A Novel Approach for Pattern Recognition

Prashanta Ku. Patra
Department of Computer Science & Engineering, College of Engineering & Technology,
Biju Patnaik University of Technology, Bhubaneswar-751003

Swati Vipsita
Department of Computer Science & Engineering, College of Engineering & Technology,
Biju Patnaik University of Technology, Bhubaneswar-751003

Subasish Mohapatra
Institute of Technical Education & Research,
Siksha 'O' Anusandhan University, Bhubaneswar-751020

Sanjit Ku. Dash
Department of Information Technology, College of Engineering & Technology,
Biju Patnaik University of Technology, Bhubaneswar-751003

ABSTRACT
In this paper, Optical Back Propagation and Levenberg Marquardt (LM) algorithms are used for Pattern Recognition. These two algorithms are compared with Classical Back Propagation algorithm with varying Learning rate and Momentum .The simulation results are obtained and are shown in different graphs. The corresponding simulation results show the efficiency of Levenberg Marquardt (LM) algorithms in comparison to Optical Back Propagation Algorithm and Back Propagation Algorithm.

Keywords: *Back propagation, Optical back propagation, Learning rate, Momentum.*

1. INTRODUCTION
Neural Networks are an effective tool in the field of pattern recognition [1]. The neural network classifies the pattern from the training data and recognizes if the testing data holds that pattern. Researches on pattern recognition using neural network have been studied mainly for the convergence time. Since 1960s, the feed forward network such as perceptrons , adaline were developed but they were limited in their training capabilities. The classical Back propagation algorithm is generally used to train the neural network .The basic drawback of this algorithm for its uncertain and long training time. To overcome the drawback of Back propagation (BP) algorithm, an efficient Optical Back propagation algorithm is implemented for improving the training rate. The Optical Back Propagation (OBP) aims to speed up the training process and escape from local minima [2].

Neural Network is an information processing unit that is much inspired by the way the human brain works. Brain can do some computation (such as pattern classification and recognition) faster than conventional computers [3].

An artificial neuron has two modes of operation:

1. In the training mode, neuron is trained to give specific output for particular input patterns.
2. In the using mode, when a taught input pattern is detected at the input node, its associated output becomes the correct output [7].

The optical character recognition, pattern recognition, stock prediction etc. are some of the applications of neural network.

2. PATTERN RECOGNITION AND NEURAL NETWORKS
Pattern recognition is concerned with the classification or description by computer of objects, events or other meaningful regularities in noisy or computer environments. Pattern recognition is the study of concepts, algorithms and implementations that provide artificial system with a perceptual capability to put patterns into categories in a simple and reliable way [5].

Motivations for the study of pattern recognition:

1. It is a part of the broader field of Artificial Intelligence, which is concerned with techniques that enable computers to do things that seem intelligent when done by people.
2. It is an important aspect of applying computers to solve problems that involve analysis and classification of measurements taken from physical process.

Thus, pattern recognition is a popular application that enables the full set of human perception to be acquired by machine.

Neural network possesses the capability of pattern recognition. Researchers have reported various neural network models capable of pattern recognition, models that have the function of self organization and can learn to recognize patterns [13]. In the training stage (Approximation), neural networks extract the features of the input data. In the recognizing stage (Generalization), the network distinguishes the pattern of the input data by the features, and the result of information is greatly influenced by the hidden layers [6]. Neural-network learning can be specified as a function approximation problem where the goal is to learn an unknown function (or a good approximation of it) from a set of input-output pairs. Every instance in any dataset used by machine learning algorithms is represented using the same set of features. The features may be continuous, real coded, categorical or binary. If instances are given with known labels (the corresponding correct outputs) then the learning is called supervised, in contrast to unsupervised learning, where instances are unlabeled. Basically, neural networks are built from simple

units, sometimes called neurons or cells by analogy with the real thing.

Layered neural networks involve a set of input cells connected to a collection of output cells by means of one or more layers of modifiable intermediate connections. The most natural use of this architecture is to implement associativity, by associating an input pattern, the representation of a concept or situation, with another item, either of the same kind or totally different. So neural networks with multiple layers are described as Associative Networks.

3. PROPOSED WORK

The error back propagation algorithm has been a significant milestone in neural network research area of interest, but the algorithm has a very poor convergence rate [9] [14]. The convergence can be defined as the number of epochs the neural network undergoes to learn a particular pattern until the MSE falls to some predefined value. To improve the convergence rate of classical BP, a new efficient algorithm OBP is implemented to train the neural network for pattern recognition. Among second order approaches, namely Newton's method, conjugate gradient's or Levenberg Marquardt optimization techniques, the LM algorithm is widely accepted.

