МАШИННОЕ ОБУЧЕНИЕ И АНАЛИЗ ДАННЫХ

УДК 004.9, 004.5, 004.41/.42, 004.43
МРНТИ 50.41.25

ACCELERATION OF NEURAL NETWORK TRAINING IN IMAGE RECOGNITION AND CLASSIFICATION PROBLEMS

B. OMAROV[1,2], N. OMAROV[3], A. AKKASOV[4], M. ZHUMAMURATOV[4]

[1]College of Computer Science & Information Technology, Universiti Tenaga Nasional
[2]Kazakhstan Innovations Lab supported by UNICEF
[3]Kazakh University of Railways and Communications
[4]International Information Technology University

Abstract: *The possibility of increasing the efficiency of learning of the neural network that recognizes images is being investigated. Network configuration is made so that all learning examples are recognized. Uses a uniform criterion for the quality of education. Levenberg-Marquardt algorithm has been chosen as an algorithm to teach the neural network, and Bayesian regularization was applied to improve Levenberg-Marquardt algorithm and make it better usable for practical tasks. In the experimental part, we improve quality of the modified LM algorithm using Bayesian regularization and determine appropriate number of hidden layers to prevent overfitting. The considered algorithms allow not only to speed up the learning process, but also to reduce the number of adjustments of the neural network parameters. The latter property is important when parallelizing the learning process on cluster computing systems.*

Keywords: *Classification, Levenberg-Marquardt Algorithm, Neural Networks, Regularization*

БЕЙНЕНІ ТАНУ ЖӘНЕ ЖІКТЕУ МӘСЕЛЕЛЕРІ БОЙЫНША НЕЙРОНДЫҚ ЖЕЛІДЕГІ ОҚЫТУДЫ ЖЕДЕЛДЕТУ

Аңдатпа: *Суреттерді танитын нейрондық желіні үйренудің тиімділігін арттыру мүмкіндігі зерттелуде желі конфигурациясы барлық оқу мысалдарын танитындай етіп жасалады, яғни машинаны үйретуде білім базасы сапасының бірыңғай критерийін басшылыққа алады. Левенберг-Марквардт алгоритмі нейрондық желіні үйретудің алгоритмі ретінде таңдалып алынды және Лейвенберг-Марквардт алгоритмін жетілдіру және тәжірибелік тапсырмалар үшін тиімдірек болуына Байес регуляризациясына сүйенеді. Эксперименттік бөлімде Байес регуляризациясын қолданып, өзгертілген Левенберг-Марквардт алгоритмінің сапасын жақсартамыз және машинаны оқытуда толық емес оқыту мәселесін болдырмау үшін жасырын қабаттардың тиісті санын анықтаймыз. Қарастырылған алгоритмдер оқу процесін жылдамдатуға ғана емес, нейрондық желілік параметрлердің түзетулер санын азайтуға мүмкіндік береді. Соңғы сипатты кластерлік есептеу жүйелерінде оқыту үрдісін параллельдеу кезінде маңызды.*

Түйінді сөздер: *жіктеу, Левенберг-Марквардт алгоритмі, нейрондық желілер*

УСКОРЕНИЕ НЕЙРОННОЙ СЕТИ В ПРОБЛЕМАХ РАСПОЗНАВАНИЯ И КЛАССИФИКАЦИИ ИЗОБРАЖЕНИЯ

Аннотация: *Изучается возможность повышения эффективности обучения нейронной сети, распознающей изображения. Конфигурация сети сделана так, чтобы все обучающие примеры были распознаны. Используется единый критерий качества образования. Алгоритм Левенберга-Марквардта*

был выбран в качестве алгоритма для обучения нейронной сети, а байесовская регуляризация была применена для улучшения алгоритма Левенберга-Марквардта и его лучшего использования для практических задач. В экспериментальной части мы улучшаем качество модифицированного алгоритма LM, используя байесовскую регуляризацию, и определяем соответствующее количество скрытых слоев, чтобы предотвратить переоснащение. Рассмотренные алгоритмы позволяют не только ускорить процесс обучения, но и сократить количество корректировок параметров нейронной сети. Последнее свойство важно при распараллеливании процесса обучения на кластерных вычислительных системах.

