form consistent centroids.

1.2.4 Reproduction

Solution strings from the current generation are copied into a mating pool according to the corresponding fitness values. Strings with higher fitness values will likely be represented in higher numbers in the mating pool, which means δ and μ generating higher clustering accuracy will more likely survive and pass their values to the next generation. This is because that there is a higher chance for δ and μ with higher clustering accuracy to hit the respective correct settings. Stochastic Universal Sampling (SUS) is the most popular reproduction strategy and utilized in this paper.

1.2.5 Crossover

Crossover is a probabilistic process that exchanges information between two parent chromosomes in the mating pool for generating two child chromosomes, so that proper settings of parameters can be grouped together into a single child chromosome. For example, one parent has the best setting of sepal length, while another has proper petal width value. A better clustering result will be achieved if the 2 good settings can be grouped together into just one offspring.

Two types of crossover operators are implemented in this paper since the ConG and CoeG make use of different encoding schemes. Uniform crossover is applied to the ConG while convex crossover is applied to the CoeG. In uniform crossover a template of the same length as ConG is randomly generated to decide which parent to contribute for each bit position. For example, 2 parents are shown in Table 1, one is underlined and the other is italic. Bits from parent 1 are passed to offspring 1 if the corresponding bits in the template are of value '1'; otherwise those bits are passed to offspring 2. This rule works in reverse way for parent 2. Therefore, the 1st four and last two bits of parent 1 are passed to offspring 1, while the rest goes to offspring 2. Similarly, parent 2 contributes different parts to offspring 1 and 2 respectively according to the template.

Table 1. An Example of Uniform Crossover

	Parent	Template	Offspring
1	1001011	1111001	1001*10*1
2	*0101101*		*0101*011

If x and y are the CoeG of two parents, then convex crossover is of the following form:

$$x' = \lambda x + (1-\lambda) y \tag{5}$$

$$y' = (1-\lambda) x + \lambda y \tag{6}$$

where x' and y' are the corresponding CoeG of two children. λ is set to 0.7 in this paper.

1.2.6 Mutation

Mutation operation randomly picks up a gene in the generated offspring strings and changes its value properly in order to allow the GA to escape from a local optimal to search for a global optimal. Two types of mutation operators are used in this paper like the described crossover above. Normal mutation is applied to the ConG while dynamic mutation is applied to the CoeG.

For a given string, if the gene x is selected for mutation, then the offspring x' = 1 – x if x is a control gene. If x is a coefficient gene, then x' is selected with equal probability from the two choices:

$$x' = x + r(u-x)(1-\frac{t}{T})^b \qquad x' = x - r(x-l)(1-\frac{t}{T})^b \tag{7}$$

$x \in [l,u]$

r: a random number chosen uniformly from [0, 1]
t: current generation number
T: the maximum number of generations
b: degree of nonuniformity.

1.2.7 Population Replacement

It is possible that offsprings become weaker than the parents as some good genes in the parents may be lost. Therefore, elitism strategy is used by copying the best or best few parents into the next generation to replace the worst children.

1.3 Hybrid Model of GA-KFLANN

The architecture of the GA-KFLANN is illustrated in Figure 5. The original KFLANN takes tolerance and vigilance values produced by the GA to cluster the input data set with selected features by GA. After the KFLANN converges or a predefined number of cycles have been reached, the fitness values of the clustering results are evaluated and fed back to GA to generate the next population of better parameters. This process continues until a preset number of generations of GA have been reached or no much improvement on the fitness value can be observed.

2 Experiments and Results

2.1 Iris Data Set

Fisher's paper [8] is a classic in the field and is referenced frequently to this day. The data set contains 150 random samples of flowers from the Iris species: Setosa, Versicolor, and Virginica. From each species there are 50 observations with 4 attributes each in centimeters.

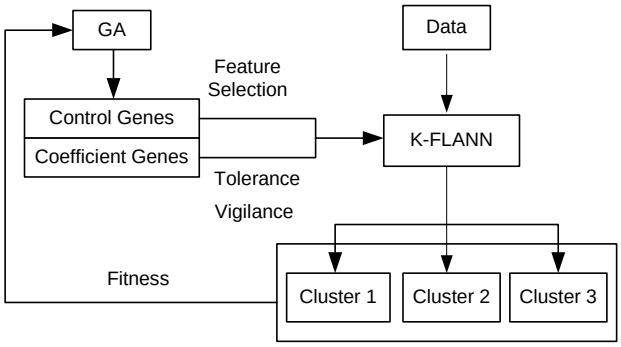

Figure 5. The Architecture of GA-KFLANN

2.1.1 Results without Control Genes (ConG)

Test results of data mining are exercised without ConG on the Iris data, which means that feature selection is turned off, and the best 4 outcomes of a run are shown in Table 2.