The paper thus presents a comparative study and intends to find the efficient algorithm for training neural networks by performing simulations.

4. DESCRIPTION OF ALGORITHMS

4.1 BACKPROPAGATION ALGORITHM

Back-propagation training algorithm when applied to a feed-forward multi-layer neural network is known as Back-propagation neural network. Functional signals flows in forward direction and error signals propagate in backward direction. That's why it is Error Back-propagation or shortly backpropagation network. The activation function that can be differentiated (such as sigmoidal activation function) is chosen for hidden and output layer computational neurons. The algorithm is based on error-correction rule [7] [14].

The algorithm defines:

1. Initialization of weights (w) and biases (b) to random small values and target (t) is fixed.
2. Forward computation: Output of each layer is y= Φ (w*x+b)

Where w=synaptic weight, x=input and b=bias value. Output of input layer is the input of hidden layer. In this way actual output is calculated.

3. Error is calculated by the difference of target and the actual output at output layer of neuron. Error, e=t-y
4. Backward computation: Error at each layer is calculated by partial differentiation.

For output layer error, eo= 0.5 * [dΦ (yhidden)/dyhidden] *e and for hidden layer error, eh= [dΦ(Yinput)/dYinput] *wout *eo .

5. Weights and biases in each layer are updated according to the computed errors.

Updated weight, Wnew =Wold- lr *elayer*xlayer

Updated bias, bnew=bold- lr*elayer where elayer is the error of the particular layer and Xlayer is the input that is fed to the layer and Ir is the learning rate.

6. Step 2 to 5 is repeated until the acceptable minimized error.

4.2 OPTICAL BACKPROPAGATION ALGORITHM

OBP resolves the shortcoming by slightly modifying the error signal function of BP algorithm, thereby greatly accelerating the convergence rate of the training process. The convergence speed of the training process can be improved significantly by OBP through maximizing the error signal, which will be transmitted backward from the output layer to each output unit in the intermediate layer [2].

In BP, the error at a single output unit is defined as:

$$\delta opk = (Ypk - Opk) \bullet f'o'k(netopk) \qquad (4.2.1)$$

Where the subscript "P" refers to the pth training vector, and "K" refers to the kth output unit. In this case, Ypk is the desired output value, and Opk is the actual output from kth unit, then δopk will propagate backward to update the output-layer weights and the hidden-layer weights.

While the error at a single output unit in adjusted OBP will be as:

$$New\delta^{o}_{pk} = (1 + e^{(Y_{pk} - O_{pk})^2}) \bullet f^{o'}{}_{k}(net^{o}_{pk}) \quad , \; if\ (Y - O) >= zero.$$

$$New\delta^{o}_{pk} = -(1 + e^{(Y_{pk} - O_{pk})^2}) \bullet f^{o'}{}_{k}(net^{o}_{pk}) \quad , \; if\ (Y - O) < zero. \qquad (4.2.2)$$

An OBP uses two forms of Newδ°pk, because the exp function always returns zero or positive values, while adapts operation for many output units need to decrease the actual outputs rather than increasing it. The Newδ°pk will propagate backward to update the output-layer weights and the hidden-layer weights. This Newδ°pk will minimize the errors of each output unit more quickly than the old δopk, and the weights on certain units change very large from their starting values.

The steps of an OBP:

Step 1. Apply the input example to the input units.
Step 2. Calculate the net-input values to the hidden layer units.
Step 3. Calculate the outputs from the hidden layer.
Step 4. Calculate the net-input values to the output layer units.
Step 5. Calculate the outputs from the output units.
Step 6. Calculate the error term for the output units, using the Newδ°pk (using equations (4.2.2) instead of δ°pk.
Step 7. Calculate the error term for the hidden units, through applying Newδ°pk, also.
Step 8. Update weights on the output layer.

$$W_{okj}(t+1) = W_{okj}(t) + (\eta \bullet New\delta_{opk} \bullet i\ pj) \qquad (4.2.3)$$

Step 9. Update weights on the hidden layer.

$$Whji(t+1) = Whji(t) + (\eta \bullet New\delta\ h\ pj \bullet X\ i) \qquad (4.2.4)$$

Step 10. Repeat steps from step 1 to step 9 until the error (Ypk – Opk) is acceptably small for each training vector pairs.

Like the classical BP the OBP is stopped when the squares of the differences between the actual and target values

summed over the output units and all patterns are acceptably small.

