Communications in Mathematics and Applications
Vol. 13, No. 1, pp. 351–365, 2022
ISSN 0975-8607 (online); 0976-5905 (print)
Published by RGN Publications
DOI: 10.26713/cma.v13i1.1680

http://www.rgnpublications.com

Review Article

Effect of Hidden Neuron Size on Different Training Algorithm in Neural Network

Arvind Kumar* and Sartaj Singh Sodhi

University School of Information, Communication & Technology, Guru Gobind Singh Indraprastha University, Sector 16C, Dwarka, New Delhi 110078, India
***Corresponding author:** arvind.usict.134164@ipu.ac.in

Received: June 23, 2021 **Accepted:** August 3, 2021

Abstract. We use different types of training algorithms in the neural network. But, we cannot say which kind of training algorithm is fast for a given problem. So, in this survey paper, we are trying to find which types of training algorithm are better for categorization problems. For this purpose, we used ten types of training algorithms in the pattern network in MATLAB. We used Levenberg-Marquardt (LM), Bayesian regularization backpropagation (BR), BFGS Quasi-Newton (BFG), Resilient Backpropagation (RP), Scaled Conjugate gradient backpropagation (SCG), Conjugate Gradient with Powell/Beale Restarts (CGB), Fletcher-Powell Conjugate Gradient (CGF), Polak-Ribiere Conjugate Gradient (GDM), One Step Secant (OSS), and Variable Learning Rate Backpropagation (GD) algorithm. In this survey paper, we also check, affects of these algorithms on neural network when we applied different types of hidden neuron size. During this survey we found some new facts. We found that RP, SCG, CGB, CGF and OSS are fastest algorithms. BFG takes more time with respect to hidden neuron size. GDM and GD take more epochs. BR algorithm is not acceptable for image categorization.

Keywords. Neural Network, Training algorithm, Pattern network, Levenberg-Marquardt (LM), Bayesian Regularization backpropagation (BR)

Mathematics Subject Classification (2020). 68T01, 68T07, 68T10, 68T45

1. Introduction

Now-a-days, neural network is used many areas. Clustering, classification, image recognition are some areas where we used neural network. In all these areas, we used training algorithm

for training the data. In the neural network first of all we take initial weights and bias of the network. Then we train the network and find the actual (or new) weight and bias of the network with the help of eqs. (1.1)-(1.3):

$$w_{\text{new}} = w_{old} + \Delta w_{ji}(n), \tag{1.1}$$

$$b_{\text{new}} = b_{old} + \eta(\text{error}), \tag{1.2}$$

where

$$\text{weight correction} \quad (\Delta w_{ji}(n)) = \eta \cdot \text{error} \cdot y_i(n). \tag{1.3}$$

Here, η is learning rate parameter, and $y_i(n)$ is the input of the ith layer of the network with number of hidden neuron size is n. In this survey paper, we take ten famous training algorithms which are currently used in neural network.

1.1 Levenberg-Marquardt (LM)

D.W. Marquardt ([36], [26]) proposed Levenberg-Marquardt (LM). If x is the input, then the LM algorithm is eq. (1.4).

$$x_{k+1} = x_k - [J^T J + \mu I]^{-1} J^T e. \tag{1.4}$$

Here J is the Jacobian matrix that contains first derivates of the network errors with respect to the weights and biases, e is a vector of network errors, J^T transpose of Jacobian matrix, μ is step size, and k is the number of iterations.
When $\mu = 0$, then eq. (1.4) becomes eq. (1.5):

$$x_{k+1} = x_k - [J^T J]^{-1} J^T e. \tag{1.5}$$

Eq. (1.5) is just like a Newton's method. When μ is very large, then eq. (1.4) becomes gradient descent with a small step size.

Many authors used LM in neural network. For example, with the help of LM algorithm in the neural network, Liang and Ning [24] showed that the parameter optimization results obtained with this method can be very close to the actual effect under different conditions and in some cases have a better control effect. Ju *et al*. [19] proposed a projected Levenberg-Marquardt (PLM). They showed that PLM method has better convergence dealing with non-smooth constraints even poor initial value setting. Särkkä and Svensson [34] showed that LM and line-search extensions of iterated extended Kalman smoother (IEKS) algorithms converge better than the classical IEKS.

1.2 Bayesian Regularization backpropagation (BR)

Whenever derivative of weight, input and transfer functions presents in the network, then BR method can train the neural network.

Suppose, X is the input values, jX is Jacobian matrix, E is the error, I is identity matrix and μ is the mean, then with the help of eqs. (1.6) to (1.8), we calculate new weight and bias variables of X. This process is also known as backpropagation process.

$$JJ = jX * jX, \tag{1.6}$$

$$Je = JX * E, \tag{1.7}$$

$$dX = -(JJ + I * \mu). \tag{1.8}$$

Here, dX is the search direction. This value is put up into the eq. (1.9) for adjusting the weight and bias values of X.

