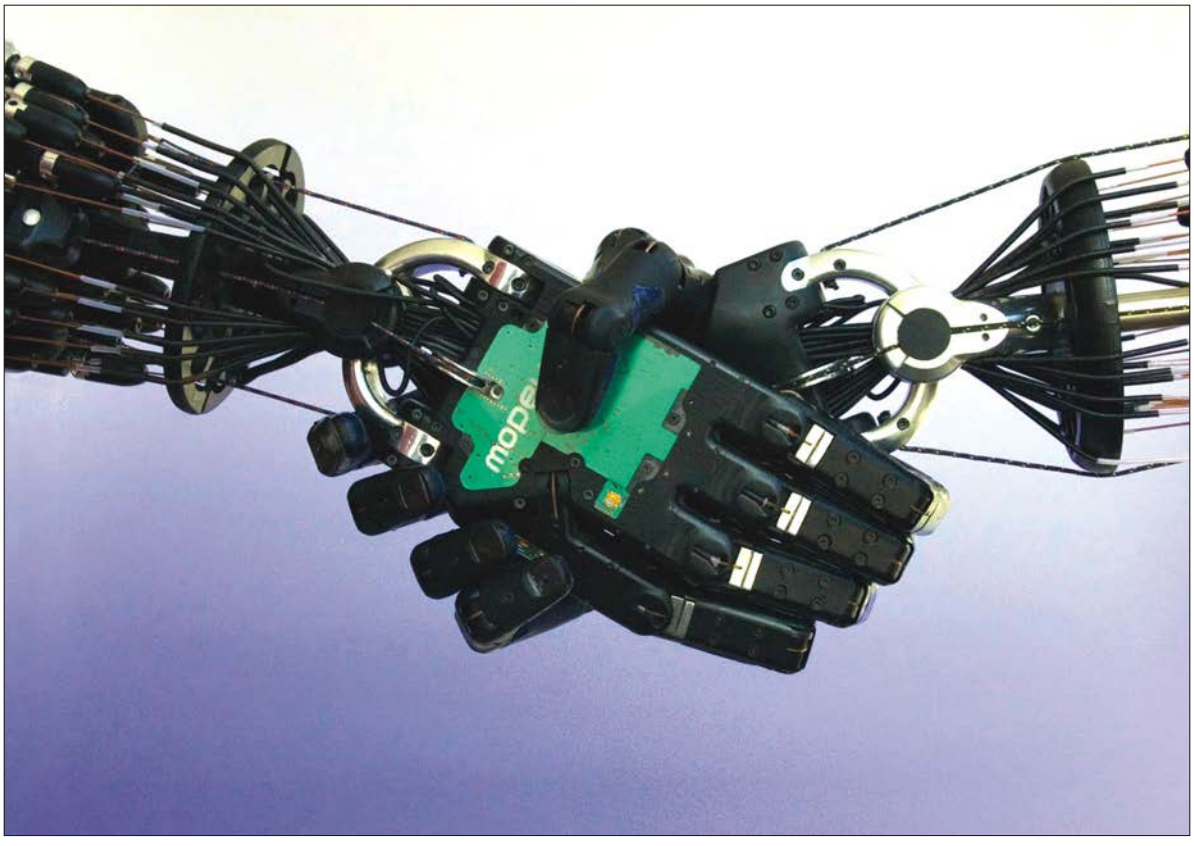

Journal of Artificial Intelligence

ISSN 1994-5450

science alert

ANSInet
an open access publisher
http://ansinet.com

Journal of Artificial Intelligence 2 (2): 56-64, 2009
ISSN 1994-5450

A Comparison Between Three Neural Network Models for Classification Problems

Essam Al-Daoud
Department of Computer Science, Faculty of Science and Information Technology,
Zarka Private University, Jordan

Abstract: This study reports the results from three artificial neural network models. Levenberg-Marquardt (LM), Generalized Regression Neural Networks (GRNN) and Learning Vector Quantization (LVQ) are applied to eight classification problems. Ten-fold cross validation is used to demonstrate the error rate of networks. The experiments show that the generalized regression neural networks outperform the other classifiers, where the average training performance is 0.0436, the testing error rate is 0.137 and the classification rate is 0.80. On the contrary, by using Levenberg-Marquardt, the average training performance is 0.092, the testing error rate is 0.169 and the classification rate is 0.59 and by using learning vector quantization the average training performance is 0.078, the testing error rate is 0.363 and the classification rate is 0.64.

