ENHANCING CLUSTERING PERFORMANCE OF FEATURE MAPS USING RANDOMNESS

[1]Rasika Amarasiri, [1]Damminda Alahakoon, [2]Malin Premaratne and [1]Kate Smith
[1]School of Business Systems /
[2]Department of Electrical and Computer Systems Engineering
Monash University, Clayton, VIC, Australia
Rasika.Amarasiri@infotech.monash.edu.au

Abstract – *This paper presents an enhancement made to a high dimensional variant of a growing self organizing map called the High Dimensional Growing Self Organizing Map (HDGSOM) that enhances the clustering of the algorithm. The enhancement is based on randomness that expedites the self organizing process by moving the inputs out from local minima producing better clusters within a shorter training time. The enhancement is described in detail and several experiments on very large text datasets illustrating the effect of the enhancement are also presented.*

Keywords – Randomness, Growing Feature Maps, High Dimensions, HDGSOM*r*, HDGSOM, GSOM

1 Introduction

The Self Organizing Map (SOM) [1] developed in the early 1980's by Kohonen has paved the way to many algorithms commonly known as feature map algorithms. Feature maps are very commonly used in data mining applications due to its ability to present the data clustered in a visual structure that reveals the relationship between the clusters [2]. The algorithms developed based on the SOM range from hierarchical SOMs such as WEBSOM [3] and GHSOM [4] to growing feature maps such as GHSOM [4], Growing Grid [5], Incremental Grid Growing [6], DASH [7] and GSOM [8]. Feature maps have been applied in almost all areas of research and the collection of references in chapter 7 of [2] is ample evidence to this.

Randomness is commonly associated with feature maps. The most common applications of randomness in feature maps have been in initializing the weights of the nodes and presenting the inputs in a random order. It has also been applied in text mining to reduce the dimensionality of the input data [9]. We have so far not found any work that has applied randomness on the actual self organizing process.

This paper presents the application of randomness in the actual self organizing process of a high dimensional growing self organizing map called the High Dimensional Growing Self Organizing Map (HDGSOM) [10]. The enhancement to the HDGSOM using randomness in the actual self organizing process was tested on several data sets and results of these experiments are also presented. The paper is organized as follows: Section 2 of the paper introduces the concept of randomness and its applications in other algorithms. Section 3 briefly introduces the

HDGSOM algorithm and the modifications proposed. Experimental results are presented in Section 4 and the conclusions and future work are presented in Section 5.

2 Randomness

Randomness is commonly used to express the apparent lack of purpose or cause. This suggests that no matter what the cause of something, its nature is not only unknown, but the consequences of its operation are also unknown [11]. Nature provides the best randomness in the environment and has lead to both destructions and survival of the fittest. The inspirations of observing nature's random selection of the fittest organism to lead the future of its kind has lead to many algorithms.
Genetic Algorithms (GA) [12] involve random mutations and cross-over operations that simulate the random selection in nature. The result is a faster and more robust solution for problems such as constraint satisfaction problems. The use of randomness in GAs allows the algorithm to escape from local minima that tend to produce non optimal results in other algorithms [13].
Simulated Annealing [14] is another widely used algorithm heavily using randomness. It simulates the slow cooling process of heated metal that ultimately end up at the ordered frozen ground state. This too is known to produce results that are globally optimized rather than locally optimized.
The use of randomness is common in the SOM algorithm in many stages including the random initialization of the weights and the random ordering of the input data. There have been instances where the selection of the winner neuron had been modified randomly [15]. We have so far not found any instances of the randomness being used in the weight adaptation of actual self organizing process. The successes of other algorithms using randomness lead us to test the applicability of randomness in the actual self organizing process to overcome problems of getting the weight vectors of the map stuck in local minima. The randomness in the self organizing process was applied to a high dimensional growing variant of the SOM called the HDGSOM which will be briefly explained in the next Section.

3 High Dimensional Growing Self Organizing Map

The High Dimensional Growing Self Organizing Map (HDGSOM) [10] is an extension of the Growing Self Organizing Map (GSOM) [8] that was modified to address issues of the GSOM when applied to high dimensional data. The original GSOM algorithm which grows outward starting from a minimal number of nodes (usually 4) resulted in rapid growing of nodes that produced warped and twisted maps when applied on high dimensional data such as those in text mining. This was mainly due to the larger errors that can get generated with very large dimensions.