Table 2. The Best 4 Results of a Run of Iris Data Clustering without ConG

# Of Clusters	Fitness	Accuracy
4	3.95417	85.3%
3	3.85691	88.0%
3	3.52583	86.7%
2	3.32374	66.7%

There is a nonlinear relation between fitness and accuracy because Versicolor and Virginica are not linearly separable from each other. This makes the fitness evaluation as mentioned previously function poorly since maximizing the between group variance *B* does not work properly. Therefore, clustering with higher fitness may not have higher accuracy. Row No. 2 has the clustering with the highest accuracy and desired number of clusters.

Table 3 shows the comparison between the GA-KFLANN and the exhaustive search on Iris data set. The highest accuracy considered here includes not only accuracy itself in Table 2 but also the number of clusters.

Table 3. Comparison between GA-KFLANN and Exhaustive Search on Iris Data Clustering

	Accuracy	Completion Time
GA-KFLANN	88.0%	< 1 minute
Exhaustive Search	96%	5 minutes

The exhaustive search yielded better accuracy but required more processing time on Iris clustering. Another consideration is that the exhaustive search actually did feature selection as well but the GA-KFLANN did not. Therefore, the exhaustive search is expected to have higher accuracy but the GA-KFLANN has greater potential in both completion time and accuracy.

2.1.2 Results with Control Genes (ConG)

Table 4 shows the best 4 results in a run of Iris data clustering with ConG and the table is sorted according to the fitness of clustering. The attributes with a tick "√" indicate the presence of the attribute to achieve the accuracy. The number of clusters is recorded in the first column.

Table 4. The Best 4 Results of a Run of Iris Data Clustering with ConG

# Of Clusters	Fitness	Accuracy	Sepal Length	Sepal Width	Petal Length	Petal Width
6	20.7	72.7%	√			
4	13.5	86.7%			√	
3	12.5	89.3%				√
3	10. 8	96%			√	√

It is clear that the last two rows have higher accuracy and the desired number of cluster. Petal width provides most information in clustering comparing to other features of Iris. Therefore, petal width is a main factor in determination of the Iris classification and the GA-KFLANN was able to perform well in both feature selection and clustering on the Iris data.

Table 5 shows the comparison among the GA-KFLANN, the exhaustive search and K-Nearest Neighbour (K-NN) on Iris data set. All 3 methods achieved pretty high accuracy, but the GA-KFLANN showed superior potential on effective and efficient search because it took much less time for completion.

Table 5. Comparison of Different Clustering Algorithms on Iris Data

	Accuracy	Completion Time	Reference
GA-KFLANN	96%	< 1 minute	
Exhaustive Search	96%	5 minutes	[12]
K-NN	95.1%		[9]

2.2 Wine Data

This data was obtained from a chemical analysis of wines grown within the region of Italy, but were derived from three different cultivators. The analysis determined the quantities of 13 constituents found in each of the three types of wines. There were 178 instances of wine samples in the data set.

2.2.1 Results without Control Genes (ConG)

The results in Table 6, of wine clustering without using ConG took only 2 minutes to generate. The highest accuracy with the correct number of clusters was however only 70.2%. In comparison, the exhaustive search on wine data set achieved 95.51% [12]. The exhaustive search however yielded this high accuracy at the expense of speed, which took 2 weeks to solve 5 attributes.

Table 6. The Best 5 Results of a Run of Wine Data Clustering without ConG

# Of Clusters	Fitness	Accuracy
3	6.21352	64.6%
3	5.86033	70.2%
3	4.42048	65.2%
4	3.54305	66.9%
3	2.34613	68.0%

2.2.2 Results with Control Genes (ConG)

Table 7 shows the results of wine clustering with ConG enabled to select features.

Table 7. The Best 4 Results of a Run of Wine Data Clustering with ConG

Attributes													Fitness	Accuracy
1	2	3	4	5	6	7	8	9	10	11	12	13		
√		√			√	√			√	√			1.21	60.11%
√	√	√			√		√	√				√	0.765	58.42%
	√		√	√	√		√						0.716	39.89%
√			√			√			√	√			0.669	90.44%

Alcohol (Item 1), Alcalinity of ash (Item 4), Flavanoids (Item 7), Colour intensity (Item 10), Hue (Item 11)

The highest accuracy achieved currently was 90.44% and this was achieved in 2 minutes. In comparison, the highest accuracy achieved in the exhaustive search was 95.51% in 2 weeks. Results of a K-NN in [9] achieved 96.7% accuracy. The features discovered to be significant using exhaustive search were Flavanoids (Item 7), Colour intensity (Item 10) and Proline (Item 13).

