On the Evaluation of Dividing Samples for Training an Extended Depth LSA Machine

ANDREAS ALBRECHT
Computer Science Department
University of Hertfordshire
Hatfield, Herts AL10 9AB
UK

GEORGIOS LAPPAS*
Computer Science Department
University of Hertfordshire
Hatfield, Herts AL10 9AB
UK

Abstract: A classic problem in neural networks is the depth investigation of the network. Is there any potential benefit when training depth-one threshold circuits by adding extra layers and further training them? This question is investigated in a powerful recently introduced artificial intelligence system, called the Logarithmic Simulated Annealing (LSA) machine, that combines the Simulated Annealing Algorithm with a Logarithmic cooling schedule and the classical perceptron algorithm. The first and second layers are trained with the LSA machine learning algorithm. For the learning procedure 50% of the available data are used for training the first layer. The first layer consists of v voting functions of P threshold circuits each one. The next 25% are displayed to the first layer and the outputs of the first layer are producing new samples of length v that are used for training the second layer. The remaining 25% are used for testing the entire network. The main idea is to smooth in the second layer the inaccuracies of the first layer, by training the second layer to evaluate the significance of each output gate of the first layer. Results of the depth investigation reveal that the second layer can produce slightly better results; however the cost of using fewer examples for training the first layer is also considerable.

Key-Words: Simulated Annealing, Optimisation, Perceptron Algorithm, Threshold Circuits, Classification, Machine Learning

1 Introduction

The LSA machine, introduced in [9], is an implementation of a learning algorithm that derives from the combination of the Logarithmic Simulated Annealing algorithm [1], [16], with the classical perceptron algorithm [20], [24]. Simulated Annealing is in our days a popular active research area [25]. The Simulated Annealing method is a feasible method that can successfully handle NP-hard problems, and is the method chosen in LSA machine for the optimization strategy.

The main idea of the LSA Machine is to use a logarithmic cooling schedule to control the unrestricted increase of the classification error on training samples caused by the Perceptron algorithm [4]. The search is guided by logarithmic simulated annealing (LSA), while the neighbourhood is defined by the Perceptron algorithm. The logarithmic cooling schedule applies an inhomogeneous Markov chain, whereas in most applications of Simulated Annealing homogeneous Markov Chains are used as the underlying model. Homogeneous Markov Chains are based on an infinite number of transitions at fixed temperatures, leading on the one hand to optimal solutions, however on the other hand to

*Also:
Department of Public Relations and Communication
TEI of Western Macedonia, Kastoria Campus, GR52100 Kastoria, GREECE

unrealistic real world problems because of the infinite time involved in the algorithm. Therefore the LSA machine, based on inhomogeneous Markov Chains approximates the optimal solutions.

According to Hajek's theorem [13] approximations to optimal solution are guaranteed under certain conditions. However, to verify whether Hajek's conditions are valid for a given configuration space is often very difficult. Nevertheless, in the various modifications of the LSA machine, there are convincing results that this approach approximates the optimal solution even if it unlikely that the configuration space meets the Hajeks convergence conditions. The run-time steps sufficiency to approach the minimum value of the objective function with probability 1-ε is [3]:

$$n^{g} + \log^{O(1)}(1/\varepsilon) \quad (1)$$

Where $\tilde{a}$ depends on the maximum Ã of the escape depth of local minima within the underlying energy landscape. Placement problem was used in this approach, however this result is independent of the problem domain and can be applied to various optimization problems The LSA machine outperforms the classical perceptron algorithm by 15% when the sample set is sufficiently large [3].

Various modifications of the LSA Machine have been applied to classify image data (CT image classification) [4], [6], [7], to gene-expression data analysis [5], [8], and to medical problems [17]. In this work we investigate an extension of the original LSA Machine by adding an extra layer and a new learning method for training the second layer. The application domain is the Winskonsin Breast Cancer Database [17], a popular binary classification domain tackled by many researches [2], [10], [11], [12],[15], [18], [21], [22], [23], [26], [27], [28], [29], [30].