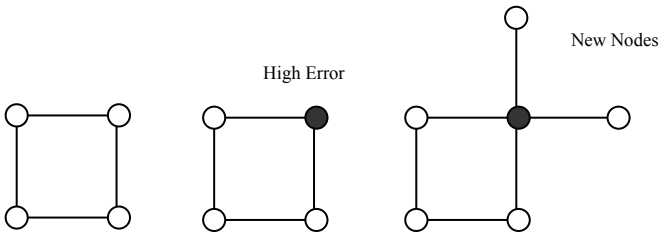

Figure 1. New node growth in HDGSOM algorithm.

The HDGSOM algorithm has a growing phase that reduces a growth threshold value in several stages interleaved with several smoothing phases that lead to a smoother growing. The growing phase is then followed by more smoothing phases that resemble the operations of the SOM

resulting in a better feature map than the original GSOM algorithm. The HDGSOM in brief is as follows.

3.1 Initialization Phase

The HDGSOM is created with 4 nodes connected to each other in a rectangular shape as in Figure 1 and initialized with random weight vectors. Since many high dimensional datasets such as those in text are very sparse, the growth thresholds used in the growing phase are calculated using the non-zero dimension distribution of the dataset. This involves calculating the average (μ) and standard deviation (σ) of the non zero dimensions of all the inputs. A non zero dimension of an input is an attribute in the input that has a value greater than zero.

Two growth threshold values, (GT_1 and GT_2) are calculated to be used in the growing phase of the algorithm: $GT_1 = -(\mu + 2\sigma) \times \ln(SF) \times 50$ and $GT_2 = -\ln(\mu + 2\sigma) \times \ln(SF) \times 50$ where *SF* is a parameter ranging between 0 and 1 called the Spread Factor that allows the data analyst to control the spread of the map. Higher values of *SF* generate larger maps giving detailed clusters while smaller *SF* values produce denser clusters. The *SF* is usually set at 0.1.

3.2 Growing Phase

During the growing phase of the HDGSOM, inputs are presented to the nodes as in the SOM algorithm and the winner node is identified. The error between the input and the weight vector is used to update the weights of the winner node and its neighbours similar to the SOM. If the accumulated error in a node exceeds a threshold called the growth threshold (GT explained next) and the node is a boundary node, then new nodes will be grown in all the vacant neighbouring positions as illustrated in Figure 1.

The growth threshold *GT* used to decide on when to grow new nodes varies from GT_1 to GT_2 in equal steps after each growing epoch. The growing epochs are interleaved with 2-3 smoothing epochs, which operate similar to the SOM. The error accumulated in each node is reset at the end of each epoch. New nodes cannot grow from nodes that have not been in the map for at least one epoch.

3.3 Smoothing Phases

The growing phase is followed by two smoothing phases that are almost similar to the smoothing phases of the SOM. These two phases have diminishing learning rates (α) that smooth out the map to produce crispier clusters.

The HDGSOM algorithm has been evaluated over several hundred datasets varying in size and content. The experiments resulted in several improvements that will be available in [16]. The work presented in this paper builds upon those improvements.

3.4 Introduction of Randomness to the HDGSOM

During the many experiments on the HDGSOM, it was identified that in order to generate a smoother map, the map had to undergo longer smoothing periods. This meant longer processing times, which were very costly for large datasets with huge dimensions. The smoothing phases were more efficient if they were applied earlier in the algorithm with higher learning rates that caused the weight vectors to update with larger values. The better results

with the larger learning rates indicate that the algorithm is getting stuck at local minima when using lower learning rates.

Inspired by the many achievements of using randomness in other algorithms to overcome problems associated with local minima, it was decided to evaluate the effect of introducing randomness to the self organizing process. This was achieved by updating the weights of the winner node and the neighbouring nodes with a fraction of the error modified by a random value. The resulting update would randomly increase or decrease the amount of training the nodes get via the self organizing process.