Ключевые слова: *классификация, алгоритм Левенберга-Марквардта, нейронные сети, регуляризация*

INTRODUCTION

Artificial neural networks are widely used for solving various tasks. Among the developing applications of ANNs are the processing of analog and digital signals, the synthesis and identification of telecommunication systems [1-2]. Application in telecommunication systems is performed using either the ANN without memory or the ANN with memory. In either case, the key element is the memory-free ANN, the similar role of which is determined by the fact that when using neurons with certain activation functions (transfer characteristics) the ANN is a universal approximator [3]. The latter means that in a given range of variation of input variables, the ANN can reproduce and simulate an arbitrary continuous function of these variables with a given accuracy. Below are discussed issues related to the so-called forward propagation ANN, that is, without feedback. A remarkable property of such ANNs is their stability [4].

After the structure of the INS is selected, its parameters must be set. The choice of ANN structure and neuron types is an independent and very difficult question. As for the parameter values, then, as a rule, they are determined in the process of solving some optimization problem. This procedure in the ANN theory is called learning [5].

There are a huge number of different methods of learning neural networks. One of the classical methods is considered to be the back propagation of error, based on the gradient minimization method. The essence of the method is in the effective calculation of the gradient, which becomes possible due to the rational storage of intermediate variables in differentiating the superposition of functions. However, in its pure form, the back distribution method works poorly. The functional is a multi-extreme excess number of weight coefficients leading to undesirable phenomena of multi-collinearity and retraining. It is not clear what the network structure should be for solving this particular problem: the number of layers, the number of neurons in them, the connections between neurons. To solve these problems, numerous heuristics have been proposed that improve the standard backpropagation method. Each heuristics has many implementation options in various packages its own custom parameters that the expert should set, based on april considerations. Not all heuristics are provided with recommendations for customization. Different heuristics can interact with each other nontrivially, which does not always lead to an improvement in the quality of the network. This raises a new problem: how to choose a set of customizable parameters for all optimization methods? It turns out that the quality of the neural network depends not only on the quality of the source data, but also on the subjective experience of the expert solving the problem.

The aim of this work is to improve the Levenberg-Marquardt method for training neural networks for the problems of classification, regression and forecasting, which automatically automatically determine the structure of the network and evaluate the parameters of all the necessary heuristics.

RELATED WORKS

In this section, we will consider the attempts to improve neural networks as heuristics aimed at improving convergence, optimization of neural network structure, heuristics

aimed at eliminating network paralysis, selection of initial values of network parameters that can be important in neural network training and its improvement [6].

Heuristics aimed at improving convergence

First order gradient methods converge rather slowly. Newtonian second-order methods are also impractical, since they require the calculation of the matrix of second derivatives of the functional Q (!), which is too large. Special tricks are needed to adapt standard optimization techniques for tuning neural networks [7]. One of the methods for increasing the rate of convergence is the diagonal Levenberg – McVardt method proposed in the review [6].

In this method, the step size is calculated individually for each weight coefficient, using only one diagonal element of the matrix of second derivatives:

$$\theta_{jh} = \frac{\theta}{\frac{\partial^2 Q}{\partial w_{jh}^2} + \mu} \qquad (1)$$

Where θ is the learning step values are constant, μ is a parameter preventing the denominator from zeroing and, as a result, unlimited step growth. Value $\frac{\theta}{\mu}$ is the pace of learning on the level of functional areas $Q(w)$

Optimization of Neural Network structure

One of the main parameters of the algorithm that users are asked to choose independently is the number of neurons in the hidden layer. In the worst case, the user needs to select the number of all layers and neurons in each of them. The complexity of the network structure affects the generalizing ability of the algorithm, the rate of convergence and the probability of paralysis [8]. Thus, I would like to get rid of user intervention in the setting of such an important parameter and choose the network structure using algorithmic methods [9].