2 Problem Formulation

The core of the LSA machine and the learning methods are presented in the next subsessions

2.1 The core of the LSA machine

Ôhe core of the LSA Machine is based on depth-two threshold circuits. Each layer has a number of depth-two threshold circuits. The input gates calculate hypotheses of the type:

$$f(\vec{x}) = \Sigma_{i=1}^{n} w_i \cdot x_i \geq \vartheta . \quad (2)$$

where n is the number of input attributes of the domain, w_i and ϑ are the input weights and threshold value of the perceptron respectively, calculated by the perceptron algorithm [24] and x_i is the input value of the attribute i. The output gates of the threshold circuit are collected by a voting function that determines the output of the depth-two circuit. The network in the first layer consists of v voting functions each with P threshold circuits.

Simulated Annealing to be explicitly defined [1], requires: a configuration space that defines the search space, an objective function that defines the function to be optimized either by maximizing or by minimizing this function, a transition mechanism that generates our new hypothesis to be examined and defines the acceptance criteria for the new hypotheses and a cooling schedule which controls the annealing procedure.

The configuration space is defined by the set of linear threshold functions $f(\vec{x})$

$$F = \{ f(\vec{x}) : f(\vec{x}) = \sum_{i=1}^{n} w_i . x_i \geq \theta_f \} \quad (3)$$

The objective function is the number of misclassified examples $|S\Delta f|$ from the sample set S calculated by each linear threshold function.

$$S = \{ |\vec{x}, \eta| : \vec{x} = (x_1, x_2, .., x_n), and \eta \in \{-1,1\} \} \quad (4)$$

where {-1,1}indicates a positive or negative example. The objective function then is determined by:

$$S\Delta f = \{ |\vec{x}, \eta| : f(\vec{x}) < \theta_f \text{ and } \eta = + \text{ or } f(\vec{x}) \geq \theta_f \text{ and } \eta = - \} \quad (5)$$

The problem of finding a linear threshold function that minimizes the number of misclassified vectors is a NP-hard problem [14]. Therefore heuristics should be applied in the training procedure. To compute our next hypotheses, the first layer of the circuit is computed by a combination of the Perceptron algorithm and the Logarithmic-cooling schedule with a heuristic of choosing the elements that are far away from being correctly classified. These elements are assigned higher probability for being our next hypotheses. To determine this a deviation function $U(x)$ is constructed in the following way

$$U(x)=\frac{u(\vec{x})}{\sum_{\vec{x}\in S\Delta f}u(\vec{x})} \quad (6)$$

where

$$u(x)=\begin{cases}-f(\vec{x}), & \text{if } f(\vec{x})<\theta_f \text{ and } \eta=1\\ f(\vec{x}), & \text{if } f(\vec{x})\geq\theta_f \text{ and } \eta=-1\\ 0, & \text{otherwise.}\end{cases} \quad (7)$$

Thus preference is given to that $\{x,\eta\}\in S\Delta f$ that maximises the deviation (6).

A new hypothesis is accepted if one of the following happens: a) it produces lower classification error to the objective function or b) it produces higher classification error to the objective function and at each annealing temperature a uniformly randomly selected sample $r\in$ [0,1] is greater than

$$e^{-(o(w_k)-o(w_{k-1}))/t(k)} \quad (8)$$

where $o(w_k), o(w_{k-1})$:the objective function of hypotheses *k* and *k-1*, and *t*(*k*) the annealing temperature of the logarithmic cooling scheme. The logarithmic cooling scheme of the LSA machine is based on Hajek's theorem [13].

$$t(z)=\Gamma/\ln(z+2), z\in\{0,1,...\} \quad (9)$$

Applying this feature to the LSA Machine we manage to use inhomogeneous markov chains of finite length to restrict the classification error. The temperature is lowered at every step implementing the idea to avoid premature convergence to local minima and escape from them with a probability that is lowered due time restricting the acceptance of new sub-optimal steps to take over time.

The training is completed when the classification error is zero, i.e. all samples from the random sampleset are learned or after a predefined number of steps *L*.

2.2 Basic learning methods of the LSA machine

The learning method in the LSA machine requires that each perceptron is trained by a randomly selected training set. The LSA machine introduced a new method to compute the threshold circuits by performing an Epicurean-style learning procedure, where several independent hypotheses are calculated from randomly chosen sub-sets of the total training samples. Each threshold function in each layer is calculated from a random selection of positive S^{pos} and negative S^{neg} samples out of the training set of positive and negative samples T^{pos}, T^{neg} available for that layer.

$$|S^{pos}|=\alpha\cdot|T^{pos}| \quad (10)$$

$$|S^{neg}|=\beta\cdot|T^{neg}| \quad (11)$$

where $\alpha,\beta\in$ [0,1]. The values of α and β, denote the number of random examples that each perceptron will be trained to learn with desirable zero or minimum error.