The easiest way of achieving this was to modify the learning rate (α) by a random number. The modified weight updation equation is: $w_i^{new} = w_i^{old} + [\alpha + (r - 0.5) \times \alpha \times 2] \times \eta \times (x_i - w_i^{old})$ where w_i^{new} is the updated weight of the i^{th} component of the weight vector of the node, w_i^{old} is the weight of the i^{th} component of the weight vector before updation, x_i is the value of the i^{th} component of the input, α is the learning rate and r is a random number in the range of 0 and 1. η is the neighbourhood kernel contribution to the weight. It is 1 for the winner and approaches zero over a Gaussian distribution as the neighbour distance increases. The equation modifies the learning rate α by increasing or decreasing it by a fraction of itself. New random numbers are generated for each updation. The selection of a random number between 0 and 1 was to simplify the computation of the random number and was experimentally found to be the most effective value for the given algorithm over many data sets.

The modified weight updation equation was applied to all self organizing sections of the HDGSOM algorithm. The modified algorithm was then tested on several large datasets using the unmodified HDGSOM algorithm and the modified HDGSOM algorithm, which we will refer to as HDGSOM*r*. The experiments are described in the next Section.

4 Experiments on the Modified HDGSOM algorithm

The modified HDGSOM algorithm (HDGSOM*r*) was tested on several benchmark text mining datasets to evaluate the effect of introducing randomness to the self organizing process. The results with only two of these datasets are presented in this Section due to space limitations and the ease of evaluating the results. The two datasets used are the 20 newsgroups dataset [17] and the 19 newsgroups dataset [18]. Both these datasets have labels to identify to which newsgroup the input belongs to that allowed us to easily evaluate the resulting maps for the quality of clustering. Each dataset had approximately 1000 newsgroup postings from each newsgroup.

The text in the datasets were extracted and cleaned to remove any symbols and numbers. The resulting text was split into words, stemmed using the Porter's stemming algorithm [19]. Some commonly used words such as 'of', 'the', 'and' were removed using a stop list. Removing words that did not occur in more than 100 postings further reduced the list of words. The resulting words were then encoded using TF-IDF method [20] and normalized to produce inputs to the algorithm.

In order to see if there is an improvement in the clustering, the number of epochs in each round of the HDGSOM algorithm and HDGSOM*r* algorithm were reduced. This would result in a map with lesser quality clusters for the HDGSOM and if there is an improvement by introducing randomness, a map with better quality clusters for the HDGSOM*r*. The number of epochs used during the experiments was 10 growing epochs each interleaved by 3 smoothing epochs (total of 40 epochs) in the growing phase, 5 epochs each in smoothing phases 1 and 2 respectively. The neighbourhood sizes for the maps were 2 for the growing phase, 2 and 1 for smoothing phase 1 and 2 respectively. These are much smaller neighbourhood sizes compared to the typical

neighbourhood sizes recommended for the SOM algorithm. The inputs were presented to both versions of the algorithm in the same order they appeared in the original dataset.

The resulting maps are presented in Figures 2-5, which illustrate the maps generated by HDGSOM and HDGSOMr on the 20 newsgroups dataset and the maps generated by the two algorithms on the 19 newsgroups dataset respectively. The newsgroup names indicated on each node are the dominant newsgroups for each node and the number of inputs from that newsgroup mapped to that node is indicated after the name. The clusters are grouped for ease of observation. It should be noted that the results presented are not the optimal results of the algorithms as the training lengths were reduced to evaluate the result of randomness.

The best improvement from the HDGSOM*r* algorithm was on the 20 newsgroups dataset where the HDGSOM algorithm produced only a small number of good clusters. 32 nodes out of the total 65 nodes (49.23%) were not grouped with a neighbouring node. The randomness introduced to the self organizing process in HDGSOM*r* has been able to push the weights much quickly out of the local minima producing crispier clusters. Only 15 out of the 65 nodes (23.08%) were not grouped with a neighbouring node. This is an improvement of over 50% with randomness.