Key words: Levenberg-Marquardt, generalized regression neural networks, learning vector quantization, classification problems

INTRODUCTION

Classification is one of the most active research and application areas of neural networks. The literature is vast and growing. In the classification problem we are given a set of l unlabeled patterns $Pj = \{x_{j1}, \ldots, x_{jl}\}$ where j is the patterns number and we want to assign this set to one of k possible classes from a pre-identified set $C = \{c_1, \ldots, c_k\}$ of classes. We have no control over the size of the set l, it can contain an arbitrary number of patterns and it can vary from problem to problem. Many classification methods are developed and used, such as K-nearest neighbor, brain learning algorithm, bayesian networks, feed forward neural networks, generalized regression neural networks, support vector machine and learning vector quantization (Makal *et al*., 2008; Abidin and Perrizo, 2006; Estevam, 2007). The aim of this study is to answer two questions: what is the best method can be used to solve a classification problem with regard to the training performance and the testing error rate? What is the fastest method can be used with regard to the training time and the response time? To answer these questions, eight datasets are used; it has varied number of the patterns, features and target classes. All the datasets are downloaded from center of machine learning and intelligent systems at the university of California. The used datasets are: Housing, Abalone, Breast cancer, *E. coli*, Image Segmentation, Glass identification, Statlog heart and letter recognition. On the other Hand, three famous and efficient classification methods are chosen: Levenberg-Marquardt algorithm (LM), Generalized Regression Neural Networks (GRNN) and Learning Vector Quantization (LVQ). In theory, each one of these methods has disadvantages; Levenberg-Marquardt has space requirements proportional to the square of the number of weights in the network. This effectively precludes its use in networks of any great size (more than a few hundred weights). The main drawback of GRNNs is that, like kernel methods in general, they suffer seriously from the curse of dimensionality. Generalized regression neural network cannot ignore irrelevant inputs without major modifications to the basic

algorithm. Learning vector quantization suffers the so called prototype under-utilization problem, i.e., only the winner is updated for each input, and because of adopting Euclidean distance measure, LVQ can cause bad performance when the data is non-spherical distribution, and especially contains noises or outliers (Lourakis and Argyros, 2005; Soares *et al*., 2008; Wu and Yang, 2003).

LEVENBERG-MARQUARDT ALGORITHM (LM)

Levenberg-Marquardt is an advanced non-linear optimization algorithm. It is reputably the fastest back propagation algorithm. LM can be thought of as a combination of steepest descent and the Gauss-Newton method. When the current solution is far from the correct one, the algorithm behaves like a steepest descent method: slow, but guaranteed to converge. When the current solution is close to the correct solution, it becomes a Gauss-Newton method. Thus, Levenberg-Marquardt continuously switches its approach and can make very rapid progress (Lourakis and Argyros, 2005). At each iteration of the learning process, the weight vector w will be updated as following:

$$w_{k+1} = w_k + d_k \tag{1}$$

$$d_k = - [J^T J + \mu I]^{-1} J^T \zeta \tag{2}$$

where, d_k is search direction, μ is damping parameter of k-th iteration, ζ is a vector of network errors and J is the Jacobian matrix that contains first derivatives of the network errors with respect to the weights (one hidden layer):

$$J = \begin{bmatrix} \frac{\partial \zeta_1}{\partial w_1} & \frac{\partial \zeta_1}{\partial w_2} & \cdots & \frac{\partial \zeta_1}{\partial w_n} \\ \frac{\partial \zeta_2}{\partial w_1} & \frac{\partial \zeta_2}{\partial w_2} & \cdots & \frac{\partial \zeta_2}{\partial w_n} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{\partial \zeta_k}{\partial w_1} & \frac{\partial \zeta_k}{\partial w_2} & \cdots & \frac{\partial \zeta_k}{\partial w_n} \end{bmatrix} \tag{3}$$

When the scalar μ is zero, this is just Newton's method, using the approximate Hessian matrix. When μ is large, this becomes gradient descent with a small step size. Newton's method is faster and more accurate near an error minimum, so the aim is to shift towards Newton's method as quickly as possible. Thus, μ is decreased after each successful step (reduction in performance function) and is increased only when a tentative step would increase the performance function. In this way, the performance function will always be reduced at each iteration of the algorithm.