Many authors used BR in neural network. For example, for solving inverse kinematics problems, Handayani *et al*. [16] used Bayesian regularization backpropagation in neural network and reached the desired coordinate. Payal *et al*. [28] showed that Bayesian regularization backpropagation (BR) algorithm is more accurate as compared to LM algorithm. BR algorithm reduces the need for lengthy cross-validation. Baigen *et al*. [4] accurately recognize fault on board equipment with the help of BR in Backpropagation Neural Network. Dalipi and Yayilgan [8] predicted skiing injuries with the help of BR and LM algorithm and showed that BR has better performance by achieving higher predictive accuracy. However, in this paper, *we show that BR algorithm is not acceptable for image categorization*.

1.3 Broyden-Fletcher-Goldfarb-Shanno Quasi-Newton (BFG)

Dennis and Schnabel [9] described Broyden-Fletcher-Goldfarb-Shanno (BFGS). This algorithm takes more time, more storage and less iteration. Suppose, X is the input values. Then, weight and bias variables of X are adjusted according with the eq. (1.9).

$$X = X + a * dX, \tag{1.9}$$

where dX is the search direction and a is the parameters which is used to minimize the performance along the search direction. In succeeding iterations, the search direction is computed according to the formula (1.10).

$$dX = -H \setminus gX, \tag{1.10}$$

where gX is the gradient and H is an approximate Hessian matrix.

Many authors used BFG in neural network. For example, Chang *et al.* [6] proposed an accelerated linear convergence rate under mild conditions with the help of stochastic limited memory Broyden-Fletcher-Goldfarb-Shanno (sL-BFGS) algorithm. Lv *et al.* [25] developed a new spectral scaling memory less BFGS algorithm for solving large scale unconstrained optimization problems. Zhao *et al.* [45] improved convergence rates and complexities of the stochastic L-BFGS. In this paper, *we show that BFG algorithm takes more time for image categorization.*

1.4 Resilient Backpropagation (RP)

This algorithm is also known as Rprop. Rprop is proposed by Riedmilier and Braun [32]. Weight and bias variables of X is adjusted according to the eq. (1.11).

$$dX = \Delta X * \text{sign}(gX), \tag{1.11}$$

where ΔX is the change of inputs. The elements of ΔX are all initialized with zero value, and gX is the gradient of the each iterations. This value is put up into the eq. (1.9) for adjusting the weight and bias values of X.

Many authors used RP in neural network. For example, Cui *et al.* [7] proposed a novel maximum power point tracking (MPPT) method which is based on Rprop neural network. Leholo *et al.* [22] showed that the train_LM gives better performance than train_SCG (Scaled

Conjugate Gradient), train_BFGS (Quasi-Newton) and train_RP. Kliment *et al.* [21] showed that the Rprop algorithm can be consider for the most robust because of its iteration process is based on the sign of the gradient, not its magnitude.

1.5 Scaled Conjugate gradient (SCG) backpropagation

The scaled conjugate gradient algorithm is based on conjugate directions [27]. Whenever derivative of weight, input and transfer functions presents in the network, then this method can train this network.

Many authors used SCG in neural network. For example, Zubir *et al.* [46] achieved 100% accuracy for successful classify agarwood oil quality using SCG algorithm in Multi-Layer Perceptron (MLP). Upadhyay *et al.* [38] achieved 96.67% and 93.23% accuracy in prediction of Service Level Agreement (SLA) in cloud computing using SCG algorithm. Wang *et al.* [39] achieved 96.5% recognition accuracy of Gas Insulated Switchgear (GIS). Upadhyay and Nagpal [37] found 97.12% classification accuracy of stressed Electroencephalogram (EEG) Spectrum.

1.6 Conjugate Gradient with Powell/Beale Restarts (CGB)

This algorithm was proposed by Powell [30]. In Powell-Beale Restarts (PBR) approach, orthogonality between the current and the previous gradient vectors is checked at the each iteration [12]. Conjugate Gradient with Powell/Beale restarts if there is very little orthogonality left between the current gradient and the previous gradient [2].

In the Conjugate Gradient Iteration process, first of all we take initial inputs $(x_0) = 0$, initial residual (r_0) =bias value and search direction $(d_0) = 0$, for next iteration (k), we calculate step size (a) with the help of eq. (1.12), residual (r_k) with the help of eq. (1.13) and improvement (p_k) with the help of eq. (1.14).

$$(a) = \frac{r_{k-1}^T r_{k-1}}{d_{k-1}^T A d_{k-1}}, \tag{1.12}$$

where A is input value

$$(r_k) = (r_{k-1}) - (a_k A d_{k-1}), \tag{1.13}$$

$$(p_k) = \frac{r_k^T r_k}{r_{k-1}^T r_{k-1}}. \tag{1.14}$$

Now, we calculate eq. (1.9) by applying eq. (1.15), which is also called search direction.

$$dx = d_k = (r_k) - (p_k d_{k-1}). \tag{1.15}$$

The improvement (p_k) can be computed in several different ways. There are two features of the Powell-Beale variation. First, we easily know when to reset the search direction to the gradient. Second, the search direction is computed from the negative gradient.

