International Journal of Latest Trends in Engineering and Technology
Vol.(16)Issue(1), pp.009-016
DOI: http://dx.doi.org/10.21172/1.162.02
e-ISSN:2278-621X

ARTIFICIAL NEURAL NETWORK BASED CLASSIFICATION AND PERFORMANCE EVALUATION USING BENCHMARK DATASETS

Josephina Paul[1]

Abstract: Classification is the fundamental task for many applications including Machine learning and Artificial Intelligence. Therefore, it is a highly demanding and active research area presently, as it has been for the past many decades. Since Artificial Neural Network supports generalization and are robust in solving non linear problems such as multiclass classification, we choose Feed-forward neural network with backpropagation in our experiments. In this study, we classify the benchmark datasets, the Iris and E.coli datasets, which are typical classification problems as they contain multiple classes. By several trials, we arrived at an optimal network architecture 4-10-3 for Iris dataset and 7-15-8 for E.coli dataset. Secondly, the success of neural network mainly depends on its training and hence, we have selected two learning algorithms having faster convergence, 'trainlm'and 'traincgb'from Matlab for training the network. The training, validation and testing of the datasets are done in the ratio 70:15:15. The overall accuracy obtained are 98% and 99.3% for the Iris dataset, and 89.9% and 91.7% for E.coli datasetwith 'trainlm'and 'traincgb' respectively. The accuracies achieved are also compared against those obtained for standard algorithms in the literature and found superior to them.

Keywords: Artificial neural network, Backpropagation algorithm, classification, confusion matrix, accuracy

I. INTRODUCTION

Classification is the fundamental task for various machine learning and pattern recognition models [1], [2],[3]such as biometric identification, voice recognition, computer vision, disease diagnosis and industrial applications, which we come across in our real life. The pursuit for error free and real time systems have made this research area highly challenging and ever attractive for the researchers. Assigning an object to a distinct group among many classes is termed as classification. The classification techniques are broadly divided into two, supervised and unsupervised [4]. In supervised, the output is produced based on a priori knowledge or previous experience. It needs labelled data to train the model and once the model is learnt, it can be used for real time processing. The accuracy of supervised systems are very high as it has already learned from classified and labelled data/outputs, compared to the accuracy of unsupervised systems. The supervised techniques like Bayesian networks, Logistic regression, Decision trees, Support vector machines and Artificial Neural Network (ANN) are into use for classification [4].ANN has been successfully implemented in various applications which require classification of the dataset, for the past few decades [1],[2],[3],[4],[5]. ANN are self adaptive models that support generalization [1],[6],[7]and therefore, it is a powerful tool for the classification and pattern recognition problems. Moreover, trained ANN has the potential to predict the outcomes of similar datasets or similar problems. The Fisher's Iris dataset [8] and E.coli dataset [7],[9] are typical classification problem on which the Feed-forward ANN is appropriate to come up with a perfect, accurate solution, if trained properly.ANN can solve complex problems due to its nonlinearity [1]. The ANN has played a significant role in the present advancement of the branches of Machine learning and Artificial Intelligence.

This paper is segmented into 5. Review of previous research is described in section 2. A detailed account of the architecture of ANN and its various models are also given in section 2. The dataset and the methodology are described in section 3. Section 4 discusses and analyse the results and section 5 details the conclusion and the future works.