Sometimes, A large number of degrees of freedom allows you to build very complex mappings, but it gives researchers some problems as:

- Jam at local minima or saddle points.
- The complex landscape of the objective function: the plate alternates with regions of strong nonlinearity. The derivative on the plateau is almost zero, and a sudden break, on the contrary, can send us too far.
- Some parameters are updated much less frequently than others, especially when there are informative, but rare signs in the data, which has a bad effect on the nuances of the generalizing network rule. On the other hand, giving too much importance to all rarely encountered symptoms in general can lead to retraining.
- Too small learning rate makes the algorithm converge for a very long time and get stuck in local minima, too large - to "fly" narrow global minima or to completely diverge

Heuristics aimed at eliminating network paralysis

Network paralysis occurs with an unlimited increase in weights [10]. The activation function argument falls into an area where the activation function has a horizontal asymptote, the derivative of the activation function tends to zero, as a result, the weights are not modified, and the network is stuck in this position.

Weight reduction (weight decay) is one way to prevent paralysis. This method is a special case of the regularization of ill-posed problems. The idea is to limit the growth of the absolute values of the weights. Add to the minimized functional the penalty term:

$$Q_\tau(w) = W(w) + \frac{\tau}{2}\|w\|^2 \qquad (2)$$

A change in the functional leads to the appearance of an additive gradient correction.:

$$\frac{\partial Q_\tau(w)}{\partial w} = \frac{\partial Q(w)}{\partial w} + \tau w \qquad (3)$$

In this case, the rule of updating the scales takes the form as

$$w = w(1-\eta\tau) - \tau\frac{\partial Q_\tau(w)}{\partial w} \quad (4)$$

Advantage of the method is that it is very simply implemented, combined with many functionals and many gradient optimization methods. In addition to preventing network paralysis, this method improves the stability of the balance in the case of multicollarity — when there are linearly dependent or strongly correlated symptoms. The additional control parameter should provide a compromise between the stability of the scales and tuning to a specific sample.

Selection of initial values of network parameters

There are various approaches to select the initial values of the characteristics of the neural network. For example, the Network Advisor of the ST Neural Networks package for a multilayer perceptron by default offers one intermediate layer with the number of elements in it equal to the half sum of the number of inputs and outputs [11].

Usually, assessment results are used based on the knowledge of the problem and the available source data, their dimensions and sample size. So Minsky in his work "Perceptrons" proved that the simplest single-layer neural networks consisting of input and output layers (known as linear perceptron) are capable of solving only linearly separable problems and provide universal linear approximation [12]. This limitation is surmountable by adding hidden layers and using multilayer neural networks. In general, we can say that when solving classification problems in a network with one hidden layer, the input vector is transformed into some new space, which may have a different dimension, and then the hyperplanes corresponding to the neurons of the output layer divide it into classes [13]. Thus, by John Holland from the University of Michigan. They have been applied for solving optimization problems and finding the global extremum of a multi-extremal function. Genetic algorithms work with code sequences (KP) irrespective of their semantic interpretation, therefore KP and its structure is defined by the concept of a genotype, and its interpretation, from the point of view of the problem to be solved, by the concept of phenotype [14]. Each KP represents, in essence, a point in the search space and is called an individual or an individual. The set of KP (individuals) forms the initial set of solutions K (population). A neural network, in particular, a multilayer perceptron, can act as a code sequence. In this case, GAs are applicable for training a neural network, i.e. minimization of the total quadratic error of the network by adjusting its weights [15].

BAYESIAN REGULARIZATION BASED IMPROVEMENT OF LM METHOD

The classical Levenberg-Marquardt algorithm copes poorly with the situation where in the training set contains elements that stands out from the general population. During the working with the practical problems, when data samples out of training dataset, such kind of problems appear [16-18]. The essence of the approach is transition from the searching the minimum point of mean square error to the searching the minimum point of the function expressed by equation (5).

$$F(Y) = \alpha E_\theta + \beta E_D \quad (5)$$

Here E_D – network error, E_θ – the sum of the squares of the network weights, α and β – hyperparameters.