4.3 LEVENBERG MARQUARDT ALGORITHM

LM algorithm is one of the second order methods which are proposed so far in order to improve convergence properties of BP algorithm. Among second order approaches, the LM algorithm is widely accepted as the most efficient one in the sense of realization accuracy. The steps involved in training a neural network using LM algorithm are as follows:

$$\vec{e} = \sum_{k=1}^{P} \frac{1}{2}(t_k - y_k)^2 , \tag{4.3.1}$$

Step 1: Present all inputs to the network and compute the corresponding network outputs and errors. Compute the mean square error over all inputs as in Eq. (4.3.1).
Step 2: Compute the Jacobian matrix, J (z) where z represents the weights and biases of the network.
Step 3: Solve the Levenberg-Marquardt weight update equation to obtain Δz.
Step 4: Recompute the error using z +Δz. If this new error is smaller than that computed in step 1, then reduce the training parameter μ by μ–, let z = z +Δz, and go back the step 1. If the error is not reduced, then increase μ by μ+ and go back step 3. The μ– and μ+ are defined by user.
Step 5: The algorithm is assumed to have converged when the norm of the gradient is less than some predetermined value, or when the error has been reduced to some error goal.
The weight update vector Δz is calculated as

$$\Delta z = \left[J^T(z)J(z) + \mu I\right]^{-1} J^T(z)E , \tag{4.3.2}$$

where E is a vector of size P calculated as

$$E = \left[t_1 - y_1\ t_2 - y_2 \ldots t_p - y_p\right]^T . \tag{4.3.3}$$

Here JT (z) J(z) is referred as the Hessian matrix.
I is the identity matrix,
μ is the learning parameter.
For μ = 0 the algorithm becomes Gauss-Newton method. For very large μ the LM algorithm becomes steepest decent or the error backpropagation algorithm. The parameter is automatically adjusted at each iteration in order to secure convergence. The LM algorithm requires computation of the Jacobian J (z) matrix at each iteration step and the inversion of JT (z) J (z) square matrix [10].

5. EXPERIMENT

In the training process, pattern mode training is followed. The images fed to the input layer are ipexmri.png and ipexrice.png. The size of the image is a 64x64 matrix. The image is then converted to grey level matrix and pixel value ranges between 0 – 255. To normalize the matrix each pixel value is divided by 255.Hence the value of matrix element ranged between 0 – 1. (Normalized value to be fed to the input layer). In the training process, a sliding window of 3x3 matrixes is chosen .The window is moved over the entire matrix to cover all the normalized pixel values. The center pixel of 3x3 matrixes is fixed as the target output.

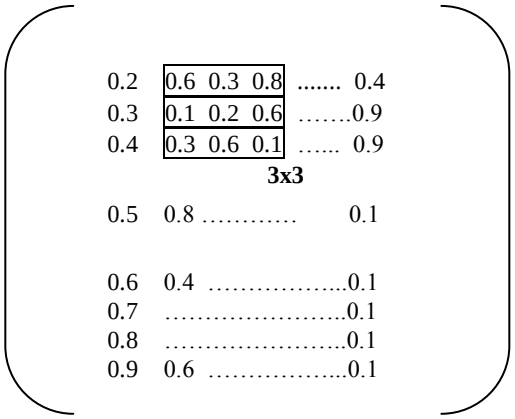


Figure 1: Normalized matrix with a sliding window of size (3 x 3)

All the matrix element of the sliding window (3x3) is fed to the input layer. So the fixed architecture of the neural network is 9 x 4 x 1.
The training process is continued till all the elements of the normalized matrix are presented to the network. The training process is stopped as the neural network has been trained until MSE falls to 0.0001.

MSE is defined as:

$$MSE = \frac{1}{2P}\sum_{p=1}^{P}\sum_{j=1}^{N_M}\left(T_j - O_j\right)^2$$

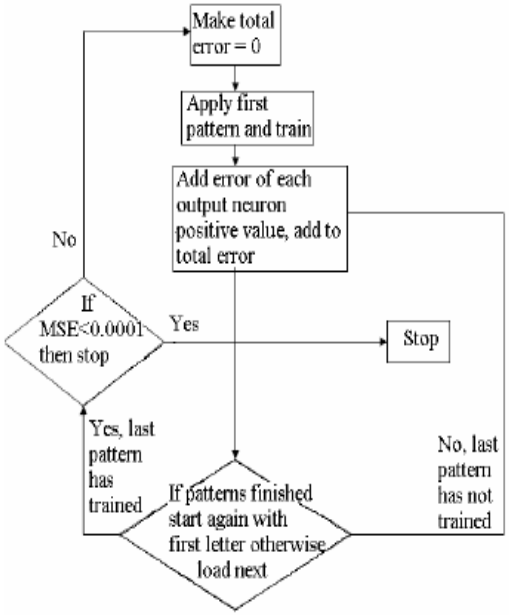


Figure 2: Flow chart of training proces

Inputs are of three types:
1. pattern= [0.2 0.4 0.7 1 0
0.7 0.4 0.2 0 1]
2. XOR GATE