---

[1] Assistant Professor, KCAET, Tavanur

II. LITERATURE SURVEY

ANN has been successfully implemented in various applications which require classification of the dataset for the past few decades [1],[3]-[10].The architecture of the ANN used in these studies are diverse in number of layers, nodes and the activation function used. ANN hybridized with multiple linear regression was implemented on two class and multiclass datasets such as Iris datasets and the authors claim better results for their method compared to individual methods such as SVM, LDA etc.[1].Promising results were obtained to studies of protein localization on ecoli and yeast data sets, using Counter propagation Artificial Neural Network (CPANNs), Supervised Kohonen Network (SKNs) and XY-fused networks XYFs [7]. An experiment using ANN with backpropagation on Iris dataset, Madhusmita Swain et al [8] obtained an accuracy of 83.3 to 96.6% while using 50:50 samples for training and testing with 4-3-1 ANN architecture. The accuracy and its consistency of ANN largely depend on the training of the neural net. Therefore, choice of appropriate training algorithms plays a significant role in getting accurate results. Researches in the direction of choosing various training and validation strategies can be seen in many studies.[9]-[10]. Protein localization studies of S.cerevisiaeand E.coli datasets are discussed in [9] in which different feed-forward networks (FNN) trained with k-fold cross validation strategy is used for classifying the datasets. In another study with two-layer feed-forward network layer on breast cancer dataset obtained an accuracy of 97.1[10], where two different activation functions were used – the Sigmoid function on hidden layer and the Softmax function on the output layer respectively.ANN has been found performing well with various error functions as well in classification problems. Multilayer perceptrons with two novelerror functions replacing the MSE function were used in classifying various datasets from the UCI Machine learning repository have shown satisfactory results in [5]. Segmentation of images using ANN for various applications are also seen in the literature. Identifying the car numbers plates by image segmentation and character recognition using FNN[3], distinguish corn plants from the weeds using ANN with backpropagation [11], land cover classification and mapping of remote sensing images [17] are some examples of image classification by using ANNs. The structure of ANN and various training algorithms used in ANN are discussed in the next section.

2.1 Artificial Neural Network

Artificial Neural Network is a computational model that imitates the complex functioning of animal brain [1]. The ANN model consists of interconnected nodes, called neurons that has analogy to biological perceptron, which can take inputs from external sources or from other nodes and, the interconnections of the nodes are associated with some weight. By applying a transfer function on the weighted sum of the inputs, the node produces an output. The simplest model of an ANN is a feed forward neural network in which the information passes in one direction only. An FFN with one input layer and one output layer is called Single layer perceptron and an FFN with additional layers or hidden layers also, are called Multilayer perceptron [4].Fig.1 shows a diagrammatic representation of multilayer perceptron. Multilayer Feed forward networks with a fair number of neurodes are good function approximators [6], when train with enough number of samples. Various layers of ANN can be described as below.

Input layer: The nodes on this layer accept data/inputs from the outside world and passes to the hidden layer without any modification.

Hidden Layer: Receives information from input layer/previous hidden layer and computes weighted sum of the inputs and transfers to the next hidden layer or output layer. Hidden layers are intermediate layers and therefore, devoid of direct connection with the outside world.

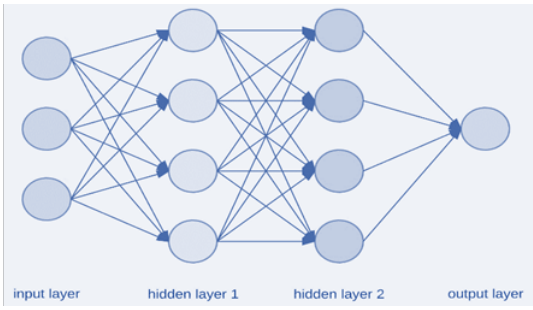

Figure 1Multilayer Perceptron

Output layer: Activation functions are applied on the intermediate outputs of hidden layers and the resultant information is obtained at this layer.ie. theactual output is communicated to the outside world through the nodes of this layer.

The output of neuron j in the hidden layer or output layer is computed as

$$\text{Output } y = f\left(\sum_{i=0}^{n} x_i w_{ij}\right),$$

Where x_0 =1, w_{0j} is bias of the neuron, x_1, x_2 ,...x_n are the input stimuli of nodes and w_{1j}, w_2j ,...w_{nj} are the weights of corresponding input signal, f is the activation function. The activation function converts the input signals to output by applying nonlinear processing and the choice of activation function depends on the problem. The activation functions used, usually, in ANN are Sigmoid, Hyperbolic tangent, Radial basis function and Rectified Linear unit functions. The representation of output of ANN with a transfer function is shown as in Fig. 2.