2.3 Extension of the LSA machine

The entire network is shown in figure 1. The entire available dataset *D* is divided to three datasets (*T1*, *T2*, *T*).

$$T1 \cup T2 \cup T = D \quad \text{and} \quad T1 \cap T2 \cap T = \varnothing \quad (12)$$

The first dataset (*T1*) consists of the 50% of the data and is used for training the first layer of the network. The training method is based on a random selection of *S1* sample sets out of the *T1* for training each threshold circuit. After training the first layer with the 50% of the data the weights are fixed and the layer is exposed to a new dataset *T2* of previously unseen examples, which are the 25% of the data *D*. This produces for every sample of *T2* a vector of length v plus the associated class *c={-1,1}* of the sample

$$New\ sample = [r_1, r_2, \ldots r_v, c] \quad (13)$$

where r_1, r_2, ..r_v the output of each depth-two threshold circuit in the first layer and are formed as:

$$r_i=\sum_{i=1}^{P} fi(\vec{x}) \quad (14)$$

The new samples from (13) are used for training *h* depth-two threshold circuits consisting of *h* voting functions with *m* threshold circuits each. In this way gates at different level are exposed to a learning procedure with different training data. The idea is to correct inaccuracies at the output gates of the first layer by using a different unseen training dataset. This learning procedure will increase the importance of the accurate first layer sub-circuits and decrease the importance of the inaccurate sub-circuits. The training method is the same as in the first layer with randomly sampling *S2* sample sets from the *T2* dataset. After training the perceptrons of the second layer the testing set of unseen