3. Images

6. SIMULATION RESULTS AND GRAPHS

The simulation process is carried on the computer having Pentium-IV processor with speed 2.6 GHz and 512 MB of RAM. The MATLAB version used is 7.5.

A. Input: XOR Gate

Table 1. No. of epochs when α = 0.7 to achieve a MSE less than or equal to 0.0001.

Learning rate	OBP	BP
0.1	19	5995
0.2	17	3413
0.3	16	3071
0.4	14	2792

The LM algorithm took 9 epochs to train XOR gate.

B. Input: pattern

Table 2. No. of epochs when η = 0.4 to achieve a MSE less than or equal to 0.0001.

Momentum	OBP	BP
0.4	33	6706
0.6	26	5363
0.8	22	4468
0.9	20	4124

The LM algorithm took 14 epochs to train pattern.

C. Inputs are images.

Table 3.The table summarizes the no. of epochs and MSE (less than or equal to 0.001). The time duration to reach the convergence is also taken into account. The architecture is fixed at 9 x 4 x 1.

Image	α	η	MSE	BP(time duration)	BP(epochs)
Ipexmri.png	0.2	0.4	0.001	8 hrs.	7652
Iexrice.png	0.5	0.3	0.001	2.5 hrs.	2794

Image	α	η	MSE	OBP(time duration)	OBP(epochs)
Ipexmri.png	0.2	0.4	0.001	1.5 hrs.	1200
Iexrice.png	0.5	0.3	0.001	25 mins.	438

The LM algorithm took 100 and 120 epochs respectively to train images.