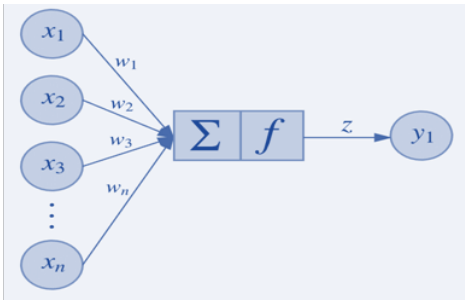

Figure 2Representation of output of a Neural net

Bias: bias is a constant added to a neuron for the flexibility of the model in fitting the data, usually set to 0-1. When all the inputs are zero, bias is set to 1 to ensure a valid output.

The number of layers and the count of neurons in each hidden layer are decided by the complexity of the problem to be solved. When the problem is linearly separable or can be mapped to a separable hyper plane, we use regression or SVM, whereas for non linear problems, ANN act as the appropriate model for the solution. For many real life applications, ANN are found excelling, especially in classification, pattern recognition, speech processing, computer vision problems and in prediction problems. Training of ANN is an important task which actually determines the efficiency of the model. Only a learnt ANN can process the unseen part of the data correctly. Usually, the input data is divided into three sets for training, validating and testing, into 70:15:15 ratio or 60:20:20 or 80:10:10.There are a good number of algorithms available to train the network of which the Back propagation algorithm [4] has gained much interest by the researchers due to its faster convergence and has been implemented in many researches [8],[10],[11].

III. MATERIALS AND METHODS

In Back propagation algorithm, the network weights are modified to achieve the minimum Mean Squared error between targeted and computed value, after every epoch/sample iteration. The weight modification is done backwards from the output layer through the hidden layers to the first hidden layer and so the algorithm is named Back propagation[4]. The algorithm is detailed as below.

3.1 Back propagation algorithm [4]

Input: dataset X, learning rate r
Output: A neural network trained to produce output classes
Method:

(1) Initialize all weights and biases
(2) While not termination condition met{
(3) For each training sample x in X {
(4) For each input layer j {
(5) $O_j = I_j$; //output of input unit is the actual input ie. no modification of input
(6) For each hidden or output layer unit j {
(7) $I_j = \sum_i w_{ij} O_i + \theta_j$;

(8) $O_j = \frac{1}{1+e^{I_j}}$; } // apply transfer function on the weighted sum of input signals
(9) For each unit j in the output layer
(10) $Err_j = O_j(1-O_j)(T_j-O_j)$; //Compute the error for each hidden layers from the last to the first
(11) $Err_j = O_j(1-O_j)\sum_k Err_k w_{jk}$ //compute the error backwards w.r.t the next higher layer, k
(12) For each weight w_{ij} in network {
(13) $\Delta w_{ij} = (l) Err_j O_i$; //increment value of weight
(14) $w_{ij} = w_{ij} + \Delta w_{ij}$; //weight update
(15) For each bias θ_j in network {
(16) $\Delta\theta_j = (l) Err_j$; //incrementing bias
(17) $\theta_j = \theta_j + \Delta\theta_j$; } // update bias
(18) }}

The error of node j in the hidden layer is computed from the sum of errors of all the neurons connected to unit j in the next layer.

$$Err_j = O_j(1-O_j)\sum_k Err_k w_{jk} \quad (1)$$

where w_{jk} is the weight of connection from unit j to node k in the immediate successive layer, and Err_k is the error of unit k. The weights are updated as below

$$\Delta w_{ij} = (l) Err_j O_j \quad (2)$$
$$w_{ij} = w_{ij} + \Delta w_{ij} \quad (3)$$

where l is the learning rate, usually in the range [0,1]. It avoids any chance of getting trapped in a local optimum.Updation of weights and biases can be done in either of two ways, after processing each tuple of data(case updating) or after one complete epoch(epoch updating) [4].

ANNs are good for achieving optimized results and therefore, finding an architecture of the ANN that give optimal solution is important,while implementing.The network is trained with different initial weights and hidden layer weights, varying number of hidden layers and neurons in them, so as to arrive at an optimum output. Here we have done several iterative experiments with different values for them and have come up with an optimised model.