GENERALIZED REGRESSION NEURAL NETWORK (GRNN)

Generalized regression neural network (GRNN) and probabilistic (PNN) networks are variants of the Radial Basis Function (RBF) network. Unlike the standard RBF, the weights of theses networks can be calculated analytically. A GRNN can be thought of as a normalized Radial Basis Function (RBF) network in which there is a hidden unit centered at every training case. These RBF units are known as kernels and are usually probability density functions (e.g., Gaussian). The hidden-to-output weights are the target values, so the output is a weighted average of the target values of training cases

close to (near) the given input case (Soares *et al*., 2008; El-Naqa *et al*., 2004). The following formula can be used to find the estimated value of random variable Y in any point of space X:

$$\hat{Y}(X) = \frac{\int_{-\infty}^{\infty} Yf(x,Y)dY}{\int_{-\infty}^{\infty} f(x,Y)dY} \tag{4}$$

f (x,y) can be estimated by observing the values of x and y. Thus, the previous formula can be rewritten as:

$$\hat{Y}(X) = \frac{\sum_{1}^{n} y_i e^{-\frac{d_i^2}{2\sigma^2}}}{\sum_{1}^{n} e^{-\frac{d_i^2}{2\sigma^2}}} \tag{5}$$

$$d_i^2 = (x - x_i)^T (x - x_i) \tag{6}$$

The only weights that need to be learned are the widths of the each kernel, known as smoothing parameters σ. These parameters provide a smooth transition from one observed value to another, even with sparse data in a multidimensional space (Övünç and Yıldırım, 2008).

LEARNING VECTOR QUANTIZATION (LVQ)

The Learned Vector Quantization algorithm (LVQ) was invented by Tuevo Kohonen, who also invented the Self-Organizing Feature Map. It is a hybrid network uses both supervised and unsupervised learning: The first layer is a single-layer competitive network (classical competitive, SOM), the second layer typically linear. The unsupervised Learning Vector Quantization (LVQ) can be seen as a special case of the SOM, where the neighborhood set contains only the winner node. Such learning rule is also called the winner-take all. The standard Kohonen algorithm iteratively adjusts the position of the exemplar vectors stored in the radial layer of the Kohonen network by considering only the positions of the existing vectors and of the training data. For superior classification performance, it is desirable that the exemplar vectors are adjusted, to reposition the Voronoi vectors slightly, so as to improve the quality of the classifier decision regions. It is a two stage process: the first stage is the unsupervised identification of a reasonably small set of features in which the essential information content of the input data is concentrated. The second stage is the classification, where the feature domains are assigned to individual classes (Seo and Obermayer, 2003; Wu and Yang, 2003). The following are the basic steps in this stage:

- For each input x_i find the winning output neuron j by the following criterion:

$$cidx_i = \arg\min_j \|x_i - ctrs_j\|, \quad j=1,2..k \tag{7}$$

- Calculate the new output for x_i. If it is classified correctly, then the winning weight vector j is moved toward the input vector according to the Kohonen ‘s rule:

$$ctrs_j = ctrs_j + \eta [x_i - ctrs_j] \quad (8)$$

- If x_i is classified incorrectly, then the winning weight vector j is moved away from the input vector according to the Kohonen 's rule:

$$ctrs_j = ctrs_j - \eta [x_i - ctrs_j] \quad (9)$$

PREPROCESSING

Three preprocessing strategies can be used: Patterns normalization, redundant features reduction and uncorrelated features elimination. Before training, it is often useful to normalize the inputs and the targets so that they always fall within a specified range as following:

$$xn = 2*(x-minp)/(maxp-minp) - 1 \quad (10)$$

where minp and maxp are the minimum and maximum values in a pattern. The MATLAB function premnmx can be used to normalize the inputs and the targets so that they fall in the range [-1,1]:

```
[pn, minp, maxp, tn, mint, maxt] = premnmx(p, t);
```