As a result, the algorithm seeks to minimize the network error and to prevent unlimited growth of its weights. The weight of the neural network can be considered as random variables and their distribution density can be expressed by the formula:

$$P(\theta \mid D, \alpha, \beta, M) = \frac{P(D \mid \theta, \beta, M)P(\theta \mid \alpha, M)}{P(D \mid \alpha, \beta, M)} \quad (6)$$

here D – training set, M – neural network model (in our case, it is feed forward neural

network that learnt on the basis of Levenberg-Marquadradt algorithm), θ - weight vector of the neural network. $P(\theta|\alpha,M)$ - priori probability, reflecting our knowledge of the initial weights of the network. $P(D|\theta,\beta,M)$ - likelihood function, which is the probability that the neural network with weights θ correctly respond to a set D [19]. $P(D|\alpha,\beta,M)$ - normalization factor, that provided the equality of the total probability 1. If we assume the training set is noisy with Gaussian noise and the network weights distribution is Gaussian distribution, then formula (6) will be transformed as:

$$P(\theta|D,\alpha,\beta,M)=\frac{\frac{1}{Z_\theta(\alpha)}\frac{1}{Z_D(\beta)}e^{-(\alpha E_\theta+\beta E_D)}}{P(D|\alpha,\beta,M)}=\frac{e^{-F(\theta)}}{Z_F(\alpha,\beta)} \quad (7)$$

Here,

$$Z_D(\beta)=\left(\frac{\pi}{\beta}\right)^{\frac{p}{2}}, \quad Z_\theta(\alpha)=\left(\frac{\pi}{\alpha}\right)^{\frac{S}{2}} \quad (8)$$

Normalization coefficient can be expressed from the formula (9):

$$P(D|\alpha,\beta,M)=\frac{Z_F(\alpha,\beta)}{Z_\theta(\alpha)Z_D(\beta)} \quad (9)$$

Coefficient Z_F remains unknown. However it can be approximated by using the same assumptions as in the Levenberg-Marquardt method as the following equation:

$$Z_F\approx(2\pi)^{\frac{S}{2}}\sqrt{\left|H^{-1}\right|}e^{-F(\theta)} \quad (10)$$

here H^{-1} - inverse matrix to approximate Hessian matrix. Then

$$\alpha=\frac{\gamma}{2E_\theta} \quad \beta=\frac{p-\gamma}{2E_D} \quad (11)$$

Here $\gamma=S-2\alpha\ trace[H]^{-1})$, $\gamma\in[0,S]$ - parameter reflecting the number of weights of a neural network taking part in decreasing a function $trace(H^{-1})$ that is the sum of the diagonal elements of the inverse matrix to approximate Hessian matrix [20].

In this work to calculate hyperparameters α and β we use the formula proposed by Jan Poland in the work [6, 7]:

$$\begin{aligned}\gamma&=S-\alpha\ trace\left([H]^{-1}\right)\\ \alpha&=\frac{\gamma}{2E_W+trace[H]^{-1})}\\ \beta&=\frac{pN}{2E_d}\end{aligned} \quad (12)$$

As a result, neural network is less susceptible to fluctuation in the training set and accurately approximates the function that given by training set.

EXPERIMENT RESULTS

The developed software prototype was used for numerical simulation of the use of a neural network to solve the problem of approximation and classification of input data. To test the approach, three well-known tasks, face recognition and gender classification, lung cancer classification object classification problem has been applied. By classification we mean the procedure for assigning an object (one example of input data) to one of two or more classes. To test the problem the data set was divided into two parts, a training set and a test set. The training set data were 70% of instances of each species, and in the test were about 30% of instances data. Figure 1 demonstrates samples of the each training dataset.

In particular, the problem of choosing the number of hidden layer neurons was considered in [7-13]. In [11] states that the optimal number of neutrons in the hidden layer (N^h) should be calculated from the formula (13).

$$N^h=\sqrt{N^{(i)}N^{(o)}} \quad (13)$$

Figure 1 – Samples of the training dataset

Here $N^{(i)}$ is number of input layer neurons, $N^{(o)}$ is number of output layer neurons.

So, we should check this in practice. In this work, we carried out a practical study of the influence the number of neurons in the hidden layer of the neural network in the learning rate and recognition quality. As selection criteria of number of neurons, we used number of training epochs of neural network and recognition quality.

Figure 2 shows the ratio of neural network training epochs, when the number of neurons in the hidden layer varied from 6 to 24, in increments of 2 neurons. As can be seen from the ratio, number of teaching epochs for the neural networks with different number of neurons in the hidden layer, the increase of this number reduces the speed of learning. As a result, not only number of training epochs increase due to the growth of the Jacobian matrix of the neural network, also total training time increases, too. However, this does not mean that the neural network with 6 neurons in the hidden layer will give the best results.