Many authors used CGB in neural network. For example, Wisesty got (2017) average accuracy 79.11% of Polak Ribiere, 93.20% average accuracy by Fletcher Reeves and 86.74% accuracy by Powell/Beale [40].

1.7 Fletcher-Powell Conjugate Gradient (CGF)

Fletcher *et al*. [11] proposed this algorithm. The entire conjugate gradient algorithm starts out by searching in the steepest descent direction (negative of the gradient) on the first iteration (eq. (1.16)).

$$P_o = -g_o\,. \tag{1.16}$$

g is the gradient value of input value and p is the search direction.

A line search is then performed to determine the optimal distance to move along the current search direction (eq. (1.17)).

$$x_{k+1} = x_k \alpha_k P_k\,, \tag{1.17}$$

where $\alpha_k P_k$ is search direction.

Then the next search direction is determined. This is determined with the help of eq. (1.18).

$$P_k = -g_k + \beta_k P_{k-1}\,. \tag{1.18}$$

Here, g_k is the gradient and β_k is computed with the help of eq. (1.19) knows as Fletcher-Reeves update method.

$$\beta_k = \frac{g_k^T g_k}{g_{k-1}^T g_{k-1}}\,. \tag{1.19}$$

This is the ratio of the norm squared of the current gradient to the norm squared of the previous gradient.

Here, weight and bias variables of X is adjusted according to the eq. (1.20) known as search direction equation.

$$dX = d_k = -gX + old_g X * \beta_k\,, \tag{1.20}$$

where gX is the gradient of the each iteration. This search direction value (dX) is put up into the eq. (1.9) for adjusting the weight and bias values of X.

Many authors used CGF in neural network. For example, Zhang *et al.* [44] showed that prediction accuracy using Davidon-Fletcher-Powell Second order Volterra filter (DFP-SOVF) model is better than that of linear prediction (LP). Abd-Ellah *et al.* showed that the traincgf (training algorithm conjugate gradient with Fletcher-Reeves updates) provides the best performance in comparison with traincgp (conjugate gradient with Polak-Ribiere updates), trainrp (resilient backpropagation), traincgb (Conjugate gradient Peal restarts) in terms of root mean square error (RMSE) values. Yumei *et al.* [42] showed that Davidon-Fletcher-Powell (DFP) algorithm is better than Normalized Least Mean Square (NLMS) algorithm. Zhang and Bai [43] showed that Davidon-Fletcher-Powell (DFP) algorithm always guarantee its stability and convergence properties.

1.8 Polak-Ribiere Conjugate Gradient (GDM)

This algorithm is described by Scales [35]. For the Ploak-Ribiere update, the constant β_k of eq. (1.19) can be computed with the help of eq. (1.21).

$$\beta_k = \frac{\delta\_g_{k-1}^T g_k}{g_{k-1}^T g_{k-1}}. \tag{1.21}$$

This is the inner product of the previous change in the gradient with the current gradient divided by the norm squared of the previous gradient.

Many authors used GDM in neural network. For example, Yang *et al.* [41] recognized signal modulation with the help of Polak-Ribiere algorithm in neural network. Pratiwi and Aditsania [31] detected cancer with the help of Genetic Bee Colony (GBC) and conjugate gradient backpropagation with modified polak ribiere (MBP-GP). Destiani and Utama [10] extracted features of EEG with the help of Artificial Neural Network (ANN) modified backpropagation (MBP) Polak-Ribiere Conjugate Gradient with line search techniques. Gao *et al.* [13] proposed a neuro-fuzzy network with the help of a Polak-Ribiere-Polyak conjugate gradient based algorithm. In this paper, *we show that GDM takes more epochs*.

1.9 One Step Secant (OSS)

This algorithm is described by Roberto [33]. In the eq. (1.9), dX is the search direction. The parameter 'a' is selected to minimize the performance along the search direction. For locate the minimize point, we use line search function. The first search direction is the negative of the gradient of performance. In the succeeding iterations the search direction is computed with the help of eq. (1.22).

$$dX = -gX + Ac * X\_step + Bc * dgX, \tag{1.22}$$

where gX is the gradient, Ac is a collective value of last iterations, Bc is a condition of last iteration, X_step is the change in the weights on the previous iteration, and dgX is the change in the gradient from the last iteration.

This algorithm requires less storage and computation per epoch that the BFGS algorithm but more storage and computation per epoch that the conjugate gradient algorithms. Its performance is also between full quasi-Newton algorithm and Conjugate gradient algorithm.

Many authors used OSS in neural network. For example, Khadse *et al.* [20] showed that more than 100 neurons are used for getting desired output using OSS. They also showed that OSS is bad choice for three phase power quality monitoring. Hassan and Abraham [17] showed that OSS was slowest algorithm.