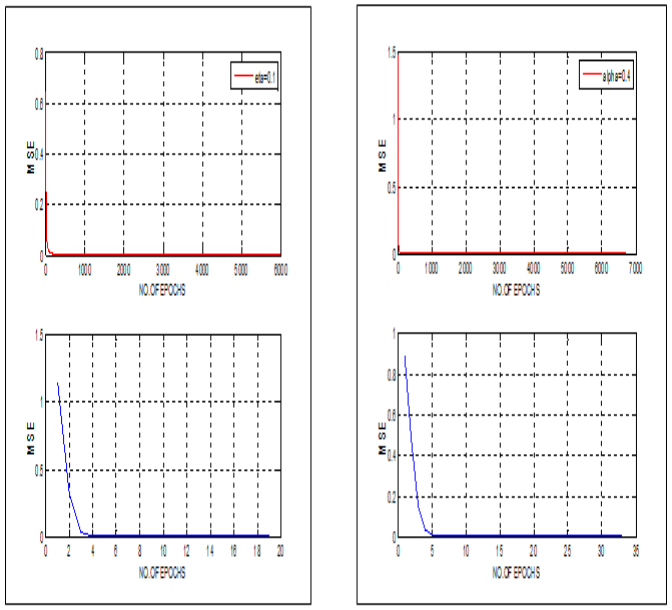


Figure 3: Plot showing No. of Epochs vs. MSE in BP and OBP

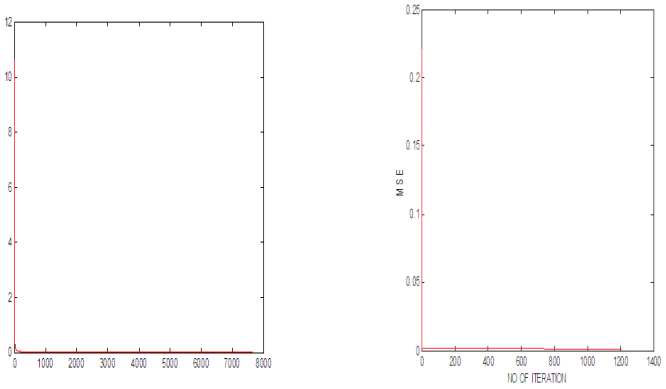


Figure 4: Plot showing No. of Epochs vs. MSE (inputs are images) in BP and OBP

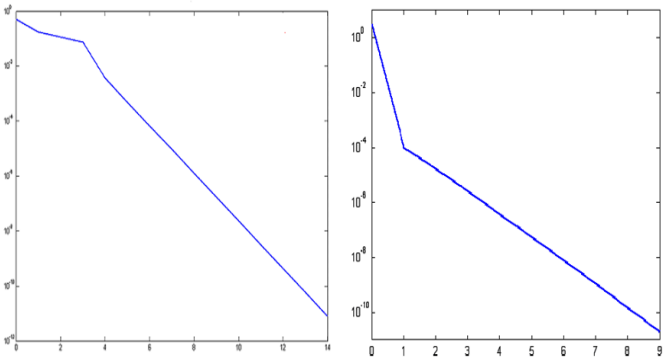
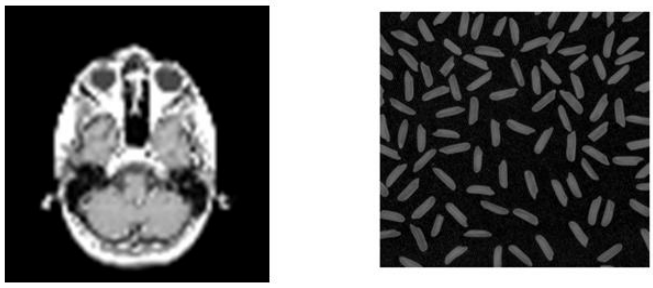


Figure 5: Plot showing output of LM algorithm



Ipexmri.png Ipexrice.png

Figure 6: Images used for training

7. CONCLUSION AND SCOPE OF FUTURE WORK

From the graphs it is concluded that the Levenberg Marquardt algorithm have the fastest convergence rate in comparison to optical backpropagation and classical back propagation algorithm.

Future work will study the effect of using the momentum on the performance of LM algorithm.

8. REFERENCES

[1] Bishop,C.M. 1995. Neural Networks for Pattern Recognition, Oxford.

[2] Mohammed A. Otair, Walid A. Salameh , Speeding Up Back-Propagation Neural *Networks*, Proceedings of the 2005 Informing Science and IT Education Joint Conference, pp. 167-173.

[3] Fredric M.Ham, Ivica kostanic, Principles of Neurocomputing for Science and Engineering (TATA McGraw-Hill 2002 chapter 1 -pg 3, 19, 26.

[4] Ralston,A. , Reilly,E.D, 1993 . Encyclopedia of Computer Science, Van Nostrand Reinhold. 3rd Ed.

[5] Liu,Y. and H.Ma, Ding, 1991, Pattern recognition using ω-orbit finite automata, K. H. Tzao, Editor, Proc. SPIE 1606, Boston, MA, Nov. 1991, pp.226-240.

[6] Satish Kumar, Neural Networks –A Classroom Approach TATA McGraw-Hill- ISBN 0-07-048292-6, pg-61,166,188.

[7] Simon Haykin , 2009. Neural Networks, A comprehensive Foundation, 2nd ed., Pearson Prentice Hall.

[8] Michalopoulos,D.,Hu,C,K.,2002, An error backpropagation artificial neural networks application in automatic car license plate recognition, Lecture Notes in Computer Science, 2002, volume 2358/2002.

[9] Werbos, P. J. 1988. Backpropagation : Past and future, Proceeding of International Conference on Neural Networks, San Diego ,CA, 343-354.

[10] Mishra,D., Yadav,A. Ray,S., and Kalra,P,K. , Levenberg-Marquardt Learning Algorithm for Integrate-and-Fire NeuronModel , Neural Information Processing - Letters and Reviews Vol.9, No.2, November 2005.

[11] Lera , G. and Pinzolas. M., Neighborhood Based Levenberg–Marquardt Algorithm for Neural Network Training, IEEE Transactions on Neural Networks, Vol. 13, No. 5, September 2002.

[12] Rajasekaran,S., Vijayalakshmi Pai G,A. Neural Network, Fuzzy Logic and Genetic Algorithm , Prentice Hall of India, pg-13-20.

[13] Pawlicki,T. , Lee,D,S.,Hull,J. and Srihari,S. ,Neural Networks and their application to handwritten digit recognition, Proc. IEEE International Conference on Neural Networks: Vol. II, San Diego,CA, 1988, pp.63-70.

[14] Rumelhart, D. E., Hinton,G. E. and Williams, J,, Learning representations by back- propagating errors", Nature,vol.323,pp.533-536,1986.

[15] Otair, M. A. & Salameh, W. A. (2004), An improved back-propagation neural networks using a modified non-linear function, Proceedings of the IASTED International Conference, 2004, pp. 442-447.