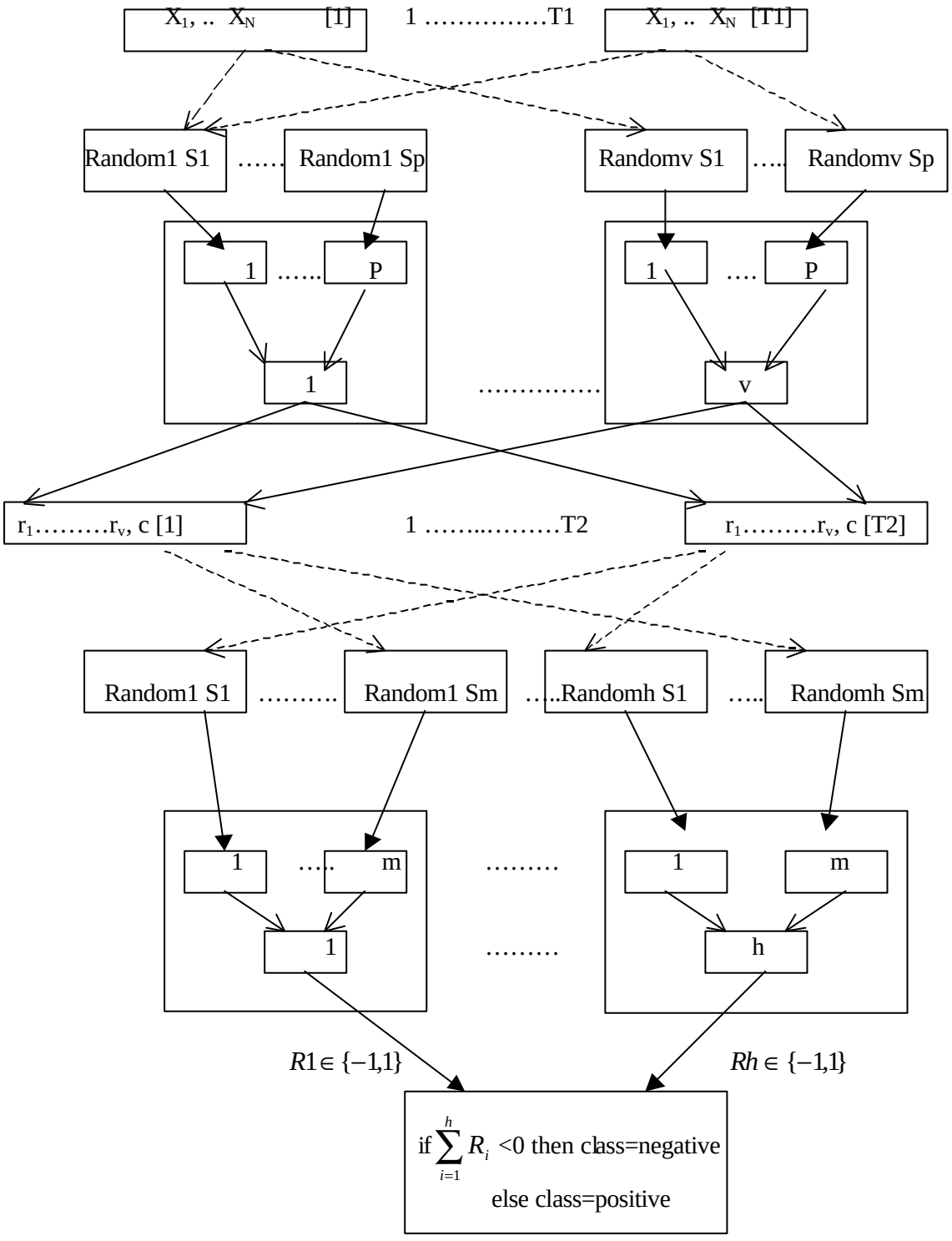

Figure 1: The Entire Network

examples, which is the third dataset consisting of 25% of the data *D*, are applied to the entire network. The outputs at first layer are collected and the class produced by the first layer is compared with the real class of the sample. The number of misclassified examples in first layer $S\Delta f1$ is recorded. Each sample of the testset T after being exposed to the first layer produces a new testing sample of the type (13). The total number of the new samples is exposed to the second layer producing a number of $S\Delta f2$ misclassified examples

After training the second layer the remaining 25% of the data consisting the testset T is used for evaluation of the entire network. In the evaluation phase each sample is exposed to layer one and layer two and at the end of the layer two a function $\sum_{i=1}^{h} R_i$ collects the outputs of the h voting functions and determines whether the sample belongs to a positive or negative class. The misclassification function on the testset is the total error e of the network.

$$e = |S\Delta f|, S \in T \quad (15)$$

3 Results

The application domain is the Winskonsin Breast Cancer Database (17). The Winsconsin Breast Cancer Database (WBCD) can be found at the UCI Repository http://www.ics.uci.edu/~mlearn/MLRepository.html. The WBCD database is the result of the efforts made at the university of Wisconsin Hospital for accurately diagnosing breast masses based solely on a Fine Needle Aspiration (FNA) test. There are 9 input features in the WBCD database. WBCD is a binary classification problem. The output is either a benign case (positive example) or a malignant case (negative examples) The data set consists of 699 samples. 16 samples have missing values, and they are discarded in this work in a pre-processing step. The remaining 683 data are divided to 444 benign (Positive examples) and 239 malignant cases (Negative examples)

The training samples used for training the network and testing the performance are divided to 50% for training the first layer, 25% for producing equally sized new samples for training the second layer and 25% for testing the entire network. WBCD is considered a benchmark database for artificial intelligence systems. Researchers [2], [10], [11], [12], [15], [18], [21], [22], [23], [26], [27], [28], [29], [30], that have tackled the database have provided the literature with results ranging from 90% [11], to 98.24% [26], on the testing data. Classification accuracy by using the LSA machine in [5] and [17] is 98.8%.

In our approach the values of $a=b=0.14$, L=10000 are selected. The value of Γ is kept for each layer to be $(T^{pos}+ T^{neg})/2$, of the randomly selected training samples.

The average total error $\varepsilon(average)$ is calculated after running the program for a considerable number of times.

(P,v,m,h)	(11,5,11,3)	(15,5,15,3)	(20,5,20,3)
$\varepsilon(average)$ Layer 1	3.0	3.0	2.9
Total Errors Layer 1	1.7%	1.7%	1.7%
$\varepsilon(average)$ Layer 2	2.5	2.5	2.4
Total Errors Layer 2	1.5%	1.5%	1.4%

Table 1: Results of classification errors in Layer1 and Layer 2

Comparing the outputs from Layer1 and Layer2 in Table 1 an important conclusion is that the extension to Layer 2 improves the quality of the classification error. Table2 shows the results of the original LSA machine, which consists of only one layer and where the 75% of the data are used for training the depth-two threshold functions.