As the efficiency of ANN has a great bias on the training, it is important to choose appropriate training algorithms that can make the ANN adapted to process the unseen samples as well. A number of functions are popular to train the network such asGradient descent algorithm, Scaled conjugate gradient backpropagation and Levenberg-Marquardt backpropagation [13] in practice.Matlab provides all these functions in its library and we have chosen two functions for training our network, the 'traincgb', the Conjugate gradient backpropagation with Powell-Beale restarts function and, 'trainlm', the Levenberg-Marquardt backpropagation function.Inthe conjugate gradient algorithms, the direction of search is reset occasionally to the negative gradient. In the Powell-Beale algorithm, this is done when the difference beween the current and previous gradient is a small epsilon value, using an equality function [13]. In the case of Levenberg-Marquardt Optimization algorithm, the smaller gradient is overcome with the incorporation of diagonal of Hessian [14]. Thus, the trainlm function achieves the solution faster and it is more efficient than the Gradient descent algorithm for medium sized datasets[14].

3.2 Iris dataset

The Iris dataset was collected by a Biologist, named Ronald Fisher, presented in his research paper in 1936, which has been in use by the researchers for the past several years. The dataset was downloaded from the UCI Machine Learning repository for this study. It consists of 150 samples with exactly 50 samples from each flower species or class- Iris setosa, Iris virginica and Iris versicolor. The classification is done based on four features of the flowers: length and width of its sepals and petals measured in centimetres. It is considered as the benchmark dataset for classification and clustering problems [1],[6],[8].Therefore, the challenge is to get most accurate classification of these three classes correctly.

3.3 E.coli dataset

The Escherichia Coli dataset was downloaded from the UCI Machine Learning repository. The protein localization study of E.coli organisms are very significant in the branch of Bioinformatics and therefore, it has been used in many papers [7]-[9],[15],[16]. Moreover, it is a benchmark dataset used by scientists for validating novel classification and clustering algorithms. This dataset consists of 336 samples taken from the amino acid sequences of the E. coli organism. The dataset contains 8 protein patterns, unevenly distributed and each instance contain 7 attributes as described in [15]. Since it is a good benchmark dataset for classification algorithms, we have chosen this dataset for our study.

IV. RESULTS AND DISCUSSION

The experiment was done on Matlab version 2010a on a Core i5 processor, 4GB RAM system. The dataset was split into 70:15:15 for training, validation and test samples. Several trials were done to fix the optimized network topology of the FFN for both the datasets and arrived at a better one. The learning algorithm used is Backpropagation algorithm with different functions Levenberg-Marquardt function ''trainlm'' and Conjugate Gradient with Powell/Beale Restarts function ''traincgb'' available in Matlab pattern recognition toolbox. The accuracy was evaluated by the True positive (TP), True negative(TN), False positive(FP) and False negative(FN) outcomes of each experiment. The accuracy is expressed in percentage of correct classification in the corresponding confusion matrices obtained. The overall accuracies obtained for training, validation and testing are considered for the discussion and the consolidated confusion matrices are given in the next section. The performance curve is also provided in this article for the reference.

4.1. Iris dataset with Feed Forward Network

We have trained the FNN with 2 layers (4-H-3) with varying number of neurons in the hidden layer to get an optimized structure and finally, H was fixed at 10 for all the experiments. The average of 20 trials are calculated and obtained an accuracy of98.0% for 'traincgb' function and 99.3% for 'trainlm' function. The confusion matrix for 'traincgb' and 'trainlm'functions are given in the Table 1 and Table 2 respectively.The first column of the table shows the total number of patterns in each class, in second, the class names and total correct outcomes of each class in the column 3,4 and 5. The correct classifications obtained out of the total classified instances, called precision, are given in the last column. The last row shows the recall, the number of correct outcomes out of total number of instances of each class. The value in the last column given in bold letters represents the F1 score, the overall accuracy in percentage.From the tables,it can be noted that all the 50 samples of Iris setosa were classified correctly (100%), 48 samples (96%) of Iris virginica and 49 samples of Iris versicolor (98%) were classified correctly for the 'traincgb' and thus it records an overall accuracy (F1 score) of 98.0%. From Table 2, it is evident that the overall accuracyobtained for 'trainlm' is a higher value, 99.3%.