1.10 Variable Learning Rate Backpropagation (GD)

For calculating derivation of performance with respect to weight and bias variables X, we used backpropagation. We used eq. (1.23) known as search direction (dX) for calculating eq. (1.9).

$$dX = mc * dX\_prev + lr * mc * dperf/dX, \tag{1.23}$$

where 'dX_prev' is the previous change search direction value to the weight or bias value. The mc is a momentum constant value, lr is a learning rate value and $dperf/dX$ is the differentiation value of performance with respect to weight or bias.

Many authors used variable learning rate backpropagation (VLBP) in neural network. For example, Atanassov *et al.* [3] made a generalized net for parallel optimization of Multi-Layer Feedforward Neural Network on assigned training pairs with variable learning rate backpropagation algorithm. Bhati *et al.* [5] showed that single layer neural network can give 100% recognition accuracy if correct learning rate is assigned to it. Li *et al.* [23] showed that the improved VLBP can quickly and accurately predict the concentration of target elements in Energy Dispersive X-Ray Fluorescence (EDXRF). In this paper, *we show that GD takes more epochs.*

2. Experiments

We do this experiment on Intel core i5, Window 8.1 and MATALB 2020a. We take total 30 images from CalTech (Figure 1) from different person's position and light conditions [29].

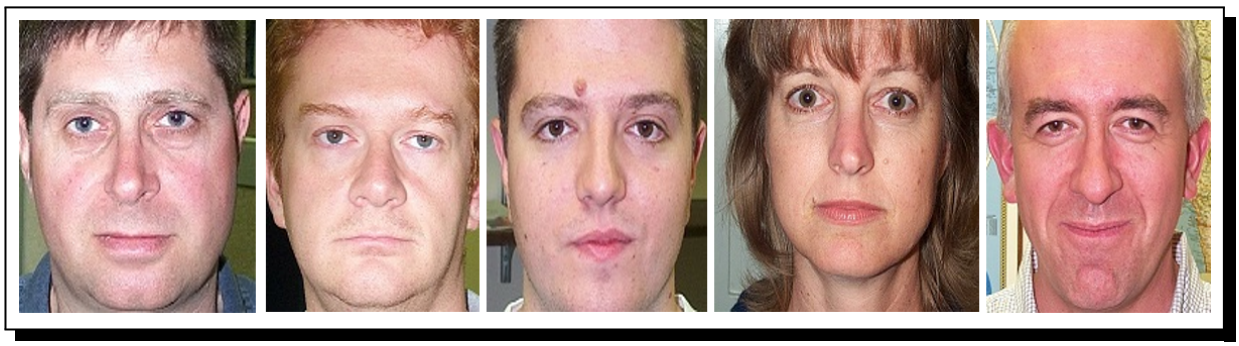


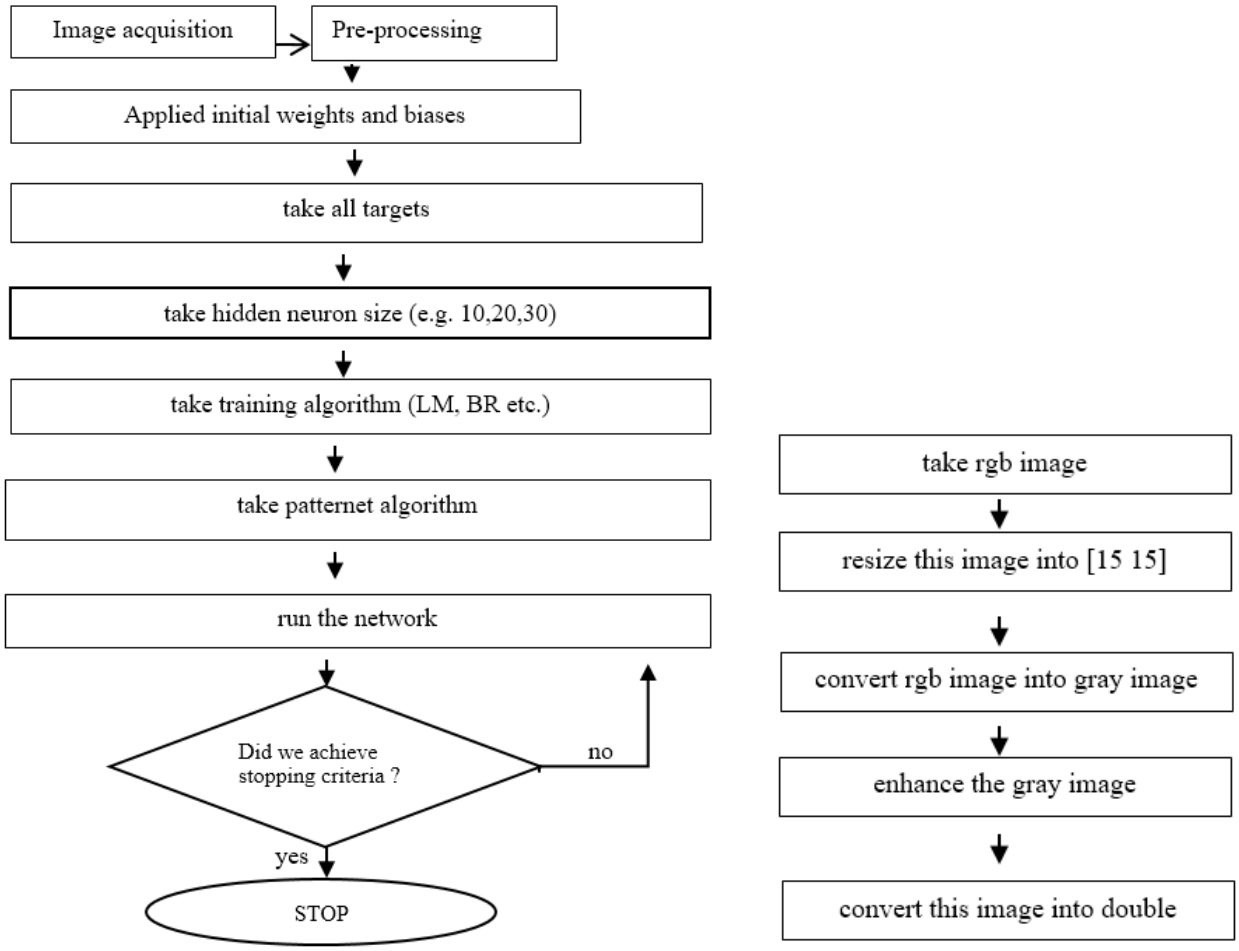
Figure 1. Sample of Caltech 101 face images [29]