(P,v)	(11,5)	(15,5)	(20,5)
$\varepsilon(average)$	2.1	2.1	2.0
Total Errors	1.2%	1.2%	1.1%

Table 2: Results by the original LSA machine

The original LSA machine performs better than extending the LSA machine to Layer 2. This leads to the conclusion that for the particular database the quality of results in the first layer depends very much on the number of available examples for training the first layer. The second layer improves the result of the first layer, however the improvement doesn't reach the result gained by using more examples for training the first layer.

Changing the 50%,25%,25% proportion of the sample sets *T1,T2,T* to a proportion of 65%,15%,25% the results shown in Table 3 indicate better classification error in layer 1, which is closer to the origin LSA machine classification error, however the layer 2 with fewer examples for training has a stable behavior of following the results of layer 1

(P,v,m,h)	(11,5,11,3)	(15,5,15,3)	(20,5,20,3)
$\varepsilon(average)$ Layer 1	2.3	2.3	2.2
Total Errors Layer 1	1.3%	1.3%	1.3%
$\varepsilon(average)$ Layer 2	2.3	2.3	2.2
Total Errors Layer 2	1.3%	1.3%	1.3%

Table 3: Results of classification errors in Layer1 and Layer 2 with *T1*=65,*T2*=15,*T*=25

Interesting is that Layer 2 using fewer examples for training doesn't increase the classification error that layer 1 outputs.

The run time for the original LSA machine ranges from 240min to 900min, while the extension of the LSA machine almost doubles this running time.

4 Discussion

We have present an extension of the LSA machine by adding an extra layer and by dividing the training set to train the new layer. The novelty in the training procedure is that the new layer is not trained by the splitted samples but from tuples that consists of vectors, which are the outputs of the first layer when the splitted dataset is exposed to the first layer. This provided us with samples that are related with the quality of output of the first layer. Training of the second layer enables to fix inaccuracies of the first layer and this was displayed by the results. This is an important conclusion of this work.

However, the impact of depth of circuits is under question for this particular dataset as the small improvements that it offers is faded by the better results on layer one where the datasets are not divided for extra layer training. For this particular dataset the number of examples in the first layer is considerable more important than the improvements offered by a second layer training. However, it is very important that the extension to next layer improves the result of the previous layer, providing a reasonable amount of data. Therefore more research on other larger datasets is needed.

References:

[1] E.H.L.Aarts, and J.H.M. Lenstra, *Local Search in Combinatorial Optimization*, Wiley&Sons, 1998.

[2] J. Abonyi, and J.A. Roubos, Structure Identification of Fuzzy Classifiers, *5th online World Conference on Soft Computing in Industrial Applications (WSCS)*, Sept 4-18, 2000.

[3] A. Albrecht, S.K. Cheung, K.S. Leung, C.K. Wong. On the convergence of inhomogeneous Markov Chains approximating equilibrium placements of flexible objects. *Computational Optimization and Applications*, 19:179-208, 2001

[4] A. Albrecht, E. Hein, K. Steinhofel, M. Taupitz, and C.K. Wong. Bounded-Depth Threshold Circuits for Computer-Assisted CT Image Classification. *Artificial Intelligence in Medicine*, 24(2):177-190, 2002.

[5] A. Albrecht, G. Lappas, S.A.Vinterbo, C.K. Wong, and M. Ohno-Machado, Two Applications of the LSA Machine, *Proceedings of the International Conference On Neural Information Processing (ICONIP '02)*, Vol 1, pp184-189, 2002.

[6] A. Albrecht, M. J. Loomes, K. Steinhofel, and M. Taupitz, Adaptive Simulated Annealing for CT Image Classification, *Pattern Recognition and Artificial Intelligence*, 16(5), 2002.

[7] A. Albrecht, K. Steinhofel, M. Taupitz, and C.K.Wong, Logarithmic Simulated Annealing for Computer-Assisted X-ray Diagnosis. *Artificial Intelligence in Medicine*, 22(3):249-260, 2001.