Table 1. Confusion matrix for E.coli proteins with FNN and 'traincgb' trainingfuncion

No. of Patterns	class	Iris setosa	Iris virginica	Iris versicolor	Precision
50	Iris setosa	50	0	0	100%
50	Iris virginica	0	48	1	98%
50	Iris versicolor	0	2	49	96.1%
	Recall	100%	96.0%	98%	98.0%

The performance curve for the 'traincgb' function is shown in Fig.3 and that for 'trainlm' function is given in Fig.4. From this, it can be noted that the performance error is a very small epsilon value for the trainlm function compared to that of the traincgb function. Also, note that the convergence is faster for trainlm function, at. 20 epochs against 29 for traincgb function

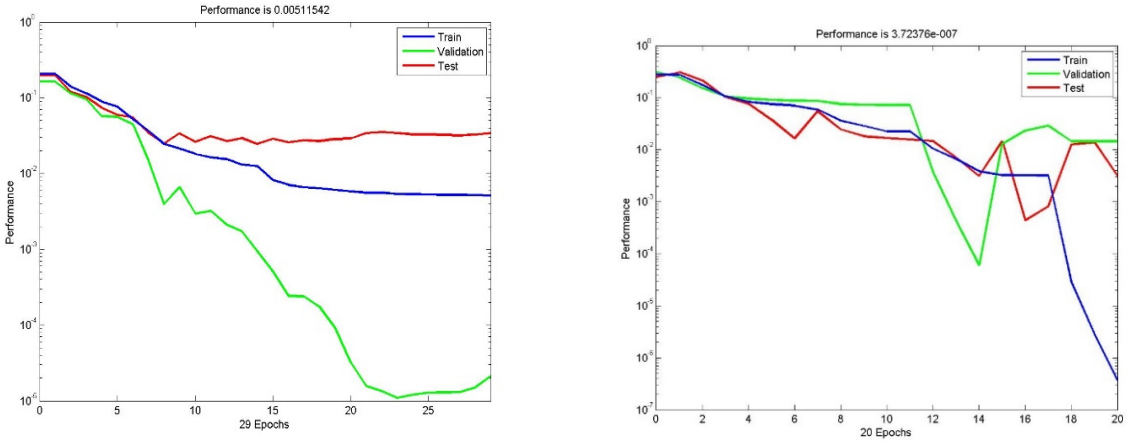

Figure 3 Performance curve of Iris dataset with (a) traincgb function (b) trainlm function

Table 2. Confusion matrix for E.coli proteins with FNN and 'trainlm' trainingfuncion

No. of Patterns	class	Iris setosa	Iris virginica	Iris versicolor	
50	Iris setosa	50	0	0	100%
50	Iris virginica	0	50	1	98%
50	Iris versicolor	0	0	49	100%
		100%	100%	98%	99.3%

Figure 4 Performance curve of Iris dataset with trainlm function

4.2. E.coli dataset with Feed Forward Network

On several trials, the network was fixed with 7-15-8 neurons and applied the training functions 'trainlm'and 'traincgb' as with the Iris dataset. On applying the FNN on full dataset divided randomly in the ratio 70:15:15 for training, validation and testing, the accuracy obtained is given in the Table 3-4.

Table 3. Confusion matrix for E.coli proteins with FNN and 'traincgb' trainingfuncion

No. of Patterns	Class	cp	imL	imS	imU	im	omL	om	pp	
143	cp	140	2	0	0	1	0	0	3	95.80%
77	imL	0	64	1	1	8	1	0	0	85.30%
2	imS	0	0	1	0	0	0	0	0	100%
2	imU	0	0	0	1	0	0	0	0	100%
35	im	0	10	0	0	25	0	0	1	69.40%
20	omL	0	0	0	0	1	18	0	0	94.70%
5	om	0	0	0	0	0	0	5	0	100%
52	pp	3	1	0	0	0	1	0	48	90.60%
		97.90	83.10	50	50	71.4	90	100	92.3	89.90%

While the accuracy of 'traincgb' function is 89.9%, the 'trainlm' function achieved 91.7% accuracy .