Algorithm. We do following steps for this purpose: (Part A of flowchart 1)

Step 1: We take all the images from datasets.

Step 2: We do pre-processing on it by the following sub-steps: (Part B of flowchart 1):

(i) Resize it into $15 * 15$.
(ii) Then we convert RGB images into gray images.
(iii) For looking good contrast and removing some illumination effect from an image, we do histogram equalization.
(iv) Then, we do double operation on it.
(v) Then, we save all the data.



(a) Main flow chart (b) Pre-processing flow chart

Flowchart 1. Methodology of this research. First Part A is Main flow chart and second Part B is Pre-processing chart

Step 3: Now, we make pattern neural network. For this purpose, in the first layer of the network, we take size of inputs = 225. In the second layer of the network (i.e., hidden layer), we test the performance of the network with different hidden neurons (n) (i.e., 10, 20 and 30). Then we use sigmoid function. In the third layer (i.e., output layer) we take five output values, we initialized the weights and the biases, and then apply the softmax function. (Here, we may also use Hyperbolic Tangent sigmoid function. In Hyperbolic Tangent sigmoid function, if n is input then output is $\frac{e^n-e^{-n}}{e^n+e^{-n}}$ [15,18]. In MATLAB we use tansig command for Hyperbolic Tangent sigmoid function. In softmax function, if n is input then output is exp(n)/sum(exp(n)) [15, 18]. In MATLAB we use softmax command for softmax function. Here, soft means softmax is continuous and differentiable [14]. Softmax function is now mostly used for the output of the classifier [14].) (Figure 2)

Now, for performing a network, we had taken minimum value of grade performance is 1e-10, value of goal performance is 1e-10 and value of epochs is 1000. After that we train the network. Our network stops this training whenever stopping criteria is matched. After completing the

training process, we got exact weight and bias of the network for the both layers. Then we calculate accuracy (Table 1 to 3). We plot confusion matrix. Figure 3 is a confusion matrix of Levenberg-Marquardt training algorithm (train_lm) and Figure 4 is a confusion matrix of Rprop training algorithm (train_rp).

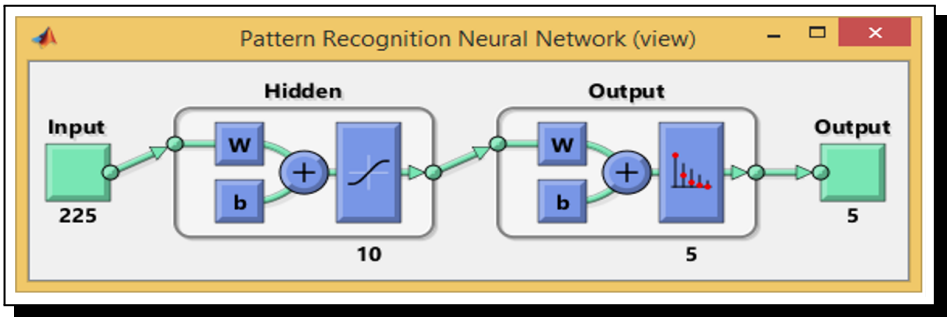


Figure 2. The pattern Network. Here image size is $15 * 15$. So, total inputs are 225. This is an example of 10 hidden neurons and output are 5. In hidden layer we use tansig function and in output layer, we use softmax function