[8] A. Albrecht, S.A.Vinterbo, C.K. Wong, and L.Ohno-Machado, A Simulated Annealing and Resampling Method for Training Perceptrons to Classify Gene-Expression Data. *Proceeding of The International Conference on Artificial Neural Networks (ICANN '02)*, Lecture Notes in Computer Science Series, Springer-Verlag, 2002

[9] A. Albrecht, and C.K. Wong, Combining the Perceptron Algorithm with Logarithmic Simulated Annealing. *Neural Processing Letters*, 14(1):75-83, 2001.

[10] A. Cannon, L.J. Cowen, and C.E. Priebe, Approximate Distance Classification, *Computing Science and Statistics 30*, 1998.

[11] D. Chiang, W. Chen, Y. Wang, and L. Hwang, Rules Generation from the Decision Tree, Journal of Information Science and Engineering, 17:325-339, 2001.

[12] N. Friedman, D. Geiger, and N. Goldszmidt, Bayesian Network Classifiers, *Machine Learning*, Vol 29, 131-163. Kluwer, Boston, 1997

[13] B. Hajek Cooling Schedules for Optimal Annealing, *Mathem.of Operations Research*, 13:311-329, 1988.

[14] K.U. Höffgen, H. U. Simon, K.S. van Horn. Robust Trainability of Single Neurons. *Journal of Computer and System Sciences*, 50:114-125, 1995.

[15] N. Japkowicz, Supervised Learning with Unsupervised Output Seperation, *In Proceedings of the IASTED International Conference on Artificial Intelligence and Soft Computing (ASC2002)*, pp. 338-343, 2002.

[16] S.Kirkpatrick, C.D. Gelat,Jr., and M.P. Vecchi, Optimization by Simulated Annealing. *Science*, 220:671-680, 1983.

[17] G. Lappas, V. Ambrosiadou, Binary and Multicategory Classification accuracy of the LSA Machine, *In Proceedings of the International Conference on Computational Methods in Science and Engineering, (ICCMSE2003)*,pp.340-345, 2003

[18] C.G. Looney, Interactive clustering and merging with a new fuzzy expected value, *Pattern Recognition* 35:2413-2423, Pergamon, 2001.
[19] C.J. Mertz and P.M. Murphy, UCI Repository of Machine Learning Databases. http://www.ics.uci.edu/~mlearn/MLRepository.html, (1996).
[20] M.L.Minsky, and S.A. Papert, *Perceptrons*. MIT Press, Cambridge, Mass., 1969.
[21] M. Madden, Evaluation of the Performance of the Markov Blanket Bayesian Classifier Algorithm, *Technical Report No. NUIG-IT-011002*, Department of Information Technology, National University of Ireland, Galway, 2002.
[22] D. Nauck, and R. Kruse, Obtaining interpretable fuzzy classification rules from medical data, *Artificial Intelligence in Medicine*, vol. 16, pp149-169, 1999.
[23] C.A. Pena-Reyes, and M. Sipper, Fuzzy CoCo: A Cooperative Coevolutionary Approach to Fuzzy Modeling, *IEEE Transactions on Fuzzy Systems*, Vol 9, Number 5, p.p. 727-737, 2001.
[24] F. Rosenblatt. *Principles of Neurodynamics*. Spartan Books, New York, 1962.
[25] P. Salamon, P. Sibani, and R. Frost, *Facts, Conjectures, and Improvements for Simulated Annealing*, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, 2002.
[26] R. Setiono, Generating concise and accurate classification rules for breast cancer diagnosis. *Artificial Intelligence in Medicine*, 18(3), p.p 205-217, 2000.
[27] R. Setiono, and H. Liu, Neural-Network Feature Selector, *IEEE Transactions on Neural Networks*, 8(3): 654-659, 1997.
[28] I. Taha, and J. Ghosh, Characterization of the Wisconsin Breast cancer Database Using a Hybrid Symbolic-Connectionist System,*Tech. Rep. UT-CVIS-TR-97-007, Center for Vision and Image Sciences*, University of Texas, Austin, 1997
[29] W.H. Wolberg, and O.L. Mangasarian. Multisurface Method of Pattern Separation for Medical Diagnosis Applied to Breast Cytology. *Proceedings of the National Academy of Sciences*, U.S.A., Vol. 87, pages 9193-9196, 1990.
[30] J. Zhang, Selecting Typical instances in Instance-Based Learning. *Proceedings of the Ninth International Machine Learning Workshop*, Aberdeen, Scotland. Morgan-Kaufmann, San Mateo, Ca, 470-479, 1992.