Table 4. Confusion matrix for E.coli proteins with FNN and 'trainlm' trainingfuncion

No. of Patterns	Class	cp	imL	imS	imU	im	omL	om	pp	
143	cp	141	2	0	0	0	0	0	4	95.90%
77	imL	0	68	1	0	10	0	0	0	86.10%
2	imS	0	0	1	0	0	0	0	0	100%
2	imU	0	0	0	2	0	0	0	0	100%
35	im	0	7	0	0	25	0	0	0	78.10%
20	omL	0	0	0	0	0	19	0	1	95.00%
5	om	0	0	0	0	0	0	5	0	100%
52	pp	2	0	0	0	0	1	0	47	94.00%
		98.60	88.30	50	100	71.4	95	100	90.4	91.70%

From the performance curve of E.coli dataset in Fig 5(a) and Fig 5(b), the MSE is found to be minimum for trainlm, compared to traincgb function.

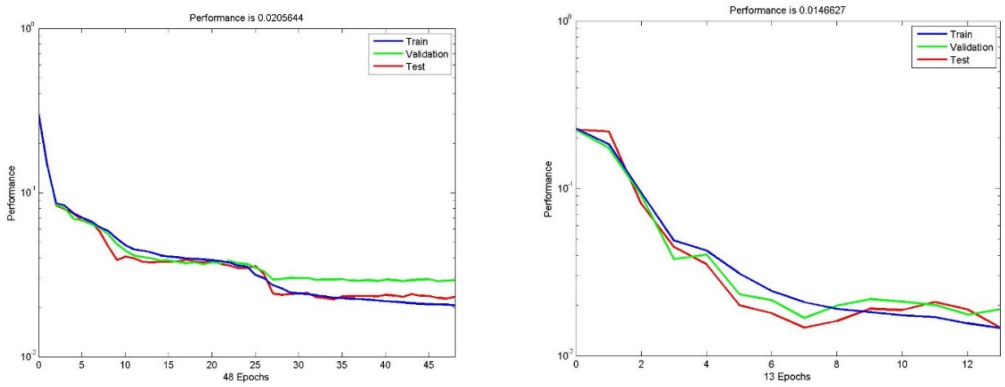

Figure 5 Performance curve of E.coli datasetwith (a) traincgb function (b) trainlm function

4.3Performance evaluation by Comparison with past research

In order to evaluate the efficiency of the algorithm, it needs to compare the accuracy with that obtained for the other proven algorithms while applying on the selected benchmark datasets. We compare our method with accuracies obtained for various standard algorithms published in previous research works for the Iris dataset and E.colidata and are presented in Table 5-6. The values of the proposed methods are given in bold letters. Five algorithms have been referenced which are KNN, SVM, FFN in [1], FFN[8] and BP-PSO[6]. For the Iris, compared to the accuracy of the methods in reference that ranging from 95.4 – 98.7, our method, FFN with trainlm algorithm obtained an accuracy 99.3%, which is obviously superior to them. At the same time, it can be noticed that the accuracy of FFN with traincgb algorithm shows better result, but SVM has even better value.

Table 5 Comparison of proposed mehod with past research on Iris dataset(accuracy in %)

Dataset	KNN [1]	SVM[1]	FFN[1]	FFN[8]	BP-PSO[6]	FFN + traincgb	FFN + trainlm
Iris	97.3	98.7	95.4	96.66	98.00	98.00	99.30

The same trend can be noted with the E.coli dataset as well. From Table 6, it is evident that the accuracies obtained for the proposed methods are much better than the referenced ones, 89.9% and 91.7% for traincgb and trainlm respectively.

Table 6 Comparison of proposed mehod with past research on E.coli dataset(accuracy in %)

Dataset	KNN [1]	FFN[9]	KNN[12]	C4.5[12]	DKNN[12]	FFN + traincgb	FFN + trainlm
E.coli	62.5	88.00	87.5	82.4	88.9	89.90	91.70

V. CONCLUSION AND FUTURE WORKS

Although many researchers have been done for classifying large datasets and images, still it is an attractive area for the researchers, as the classification is very fundamental to various emerging applications such as machine learning, big data analytics and deep learning. ANN is a powerful function approximator and good predictor of unseen part of the datasets [1]. Moreover, studies have proven that they are robust in solving linear as well as non-linear problems [2]. In the study, we have simulated FFNs with two training functions on the chosen benchmark datasets, Iris and E.coli. The overall accuracies obtained for training, validation and testing are analysed. We could achieve superior accuracy for all our simulations. For Iris dataset we got98.0% and 99.3% accuracy with 'traincgb' and 'trainlm' algorithm respectively. Similarly, for the E.coli dataset, we could achieve an accuracy of 89.9% for the 'traincgb' and 91.7% for the 'trainlm' algorithm. The Backpropagation algorithm with a learning rate 0.02 accelerated the convergence of errors in weights. We have compared our proposed models with a few standard algorithms in the past research and found considerable improvement in the accuracies for the proposed method against them.