Table 1. When hidden layer size = 10

Sr.	Training Algorithm	Mean time (S)	Accuracy (%)	Epochs
1	LM	5.5685	100	10
2	BR	5.5978	20	4
3	BFG	106.0750	100	13
4	RP	0.6091	96.7	10
5	SCG	1.8723	100	31
6	CGB	0.9749	100	23
7	CGF	0.6839	100	22
8	OSS	0.7949	100	13
9	GDM	2.4290	100	1000
10	GD	2.4521	100	1000

Table 2. When hidden layer size = 20

Sr.	Training Algorithm	Mean time (S)	Accuracy (%)
1	LM	21.1191	100
2	BR	36.8752	43.3
3	BFG	465.6800	86.7
4	RP	0.6241	80
5	SCG	0.6719	96.7
6	CGB	0.7188	96.7
7	CGF	0.6831	86.7
8	OSS	0.5937	86.7
9	GDM	2.5455	96.7
10	GD	2.5761	96.7

Table 3. When hidden layer size = 30

Sr.	Training Algorithm	Mean time (S)	Accuracy(%)	Epochs
1	LM	101.488	100	11
2	BR	136.578	56.7	4
3	BFG	More times taken		
4	RP	0.6431	100	9
5	SCG	0.7961	100	29
6	CGB	0.7761	100	20
7	CGF	0.7056	100	22
8	OSS	0.6155	100	12
9	GDM	2.6558	100	1000
10	GD	2.5933	100	1000

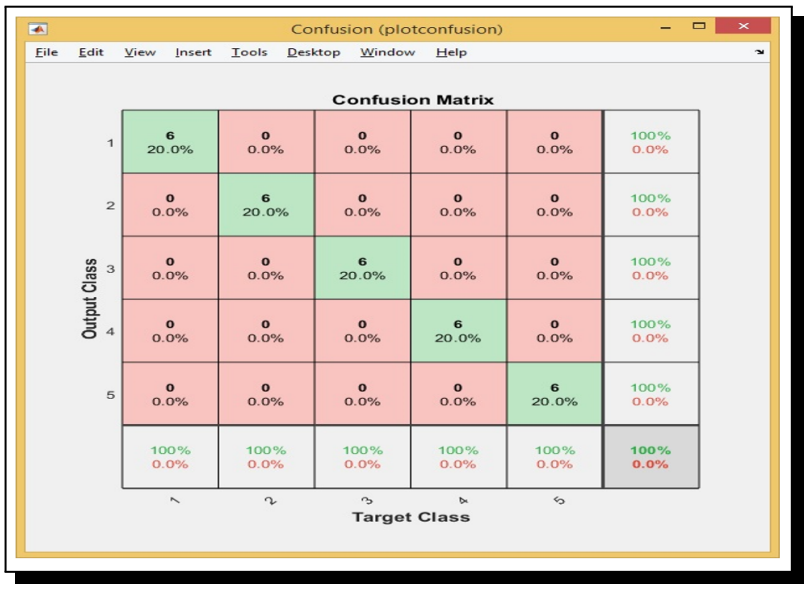
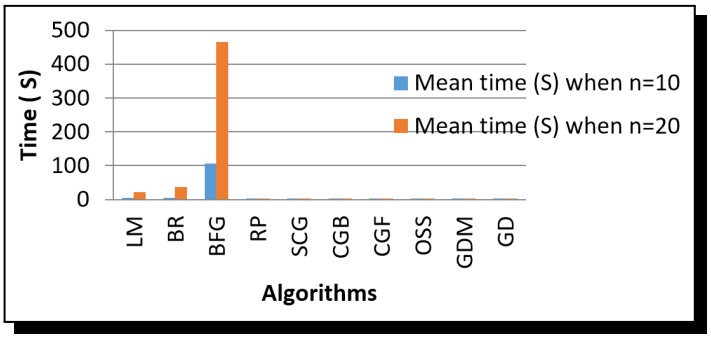


Figure 3. Confusion matrix made with the help of train_lm

Figure 4. Confusion matrix made with the help of train_rp



Graph 1. Time vs. Algorithm

In Graph 1, BFG algorithm takes more time with compare to other algorithms. SCG, CGB, CGF and OSS are the fastest algorithms.

3. Results and Discussion

With the help of 30 images, for getting better accuracy, we used pattern network with different hidden neurons (n) (such as 10, 20 and 30) and different training algorithms (such as LM, BR etc.). After analysis of different images on different hidden neurons and different training algorithms, we found the following results:

(1) In Table 1, time taken by RP is 0.6091s, time taken by SCG is 1.8723s, time taken by CGB is 0.9749s, time taken by CGF is 0.6839s and time taken by OSS is 0.7949s. In Table 2, time taken by RP is 0.6241s, time taken by SCG is 0.6719s, time taken by CGB is 0.7188s, time taken by CGF is 0.6831s and time taken by OSS is 0.5937s. In Table 3, time taken by RP is 0.6431s, time taken by SCG is 0.7961s, time taken by CGB is 0.7761s, time taken by CGF is 0.7056s and time taken by OSS is 0.6155s. This means we say that RP, SCG, CGB, CGF and OSS are fast algorithms. (Graph 1)

(2) BFG takes 106.075s time in Table 1, 465.68s time in Table 2 and more times in Table 3. This means we say that BFG takes more time with respect to hidden neuron size. (Graph 1)

(3) In Table 1 and Table 2, GDM and GD take 1000 epochs (maximum) for getting 100% accuracy. This means we say that GD and GDM take more epochs.