In some situations, the dimension of the input vector is large, but the components of the vectors are highly correlated (redundant). It is useful in this situation to reduce the dimension of the input vectors. An effective procedure for performing this operation is principal component analysis PCA. It eliminates those components that contribute the least to the variation in the data set:

```
[ptrans, transMat] = prepca (p, 0.02);
```

This means that prepca eliminates those principal components that contribute less than 2% to the total variation in the data set.

The third strategy is the elimination of the uncorrelated features. In this study the MATLAB correlation coefficients function CORRCOEF(X, Y) is used. The correlation between each feature and the target is calculated, the features that have the lowest correlation can be ignored and the other features are processed. For example, It can be observed in Table 1 that the symmetry feature and fractal dimension feature have the lowest correlation, thus they are ignored and the other 8 features are used.

Table 1: Diagnostic breast cancer features

Feature	Max. value	Min. value	Correlation
Radius	28.1	6.9810	0.7300
Texture	39.3	9.7100	0.4152
Perimeter	188.5	43.7900	0.7426
Area	2501.0	143.5000	0.7090
Smoothness	0.2	0.0526	0.3586
Compactness	0.3	0.0194	0.5965
Concavity	0.4	0.0000	0.6964
Concave points	0.2	0.0000	0.7766
Symmetry	0.3	0.1060	0.3305
Fractal dimension	0.1	0.0500	-0.0128

DATASETS

Eight classification problems are used from the center of machine learning and intelligent systems at the university of California. All the datasets have many features and one discrete target excluding the housing dataset, where the target is continuous. However, the target of the datasets is represented as symbols and needs to be encoded as integer numbers. The basic information about the datasets is summarized in Table 2. The following is a brief description about the datasets:

- **Boston Housing Data (H):** Concerns housing values in suburbs of Boston, the attributes in this dataset includes: proportion of residential land zoned, per capita crime rate by town, proportion of non-retail business and the average number of rooms and weighted distances to five Boston employment centers
- **Breast Cancer (BC):** Features are computed from a digitized image of a fine needle aspirate (FNA) of a breast mass. They describe characteristics of the cell nuclei present in the image. Separating plane described above was obtained using Multisurface Method-Tree (MSM-T). a classification method which uses linear programming to construct a decision tree. Relevant features were selected using an exhaustive search in the space of 1-4 features and 1-3 separating planes
- **Abalone (A):** Predicting the age of abalone from physical measurements. The age of abalone is determined by cutting the shell through the cone, staining it and counting the number of rings through a microscope
- ***E. coli* (E):** is used to predict the protein localization sites in gram-negative bacteria. The target classes such as: cytoplasm, perisplasm and outer membrane. The features including presence of charge on N-terminus of predicted lipoproteins, score of discriminant analysis of the amino acid content of outer membrane and periplasmic proteins, score of the ALOM membrane spanning region prediction program and score of ALOM program after excluding putative cleavable signal regions from the sequence
- **Image Segmentation (IS):** The instances were drawn randomly from a database of 7 outdoor images. The images were hand segmented to create a classification for every pixel. Each instance is a 3×3 region
- **Glass Identification (GI):** The study of classification of types of glass was motivated by criminological investigation. At the scene of the crime, the glass left can be used as evidence, if it is correctly identified. Target classes including window glass, building windows, vehicle windows, containers, tableware and headlamps
- **Statlog Heart (SH):** It is used to diagnose the heart disease. The attributes in this dataset include: the age, sex, chest pain type, resting blood pressure, serum cholestoral in mg/dl, fasting blood sugar, maximum heart rate achieved and the slope of the peak exercise ST segment
- **Letter Recognition (LR):** The objective is to identify each of a large number of black-and-white rectangular pixel displays as one of the 26 capital letters in the English alphabet. The character