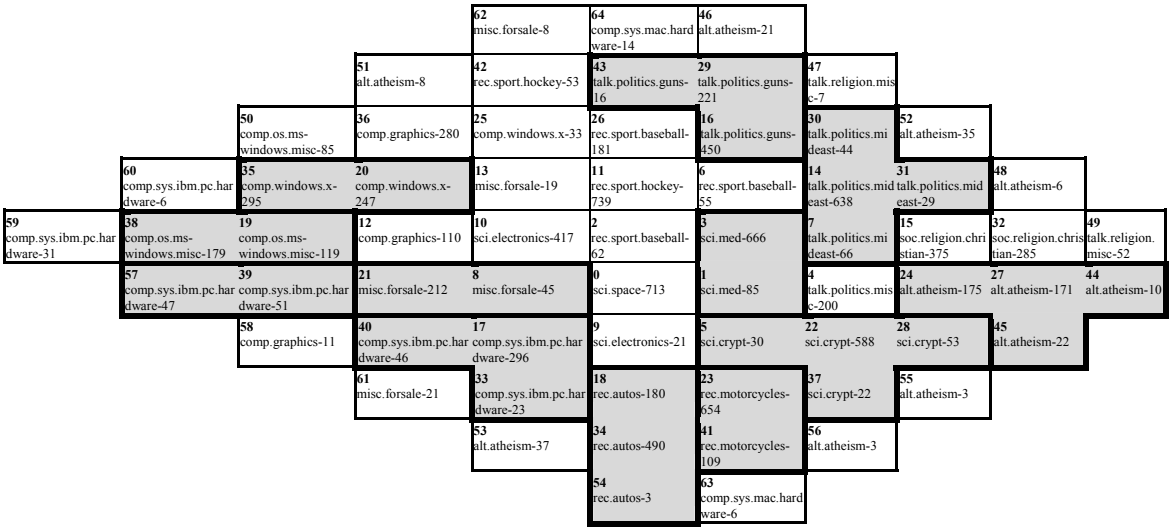

Figure 2. Map from HDGSOM on 20 newsgroups dataset.

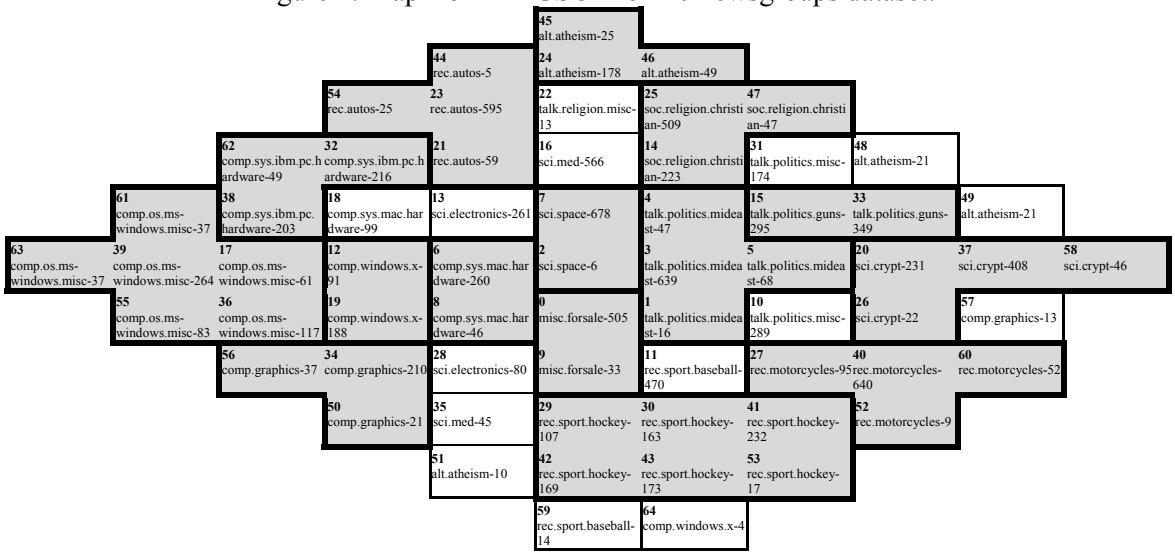

Figure 3. Map from HDGSOM*r* on 20 newsgroups dataset.