The graphs in Figure 3 confirm the minimization of the number of neurons in the hidden layer of the neural network does not improve the recognition quality. 9 and 15 neurons in the hidden layer gives the best result. However, the number of training epochs neural network with 18 hidden layer is substantially greater than others. Therefore, we assume that the best results in the "learning rate recognition quality" ratio gives the neural network with 9 neurons in the hidden layer.

CONCLUSION

In the past decades, the world has seen the rapid development of neuro-information technologies. The relevance of research in this direction is confirmed by a large number of different applications of neuro-information systems. It is an automation of pattern recognition processes, adaptive control, approximation of functionals, forecasting, creation of expert systems, and many other applications.

In our research, a mathematical model of pattern recognition using neural networks was proposed. Modification of Levenberg-Marquardt algorithm using Bayesian regularization was described and tested. Using the modified method feed forward neural network was constructed and tested for classification problems. The proposed method performs better in classification task and also maintains a good trade-off between sensitivity and specificity. The proposed method is also computationally cost effective. Therefore, the proposed method can be a useful tool for classification in medical data mining.

Further development of this technique consists in expanding the list of adaptable elements, including in it elements with multidimensional inputs and outputs.

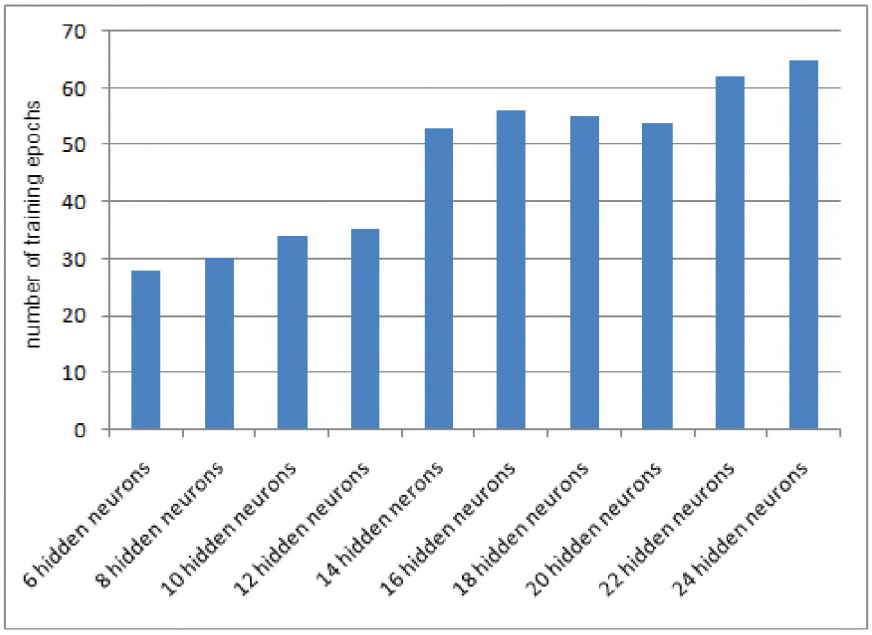

Figure 2 – Variable number of hidden neurons and learning rate

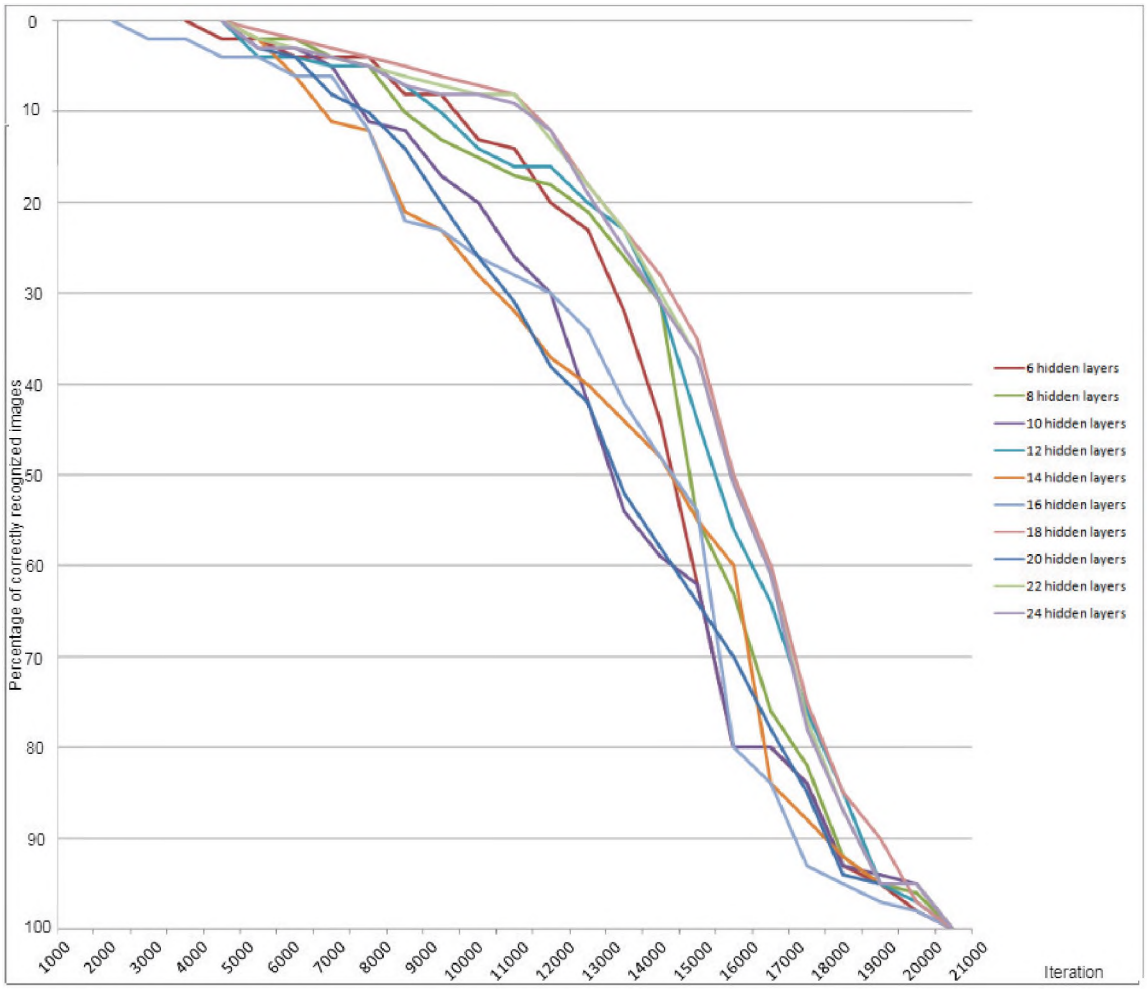

Figure 3 – Testing of neural networks with different number of neurons in the hidden layer