Table 2: The basic information about the datasets

Dataset	#Patterns	#Features	#Classes
Housing (H)	506	13	Continuous
Abalone (A)	4177	8	29
Breast Cancer (BC)	569	10	2
E. coli (E)	336	7	8
Image Segmentation (IS)	2310	19	7
Glass Identification (GI)	214	9	7
Statlog Heart (SH)	270	13	2
Letter Recognition (LR)	2500	16	26

images were based on 20 different fonts and each letter within these 20 fonts was randomly distorted to produce a file of 20,000 unique stimuli. Each stimulus was converted into 16 primitive numerical attributes (statistical moments and edge counts) which were then scaled to fit into a range of integer values from 0 through 15.

EXPERIMENTAL RESULTS

MATLAB 7.0 is used to implement the three methods: LM, GRNN and LVQ. The data is normalized, redundant features are reduced and uncorrelated features are ignored. The target in the housing dataset is quantized such that each interval with length 0.1 is considered a class, therefore the number of the classes in the housing dataset becomes 20 classes. Table 3 summarizes the best training parameters and training time for each datasets and method, where LR is the learning rate, M is the number of the neurons in the hidden layer, T is the training time, Ep is the number of the epochs and σ is the width (spread). It can be observed that the best training time and the fewer parameters method is the generalized regression neural network, but the worst training time is the learning vector quantization.

A K-folding scheme with K = 10 is applied (using K-folding means dividing the dataset to k sets and use k-1 of them for training and one for testing. This is repeated k times and then the average of the result is taken). The training procedure for each dataset is repeated 10 times, each time with 90% of the patterns as training and 10% for testing. All the reported results are obtained by averaging the outcomes of the 10 separate tests. Table 4 shows the training performance, the testing error rate and the average response time to the testing sets (R). It is clear the best training performance and the testing error rate can be achieved by using the generalized regression neural network and the best response time can be achieved by using Levenberg-Marquardt method. Figure 1 shows the testing error rate for each dataset and method.

Although, the testing error rate considers a good indicator to the classification rate, but in some cases this correlation is not true. For example in Table 4 the testing error rate of the *E. coli* dataset by

Table 3: The best training parameters and training time for each datasets and method

	LM			GRNN			LVQ		
Dataset	LR	M	Ep	T	σ	T	LR	Ep	T
H	0.010	10	18	0.50	0.6000	0.420	0.20	40	46.7
BC	0.005	4	20	6.50	0.1000	0.420	0.10	50	78.9
A	0.001	5	39	4.70	0.1141	1.530	0.50	8	1153.6
E	0.001	3	15	3.60	0.1170	1.182	0.10	40	110.9
IS	0.001	10	54	21.20	0.1000	0.120	0.07	20	192.3
GI	0.005	5	18	0.56	0.1000	0.120	0.05	80	65.0
SH	0.050	5	14	1.15	1.0000	0.130	0.10	250	234.6
LR	0.001	10	65	36.50	0.0500	1.600	0.10	5	195.1

Table 4: The training performance, the testing error rate and the response time

	LM			GRNN			LVQ		
Dataset	Train	Test	R	Train	Test	R	Train	Test	R
H	0.0071	0.0213	0.020	0.0000	0.0356	0.0500	0.0678	0.052	0.0200
BC	0.1294	0.1402	0.010	0.0056	0.1621	0.0600	0.0742	0.280	0.0200
A	0.0232	0.0223	0.010	0.0212	0.0318	1.1600	0.0603	0.059	0.7010
E	0.1359	0.1980	0.010	0.0116	0.1637	0.0200	0.0578	0.423	0.0200
IS	0.0423	0.0698	0.010	0.0062	0.0504	9.6000	0.0583	0.330	0.1600
GI	0.0512	0.1966	0.010	0.0063	0.1736	0.0200	0.1095	0.386	0.0100
SH	0.2542	0.5563	0.010	0.2984	0.4012	0.0200	0.1358	0.888	0.0100
LR	0.0952	0.1481	0.200	0.0000	0.0787	17.900	0.0625	0.483	0.2200
Average	0.0923	0.1690	0.035	0.0436	0.1371	3.6037	0.0782	0.363	0.1451