(4) In Table 1, BR gives 20% accuracy, in Table 2 BR gives 43.3% accuracy and in table 3 BR gives 56.7%. This means we say that BR algorithm is not acceptable for image categorization.

4. Conclusion

In this survey paper, we used ten type of training algorithm. We used Levenberg-Marquardt (LM), Bayesian regularization backpropagation (BR), BFGS Quasi-Newton (BFG), Resilient Backpropagation (RP), Scaled Conjugate gradient backpropagation (SCG), Conjugate Gradient with Powell/Beale Restarts (CGB), Fletcher-Powell Conjugate Gradient (CGF), Polak-Ribiere Conjugate Gradient (GDM), One Step Secant (OSS), and Variable Learning Rate

Backpropagation (GD) algorithm. So, first of all we explain functions of these ten types of training algorithms. Then we design a pattern network. We test all these ten algorithms on our designed network with the help of 30-images. Our main aim of this survey is to find out which type of training algorithm is fastest for an image categorization problem and affects of these algorithms on neural network when we applied different types of hidden neuron size. During this survey we found some new facts. We found that RP, SCG, CGB, CGF and OSS are fastest algorithms. BFG takes more time with respect to hidden neuron size. GDM and GD take 1000 epochs (maximum) for 100% classifications. BR gives 20%, 43.3% and 56.7% accuracy. So, BR algorithm is not acceptable for image categorization. We consider RP, SCG, CGB, CGF and OSS algorithm in the case of image categorization.

Competing Interests

The authors declare that they have no competing interests.

Authors' Contributions

All the authors contributed significantly in writing this article. The authors read and approved the final manuscript.

References

[1] A.R. Abd-Ellah, M.H. Essai and A. Yahya, Comparison of different backpropagation training algorithms using robust M-estimators performance functions, *IEEE 10th International Conference on Computer Engineering & Systems* (ICCES), (2015), 384 – 388, DOI: 10.1109/ICCES.2015.7393080.

[2] C.K. Arthur, V.A. Temeng and Y.Y. Ziggah, Performance evaluation of training algorithms in backpropagation neural network approach to blast-induced ground vibration prediction, *Ghana Mining Journal* **20**(1) (2020), 20 – 33, DOI: 10.4314/gm.v20i1.3.

[3] K. Atanassov, M. Krawczak and S. Sotirov, Generalized net model for parallel optimization of feed-forward neural network with variable learning rate backpropagation algorithm, *IEEE 4th International IEEE Conference Intelligent Systems*, (2008), 2, 16, DOI: 10.1109/IS.2008.4670540.

[4] C. Baigen, Y. Jiaming, S. Weif and C. Bin, Research on fault recognition method of on-board equipment based on BP neural network optimized by Bayesian regularized, *IEEE Chinese Automation Congress (CAC)*, (2017), 6835 – 6840, DOI: 10.1109/CAC.2017.8244008.

[5] R. Bhati, S. Jain, N. Maltare and D.K. Mishra, A comparative analysis of different neural networks for face recognition using principal component analysis, wavelets and efficient variable learning rate, *IEEE International Conference on Computer and Communication Technology* (ICCCT), (2010), 526 – 531, DOI: 10.1109/ICCCT.2010.5640486.

[6] D. Chang, S. Sun and C. Zhang, An accelerated linearly convergent stochastic L-BFGS algorithm, *IEEE Transactions on Neural Networks and Learning Systems* **30**(11) (2019), 3338 – 3346, DOI: 10.1109/TNNLS.2019.2891088.

[7] Y. Cui, Z. Yi, J. Duan, D. Shi and Z. Wang, A Rprop-Neural-Network-Based PV maximum power point tracking algorithm with short-circuit current limitation, *IEEE Power & Energy Society Innovative Smart Grid Technologies Conference* (ISGT), (2019), 1 – 5, DOI: 10.1109/ISGT.2019.8791596.

[8] F. Dalipi and S.Y. Yayilgan, The impact of environmental factors to skiing injuries: Bayesian regularization neural network model for predicting skiing injuries, *IEEE 6th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, (2015), 1 – 6, DOI: 10.1109/ICCCNT.2015.7395218.

[9] J.E. Dennis and J.R.B. Schnabel, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, SIAM Publications, Philadelphia, (1996), URL: https://books.google.co.in/books?hl=en&lr=&id=ksvJTtJCx9cC&oi=fnd&pg=PR1&ots=BKmNCQ5Hxq&sig=g7DGZc4S60CQTlCaXQcOfEcXEYQ#v=onepage&q&f=false.