The results from the 19 newsgroups dataset did not show much of an improvement as the HDGSOM algorithm was also able to produce good results with the given number of epochs. This could be because the content in this dataset is much more organized than in the 20 newsgroups dataset. The map from the HDGSOM had 17 out of 57 nodes (29.82%) not grouping with neighbouring nodes and the map from HDGSOM*r* had 18 out of 71 nodes (25.35%) not grouping with neighbouring nodes. Even though the results were somewhat similar, closer observations indicated that the clusters produced by HDGSOM*r* were of higher quality than those from the HDGSOM.

Results from several other datasets also indicated that the introduction of randomness into the self organizing process can increase the performance of HDGSOM algorithm to produce better quality clusters within a shorter time period.

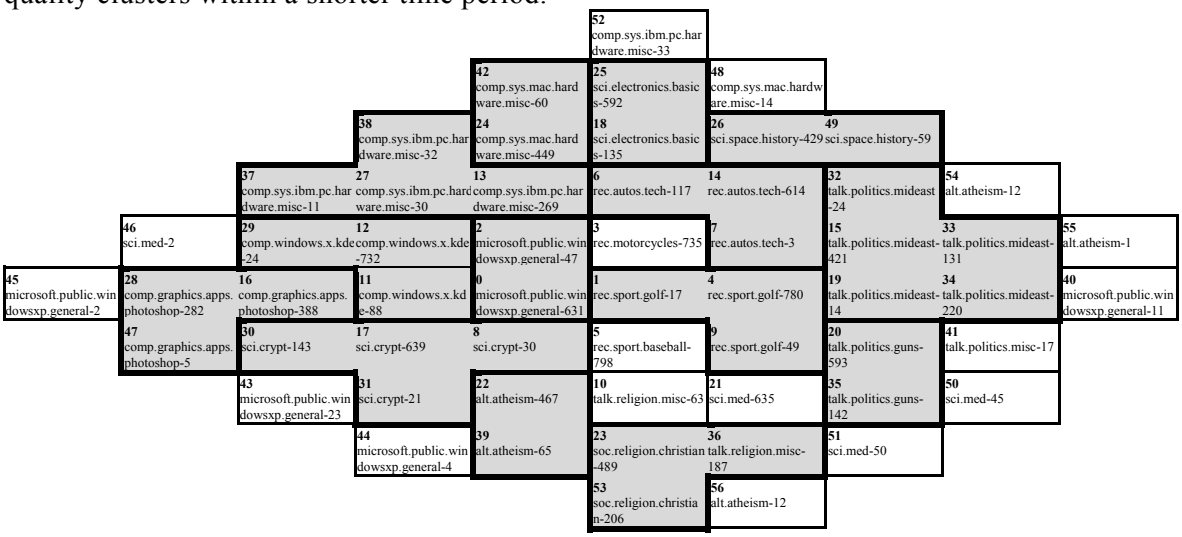

Figure 4. Map from HDGSOM on 19 newsgroups dataset.