REFERENCES

1. AN Ru, LI Wen Jing, Han Hong Gui. QIAO Jun Fei. An Improved Levenberg-Marquardt Algorithm with Adaptive Learning Rate for RB F Neural Network. Proceedings of the 35th Chinese Control Conference July 27-29, **2016**
2. Henri P. Gavin. The Levenberg-Marquardt method for nonlinear least squares curve-fitting problems. March 22, **2017**.
3. Liyan Qi, Xiantao Xiao and Liwei Zhang. A PARAMETER-SELF-ADJUSTING LEVENBERG-MARQUARDT METHOD FOR SOLVING NONSMOOTH EQUATIONS. Journal of Computational Mathematics Vol.34, No.3, **2016**, 317–338.
4. Soufiane Haddout and Mbarek Rhazi. Levenberg-Marquardt's and Gauss-Newton algorithms for parameter optimisation of the motion of a point mass rolling on a turntable. European Journal Of Computational Mechanics Vol. 24 , Iss. 6, **2015**
5. Murat Kayri. Predictive Abilities of Bayesian Regularization and Levenberg–Marquardt Algorithms in Artificial Neural Networks: A Comparative Empirical Study on Social Data. Mathematical and Computational Applications, May **2016**.
6. Altayeva, A.B., Omarov, B.S., Aitmagambetov, A.Z., Kendzhaeva, B.B., Burkitbayeva, M.A.. Modeling and exploring base station characteristics of LTE mobile networks. Life Science Journal 11(6),31, pp. 227-233. **2014**.
7. Poland J. On the Robustness of Update Strategies for the Bayesian Hyperparameter alpha. Jan Poland, November, **2001**.
8. B. S. Omarov, A. Suliman, K. Kushibar K., Journal of Theoretical and Applied Information Technology 91(2), pp. 238-248, **(2016)**
9. Omarov, B., Suliman, A., Tsoy, A. Parallel backpropagation neural network training for face recognition. Far East Journal of Electronics and Communications. Volume 16, Issue 4, December 2016, Pages 801-808. (2016)
10. A. Altayeva, B. Omarov, H.C. Jeong, Y.I. Cho. Multi-step face recognition for improving face detection and recognition rate. Far East Journal of Electronics and Communications 16(3), pp. 471-491, **(2016)**
11. Satish Saini, Ritu Vijay. 2014. Optimization of Artificial Neural Network Breast Cancer Detection System based on Image Registration Techniques. International Journal of Computer Applications (0975 – 8887) Volume 105 – No. 14, November **2014**.
12. Aimi Abdul Nasir, Mohd Yusoff Mashor, and Rosline Hassan. 2013. Classification of Acute Leukaemia Cells using Multilayer Perceptron and Simplified Fuzzy ARTMAP Neural Networks. The International Arab Journal of Information Technology, Vol. 10, No. 4, July **2013**
13. Devesh Batra, 2014. Comparison Between Levenberg-Marquardt And Scaled Conjugate Gradient Training Algorithms For Image Compression Using MLP . International Journal of Image Processing (IJIP), Volume (8) : Issue (6) : **2014**
14. Muhammad Ibn Ibrahimy, Md. Rezwanul Ahsan, Othman Omran Khalifa. 2013. Design and Optimization of Levenberg-Marquardt based Neural Network Classifier for EMG Signals to Identify Hand Motions. MEASUREMENT SCIENCE REVIEW, Volume 13, No. 3, **2013**.
15. Abdel-Zaher Ahmed, Eldeib Ayman. 2016. Breast cancer classification using deep belief networks. Expert Systems With Applications, 46 (**2016**) 139–144
16. Lv, C., Xing, Y., Zhang, J., Na, X., Li, Y., Liu, T., ... & Wang, F. Y. (2018). Levenberg–Marquardt Backpropagation Training of Multilayer Neural Networks for State Estimation of a Safety-Critical Cyber-Physical System. IEEE Transactions on Industrial Informatics, 14(8), 3436-3446.
17. Flores-Martínez, N. L., Pérez-Pérez, M. C. I., Oliveros-Muñoz, J. M., López-González, M. L., & Jiménez-Islas, H. (2018). Estimation of diffusion coefficients of essential oil of pimenta dioica in

edible films formulated with aloe vera and gelatin, using levenberg-marquardt method. Revista Mexicana de Ingeniería Química, 17(2), 485-506.

18. Choudhury, A., & Greene, D. (2018). Prognosticating autism spectrum disorder using artificial neural network: Levenberg-marquardt algorithm. Avishek Choudhury, Christopher M Greene. Prognosticating Autism Spectrum Disorder Using Artificial Neural Network: Levenberg-Marquardt Algorithm. Archives of Clinical and Biomedical Research, 2(2018), 188-197.
19. Choudhary, S., Doon, R., & Jha, S. K. (2019). Prediction of the Material Removal Rate and Surface Roughness in Electrical Discharge Diamond Grinding Using Best-Suited Artificial Neural Network Training Technique. In Applications of Artificial Intelligence Techniques in Engineering (pp. 487-495). Springer, Singapore.
20. Heravi, A. R., & Hodtani, G. A. (2018). Comparison of the convergence rates of the new Correntropy-based Levenberg–Marquardt (CLM) method and the Fixed-Point Maximum Correntropy (FP-MCC) algorithm. Circuits, Systems, and Signal Processing, 37(7), 2884-2910.