The dataset affected with noise require several processing of data and need fine tuning of the ANN architecture to obtain desired results. Similarly, processing big datasets are also challenging. We will focus on these aspects in the next research.

REFERENCES

[1] Mehdi Khashei, Ali ZeinalHamadani, Mehdi Bijari, A novel hybrid classification model of artificial neural networks and multiple linear regression models.

[2] Suresh Chandra Satapathy, J.V.R. Murthy, P.V.G.D. Prasad Reddy, B.B. Misra, P.K. Dash, G. Panda, Particle swarm optimized multiple regression linear model for data classification, Applied Soft Computing 9 (2009) 470–476

[3] R.Parisi, E.D.Di Claudio, G.Lucarelli and G.Orlandi,Car plate recognition by neural networks and image processing, *IEEE Trans. on Neural Networks,* vo1.7, no.6, November 1996.

[4] Jiawei Han and Micheline Kamber, Data Mining, Concepts and Techniques, Second Ed., Morgan Kaufmann Publishers

[5] Luís M. Silva, J. Marques de Sá, Luís A. Alexandre, Data classification with multilayer perceptrons using a generalized error function, Neural Networks 21 (2008) pp.1302-1310

[6] Jing-Ru Zhang Jun Zhang, Tat-Ming Lok, Michael R. Lyu, A hybrid particle swarm optimization–back-propagation algorithm for feedforward neural network training, Applied Mathematics and Computation 185 (2007) pp. 1026–1037

[7] Md. ShahriareSatu, Tania Akter and Md. Jamal Uddin, Performance analysis of classifying localizationsites of protein using data mining techniques and artificial neural networks, International Conference on Electrical, Computer and Communication Engineering (ECCE), February 16-18, 2017, Cox's Bazar, Bangladesh.

[8] Madhusmita Swain, Sanjit Kumar Dash, Sweta Dash and AyeskantaMohapatra, An approach for iris plant classification using neural network, International Journal on Soft Computing Vol.3, No.1, February 2012

[9] Aristoklis D. Anastasiadis, George D. Magoulas, Analysing the localisation sites of proteins through neural networks ensembles, Neural Computing & Application (2006), DOI 10.1007/s00521-006-0029-y

[10] Kalpana Kaushik, Anil Arora, Breast Cancer Diagnosis using Artificial Neural Network, International Journal of Latest Trends in Engineering and Technology, ISSN: 2278-621X, 2016

[11] C.-C. Yang, S.O. Prasher, J.A. Landry, H.S. ramaswamy and A. Ditommaso, Application of artificial neural networks in image recognition and classification of crop and weeds, Canadian Agricultural Engineering Vol. 42, No. 3 July/August/September 2000

[12] HafidaBouziane, BelhadriMessabih, and Abdallah Chouarfia, Meta-Learning for Escherichia Coli Bacteria Patterns Classification, Proceedings ICWIT 2012 139, pp 139-150

[13] Matlab documentation version 2010a.

[14] Sam Roweis, Levenberg-MarquardtOptimization

[15] Nakai, K., Kanehisa, M.: A knowledge base for predicting protein localization sites in eukaryotic cells. Genomics. 14, 897-911 (1992).

[16] Horton, P., Nakai, K.: A probabilistic classification system for predicting the cellular localization sites of proteins. In :Proceedings of Intelligent Systems in Molecular Biology, pp 109-115. St. Louis, USA (1996).

[17] Daniel L. Civco, Artificial neural networks for land-cover classification and mapping, Int. Journal of Geographical Information Systems(1993), vol. 7(2), pp 173-186.