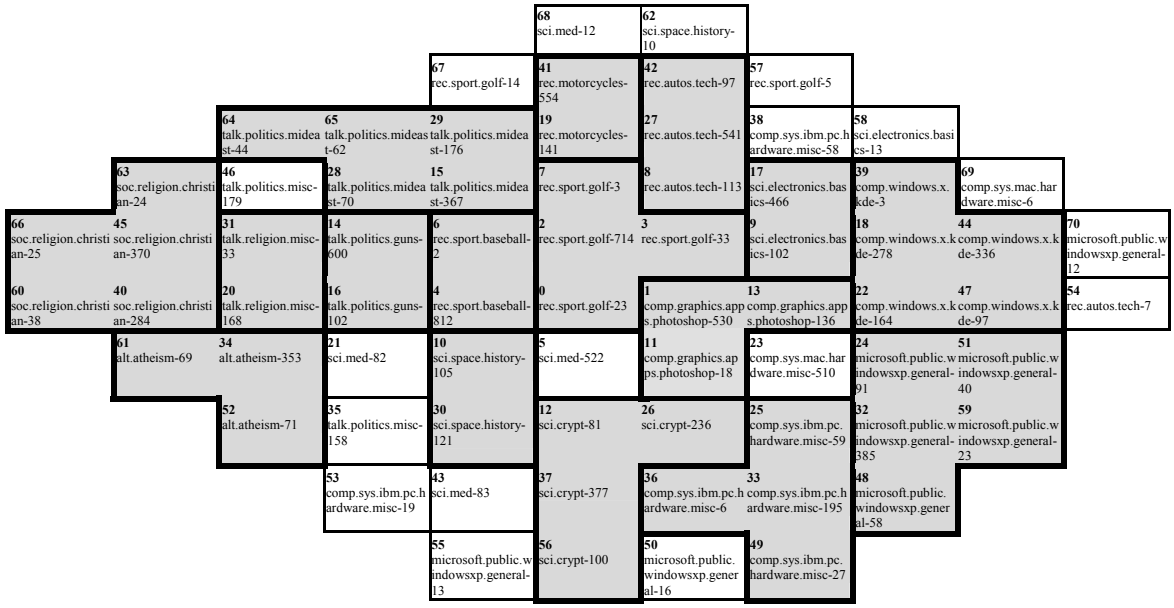

Figure 5. Map from HDGSOM*r* from 19 newsgroups dataset.

5 Conclusions and Future Work

The High Dimensional Growing Self Organizing Map (HDGSOM) [10] is an extension of the Growing Self Organizing Map (GSOM) that was developed to eliminate some of the problems faced by the GSOM algorithm when applied to data with very large dimensions. This paper presented an improvement on the HDGSOM algorithm where the self organizing process of the algorithm was modified by a random value to induce a momentum to the learning. The new algorithm was named HDGSOM*r*.

Experimental results indicate that a simple random momentum to the self organizing process helps the algorithm to move out from local minima and produce better quality clusters within shorter training periods. Since the modification on the HDGSOM*r* algorithm was done to the self organizing process of the algorithm, we believe that this modification can also be applied to other feature map algorithms successfully.

The processing time in the HDGSOM*r* was not significantly larger than the HDGSOM with both algorithms finishing processing minutes apart. The difference could actually be less as the algorithms were implemented using Java which is relatively slower in processing than compiled versions of software and many debugging and informative processing was included in the programs that delayed the processing time. The average time for processing the above dataset was around 6 hours on a Pentium 4 2.8 GHz processor with 1GB of memory.

The combination of randomness and the HDGSOM (HDGSOM*r*) has several significant advantages when processing larger datasets with massive dimensions.

1. The HDGSOM*r* starts with a random set of initial weights that does not need any special processing such as K-Means [21] or genetic algorithms [22] to initialize for faster convergence of the map.
2. It starts with a minimal number of nodes that result in faster processing initially reducing the processing time significantly.
3. The interleaved smoothing phases of the HDGSOM*r* algorithm allow the map to get centred over the centre of gravity of the dataset producing a balanced map.
4. The smaller neighbourhood size used on the algorithm reduces the computational complexity over the conventional SOM algorithm.
5. The randomness in the self organizing process in HDGSOM*r* allows the node weight vectors to move out from local minima and converge quickly.

All of the above improvements become very significant when processing very large datasets with massive dimensions such as those in text and web mining.

We are currently in the process of evaluating the effect of randomness in the original SOM algorithm and testing the HDGSOM*r* algorithm on a massive document collection similar to the document collection in [23]. Results of these experiments will be available in future publications.

References

[1] T. Kohonen (1982), Self Organized formation of Topological Correct Feature Maps. *Biological Cybenetics*.**vol. 43**, p. 59-69.

[2] T. Kohonen (2001), *Self Organizing Maps*. Third ed. 2001, Verlag: Springer.

[3] S. Kaski, T. Honkela, K. Lagus, and T. Kohonen (1998), WEBSOM- Self Organizing maps of document collections. *Neurocomputing*.**vol. 21**, p. 101-117.