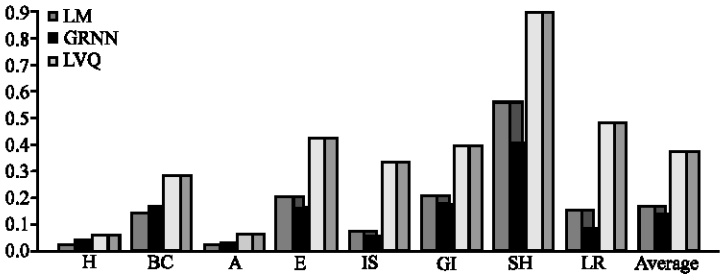

Fig. 1: The testing error rate for each dataset and method

Table 5: The classification rate for each method and dataset

	LM			GRNN			LVQ		
Dataset	Correct	Wrong	Rate	Correct	Wrong	Rate	Correct	Wrong	Rate
H	42.00	9.00	0.821	37.0	14.00	0.725	32.00	19.00	0.627
BC	55.00	2.00	0.9649	53.0	4.00	0.929	53.00	4.00	0.929
A	252.00	236.00	0.602	259.0	229.00	0.619	197.00	221.00	0.471
E	10.00	24.00	0.2941	26.0	8.00	0.764	26.00	8.00	0.764
IS	180.00	51.00	0.7792	216.0	15.00	0.935	187.00	44.00	0.809
GI	8.00	13.00	0.381	16.0	5.00	0.761	13.00	8.00	0.619
SH	23.00	4.00	0.8519	24.0	3.00	0.888	21.00	6.00	0.777
LR	41.00	359.00	0.1025	206.0	44.00	0.824	45.00	205.00	0.180
Average	76.37	87.25	0.59	104.6	40.25	0.800	71.75	64.37	0.640

using LM is 0.198 and by using LVQ is 0.423, but as shown in Table 5, the classification rate by using LM is 0.29 which considers very low with compare to the classification rate by using GRNN or LVQ. Another example, in case of GRNN with GI and SH datasets the testing error rates are 0.1736 and 0.4012 respectively, but their classification rates are higher than other datasets , this results due to the dataset distribution, the statistical properties of the dataset and the number of the target classes.

DISCUSSION

Before discussing the various studies, let us list the pitfalls that await anyone carrying out comparative studies: Firstly, there may be problems due to differences in the way the data were pre-processed, for example by removing or replacing missing values, or using binary or distributed encoding. Secondly, the class definitions may be more suited to some algorithms than others. Thirdly, some comparative studies used variant, but not identical, datasets and algorithms. However, most of the previous studies support the findings of this study. Kayaer and Yildrim (2003) applied MLP (LM), GRNN and RBF to the Pima Indians Diabetes dataset. The best result achieved on the test data was by using GRNN structure and the classification rate was 80.21%, which is very close to one with the highest true classification result that was achieved by using the more complex structured network. Pour *et al.* (2008) used spare parts dataset, the data were gathered from Arak petrochemical company's inventory control software package. They implemented five classifier models, and their results showed that GRNN among other models outperformed MLP. Burcu and Tulay (2006) classified the 26 capital letters in the English alphabet by using Probabilistic Neural Network (PNN) and General Regression Neural Networks (GRNN) Simulation results illustrated that GRNN and PNN are suitable and effective methods for solving classification problems with higher classification accuracy and better generalization performances than their counterparts. Makal *et al.* (2008) study, MLP, RBF and GRNN are used to analyze and detect the splice junctions. The real performances of these networks are found

by applying Receiver Operating Characteristic (ROC) analysis. According to their results, GRNN has the most correct result for both specificity and sensitivity values. However, as expected in this study, a few studies proved that LM outperforms GRNN, for example, Kahraman and Tulay (2009) reported that MLP achieved 91% acceptable results through 120 test data of the time characteristics of digital integrated circuits where GRNN had 77%. On the other hand, when LM is compared with LVQ, the previous studies and this study showed that, LM is superior than LVQ, for example (Baradaran *et al*., 2008) reported that the error rate of applying MLP to the cancer dataset was 14.28%, and the error rate of applying LVQ was 33.33%.