It is clear that the exhaustive search can achieve higher accuracy in clustering and locate the most significant factors, while the GA-KFLANN has relatively lower accuracy and more factors selected due to its random evolutionary nature. However, the exhaustive search also takes unacceptable long time to complete. Therefore, the GA-KFLANN shows greater potential in clustering as well as feature selection.

2.3 Wisconsin Breast Cancer Data

This breast cancer databases was obtained from the University of Wisconsin Hospitals, Madison from Dr. William H. Wolberg [11]. There were 699 patterns from 2 classes and 9 attributes. Table 8 shows the results of breast cancer clustering sorted according to fitness value.

Four results out of five have higher than 90% accuracy and the highest one achieved is 95.6%. The last row of Table shows the number of appearance for each attributes and it is clear that attribute 3, 4, 5 and 9 appear more frequently than others.

Table 8. The Best 5 Results of a Run of Wisconsin Breast Cancer Data Clustering

Attributes									Fitness	Accuracy
1	2	3	4	5	6	7	8	9		
	√			√	√		√	√	20.3	90.7%
√		√	√			√			16.8	89.4%
	√	√	√	√				√	7.34	91.5%
	√	√	√	√	√		√	√	7.34	91.5%
		√	√	√		√		√	2.81	95.6%
1	3	4	4	4	2	2	2	4		

Therefore, some evidence from the clustering results supports that the 4 attributes likely to be the significant in representing the dataset were, Uniformity of Cell Shape (Item 3), Marginal Adhesion (Item 4), Single Epithelial Cell Size (Item 5), Mitoses (Item 9).

The GA-KFLANN seemed to perform well in this data set. As there were too many attributes, it was not viable to conduct an exhaustive search. However, a comparison was made with K-NN, which achieved 96.6% accuracy on this data set [2]. Both performed pretty good clustering.

3 Conclusions

Although the data used were from well-known sources which have been investigated by many, the emphasis of the experiments was on the technique which was used to boost searching, extract the features from the data and get accurate clustering. From the 3 data sets, the analysis resulted in the determination of significant factors. This information was extracted without the need to have an understanding of the data. Further investigations are underway to determine if there is a proper fitness evaluation method to guide the search of GA for optimal clustering parameters.

References

[1] B. Everitt, *Cluster Analysis*, 2nd ed., New York, Halsted Press, 1980.

[2] B. Ster and A. Dobnikar, Neural Networks in Medical Diagnosis: Comparison with Other Methods,, In A. Bulsari et al., editor, Proceedings of the International Conference EANN '96, pp 427-430, 1996.

[3] D. J. Evans and L. P. Tay, *Fast Learning Artificial Neural Networks for Continuous Input Applications*, Kybernetes, Vol. 24, No. 3, 1995.

[4] L. P. Tay and D. J. Evans, "Fast Learning Artificial Neural Network (FLANN II) Using Nearest Neighbour Recall", *Neural Parallel and Scientific Computations*, Vol. 2, No. 1, 1994.

[5] L. P. Tay and S. Prakash, K-Means Fast Learning Artificial Neural Network, an Alternative Network for Classification, ICONIP, 2002.

[6] L. P. Wong and L. P. Tay, Centroid Stability with K-Mean Fast Learning Artificial Neural Networks, IJCNN, Vol.2, pp 1517 – 1522, 2003.

[7] L. P. Wong, J. Xu and L. P. Tay, Liquid Drop Photonic signal using Fast Learning Artificial Neural Network, ICICS, Vol.2, pp. 1018- 1022, 2003.

[8] R. A. Fisher, "The Use of Multiple Measurements in Taxonomic Problems", *Annual Eugenics*, 7, Part II, 179-188, 1936.

[9] S. D. Bay, Combining Nearest Neighbor Classifiers through Multiple Feature Subsets, Proc. 17th Intl. Conf. on Machine Learning, pp. 37-45, Madison, WI, 1998.

[10] U. Maulik and S. Bandyopadhyay, "Genetic Algorithm-Based Clustering Technique", *Pattern Recognition*, Vol. 33, No. 9, pp. 1455-1465, 2000.

[11] W. H. Wolberg and O. L. Mangasarian, "Multisurface Method of Pattern Separation or Medical Diagnosis Applied to Breast Cytology", *Proceedings of the National Academy of Sciences*, U.S.A., Vol. 87, pp 9193-9196, 1990.

[12] X. Yin and L. P. Tay, Feature Extraction Using The K-Means Fast Learning Artificial Neural Network, ICICS, Vol.2, pp. 1004- 1008, 2003.