[4] A. Rauber, D. Merkl, and M. Dittenbach (2002), The Growing Hierarchical Self-Organizing Map: Exploratory Analysis of High Dimensional Data. *IEEE Transactions on Neural Networks*.**vol. 13**(6), p. 1331-1341.
[5] B. Fritzke (1995), Growing grid-a self-organizing network with constant neighborhood range and adaptation strength. *Neural Processing Letters*.**vol. 2**(5), p. 9-13.
[6] J. Blackmore and R. Miikkulainen (1993). *Incremental grid growing: encoding high-dimensional structure into a two dimensional feature map*. in *IEEE International Conference on Neural Networks (ICNN'93)*. San Francisco, CA, USA: IEEE.p 450-455
[7] C. Hung and S. Wermter (2003). *A Dynamic Adaptive Self-Organising Hybrid Model for Text Clustering*. in *The Third IEEE International Conference on Data Mining (ICDM'03)*.p 75-82
[8] D. Alahakoon, S.K. Halgamuge, and B. Sirinivasan (2000), Dynamic Self Organizing Maps With Controlled Growth for Knowledge Discovery. *IEEE Transactions on Neural Networks, Special Issue on Knowledge Discovery and Data Mining*.**vol. 11**(3), p. 601-614.
[9] S. Kaski (1998). *Dimensionality reduction by random mapping: Fast similarity computation for clustering*. in *IJCNN'98, International Joint Conference on Neural Networks*. IEEE Service Center, Piscataway, NJ.p 413-418
[10] R. Amarasiri, D. Alahakoon, and K. Smith (2004). *HDGSOM: A Modified Growing Self-Organizing Map for High Dimensional Data Clustering*. in *Hybrid Intelligent Systems 2004*: IEEE Computer Society.p 216 - 221
[11] Wikipedia (2005), Randomness, available from http://en.wikipedia.org/wiki/Random Last accessed on 13/04/2005
[12] J. Holland (1973), Genetic algorithms and the optimal allocations of trials. *SIAM Journal of Computing*.**vol. 2**(2), p. 88-105.
[13] S. Chiraphadhanakul, P. Dangprasert, and V. Avatchanakorn (1997). *Genetic Forecasting Algorithm with Financial Applications*. in *1997 IASTED International Conference on Intelligent Information Systems (IIS '97)*. Grand Bahama Island, BAHAMAS.p 174-178
[14] S. Kirkpatrick, C.D.G. Jr, and M.P. Vecchi (1983), Optimization by Simulated Annealing. *Science*.**vol. 220**(4598), p. 671-680.
[15] T. Kwok and S. K.A. (2004), A noisy self-organizing neural network with bifurcation dynamics for combinatorial optimization. *IEEE Transactions on Neural Networks*.**vol. 15**(1).
[16] R. Amarasiri, D. Alahakoon, and K. Smith (2005), HDGSOM: An Enhanced Growing Self Organizing Map for Data with Massive Dimensions. *International Journal on Hybrid Intelligent Systems (IJHIS)*.
[17] K. Lang (1995). *Newsweeder: Learning to filter netnews*. in *Twelfth International Conference on Machine Learning*.p 331-339
[18] M. Fisher (2002), 19 newsgrops dataset, available from http://www.dcs.ex.ac.uk/people/mjfisher/newsgroup.htm Last accessed on 14/04/2005
[19] M. Porter (1980), *An algorithm for suffix stripping*, in *Program*. p. 130-137.
[20] G. Salton and C. Buckley (1988), Term-weighting approaches in automatic text retrieval. *Information Processing & Management*.**vol. 24**(5), p. 513-523.
[21] M.-C. Su, T.-K. Liu, and H.-T. Chang (2002), Improving the Self-Organizing Feature Map Algorithm Using an Efficient Initialization Scheme. *Tamkang Journal of Science and Engineering*.**vol. 5**(1), p. 35-48.
[22] S.-J. Huang and C.-C. Hung (1995). *Genetic algorithms enhanced Kohonen's neural networks*. in *IEEE International Conference on Neural Networks*.p 708-712 vol.2
[23] T. Kohonen, S. Kaski, K. Lagus, J. Salojärvi, J. Honkela, V. Paatero, and A. Saarela (2000), Self Organization of a Massive Document Collection. *IEEE Transactions on Neural Networks, Special Issue on Neural Networks for Data Mining and Knowledge Discovery*.**vol. 11**(3), p. 574-585.