CONCLUSION

This study has presented three classifier models. These models have been applied on eight classification problems. As it has been demonstrated experimentally, the generalized regression neural networks outperform the other classifiers. It has the highest classification rate in 62% (5 out of 8 datasets) of the tested datasets. On the contrary, by using Levenberg-Marquardt, It has the highest classification rate in only 25% (2 out of 8 datasets) of the tested datasets. Then, we can conclude that GRNN is simple and need fewer training parameters, shorter training time and has the best training performance and testing error rate. However, in the real time applications, this study recommends using LM due to the low response time. In future, It would be interesting to test these methods and other classification models on larger set of the datasets (more than eight datasets). We hope to extend our work to answer why and when a particular model should be used.

REFERENCES

Abidin, T. and W. Perrizo, 2006. SMART-TV: A fast and scalable nearest neighbor based classifier for data mining. Proceedings of the 2006 ACM Symposium on Applied Computing, Apr. 23-27, Dijon, France, pp: 536-540.

Baradaran, M.N., S. Shogian and M.H. Zarifi, 2008. Cancer diagnosis using artificial neural networks. Int. J. Comput. Sci. Network Security, 8: 233-236.

Burcu, E. and Y. Tulay, 2006. Statistical Neural Network Based Classifiers for Letter Recognition. In: Intelligent Computing in Signal Processing and Pattern Recognition, Huang, D.S., K. Li and G.W. Irwin (Eds.). Springer-Verlag, Berlin Heidelberg, pp: 1081-1086.

El-Naqa, I., Y. Yang, N. Galatsanos, R. Nishikawa and M. Wernick, 2004. A similarity learning approach to content-based image retrieval: Application to digital mammography. IEEE Trans. Med. Imaging, 23: 1233-1244.

Estevam, R., R. Eduardo and H.F. Ebecken, 2007. Bayesian networks for imputation in classification problems. J Intel. Inform. Syst., 29: 231-252.

Kahraman, N. and Y. Tulay, 2009. Technology independent circuit sizing for standard cell based design using neural networks. Digital Signal Process., 19: 708-714.

Kayaer, K. and T.Y. Yildirim, 2003. Medical diagnosis on Pima Indian diabetes using general regression neural networks. Proceedings of the International Conference on Artificial Neural Networks/International Conference on Neural Information Processing, Jun. 26-29, Istanbul, Turkey, pp: 181-184.

Lourakis, A. and A. Argyros, 2005. Is Levenberg-Marquardt the most efficient optimization algorithm for implementing bundle adjustment. Proceedings of the 10th IEEE International Conference on Computer Vision, Oct. 17-20, IEEE Xplore, London, pp: 1526-1531.

Makal, S., L. Ozyilmaz and S. Palavaroglu, 2008. Neural network based determination of splice junctions by ROC analysis. Proc. World Acad. Sci. Eng. Technol., 33: 630-632.

Övünç, P. and T. Yıldırım, 2008. Genetic optimization of GRNN for pattern recognition without feature extraction. Expert Syst. Appl., 34: 2444-2448.

Pour, A.N., R.B. Tabar and A. Rahimzadeh, 2008. A hybrid neural network and traditional approach for forecasting lumpy demand. Proc. World Acad. Sci. Eng. Technol., 30: 384-389.

Seo, S. and K. Obermayer, 2003. Soft learning vector quantization. Neural Comput., 15: 1589-1604.

Soares, C., L. Montgomery, K. Rouse and J.E. Gilbert, 2008. Automating microarray classification using general regression neural networks. Proceedings of the 7th International Conference on Machine Learning and Applications, Dec. 11-13, IEEE Xplore, London, pp: 508-513.

Wu, K.L. and M.S. Yang, 2003. A fuzzy-soft learning vector quantization. Neurocomputing, 55: 681-697.