[10] D.K. Destiani and D.Q. Utama, Study of wavelet and line search techniques on modified backpropagation Polak-Ribiere algorithm for heart failure detection, *IEEE 6th International Conference on Information and Communication Technology* (ICoICT), (2018), 319 – 323, DOI: 10.1109/ICoICT.2018.8528790.

[11] R. Fletcher and C.M. Reeves, Function minimization by conjugate gradients, *The Computer Journal* **7**(2) (1964), 149 – 154, DOI: 10.1093/comjnl/7.2.149.

[12] M. Forouzanfar, H.R. Dajani, V.Z. Groza, M. Bolic and S. Rajan, Comparison of feed-forward neural network training algorithms for oscillometric blood pressure estimation, *IEEE 4th International Workshop on Soft Computing Applications*, (2010), 119 – 123, DOI: 10.1109/SOFA.2010.5565614.

[13] T. Gao, J. Wang, B. Zhang, H. Zhang, P. Ren and N.R. Pal, A Polak-Ribière-Polyak conjugate gradient-based neuro-fuzzy network and its convergence, *IEEE Access* **6** (2018), 41551 – 41565, DOI: 10.1109/ACCESS.2018.2848117.

[14] I. Goodfellow, Y. Bengio and A. Courvlle, *Deep Learning*, MIT Press (2016), URL: https://mitpress.mit.edu/books/deep-learning.

[15] M.T. Hagan, *Neural Network Design*, 2nd edition, Kindle e-Book, (2014), URL: https://hagan.okstate.edu/NNDesign.pdf.

[16] A.N. Handayani, N. Lathifah, H.W. Herwanto, R.A. Asmara and K. Arai, Neural network bayesian regularization backpropagation to solve inverse kinematics on planar manipulator, *IEEE 7th International Conference on Informatics, Electronics & Vision (ICIEV) and 2nd International Conference on Imaging, Vision & Pattern Recognition* (icIVPR), (2018), 99 – 104, DOI: 10.1109/ICIEV.2018.8640958.

[17] A.K.I. Hassan and A. Abraham, Modeling consumer loan default prediction using ensemble neural networks, *IEEE International Conference on Computing, Electrical and Electronic Engineering* (ICCEEE), (2013), 719 – 724, DOI: 10.1109/ICCEEE.2013.6634029.

[18] S. Haykin, *Neural Networks and Learning Machines*, 3rd edition, Pearson Prentice Hall (2009), URL: https://www.pearson.com/us/higher-education/program/Haykin-Neural-Networks-and-Learning-Machines-3rd-Edition/PGM320370.html.

[19] Y. Ju, J. Wang, Z. Zhang, Y. Huang and Y. Lin, A calculation method for three-phase power flow in micro-grid based on smooth function, *IEEE Transactions on Power Systems* **35**(6) (2020), 4896 – 4903, DOI: 10.1109/TPWRS.2020.2995521.

[20] C.B. Khadse, M.A. Chaudhari and V.B. Borghate, Comparison of seven backpropagation algorithms for three phase power quality assessment, *IEEE TENCON IEEE Region 10 Conference*, (2017), 2548 – 2553, DOI: 10.1109/TENCON.2017.8228291.

[21] T. Kliment, P. Lipovsky, K. Draganova and V. Moucha, Comparison of Speed and Accuracy of Chosen Magnetometer Calibration Algorithms in the Presence of External Interference, *IEEE New Trends in Signal Processing* (NTSP), (2018), 1 – 5, DOI: 10.23919/NTSP.2018.8524105.

[22] S. Leholo, P. Owolawi and K. Akindeji, Solar Energy Potential Forecasting and Optimization Using Artificial Neural Network: South Africa Case Study, *IEEE Amity International Conference on Artificial Intelligence* (AICAI), (2019), 533-536, DOI: 10.1109/AICAI.2019.8701372.

[23] F. Li, L. Ge, W. Dan, Y. Gu, Q. He and K. Sun, Application of improved variable learning rate back propagation neural network in energy dispersion x-ray fluorescence quantitative analysis, *IEEE 4th International Conference on Cloud Computing and Big Data Analysis* (ICCCBDA), (2019), 428 – 432, DOI: 10.1109/ICCCBDA.2019.8725682.

[24] S.Y. Liang and Y.T. Ning, Parameter optimization of load frequency control system composed of hydroelectric and thermal power units based on Levenberg-Marquardt algorithm, *IEEE 5th Asia Conference on Power and Electrical Engineering* (ACPEE), (2020), 75 – 80, DOI: 10.1109/ACPEE48638.2020.9136561.

[25] J. Lv, S. Deng and Z. Wan, An efficient single-parameter scaling memoryless Broyden-Fletcher-Goldfarb-Shanno algorithm for solving large scale unconstrained optimization problems, *IEEE Access* **8** (2020), 85664 – 85674, DOI: 10.1109/ACCESS.2020.2992340.

[26] D.W. Marquardt, An algorithm for least-squares estimation of nonlinear parameters, *Journal of the Society for Industrial and Applied Mathematics* **11**(2) (1963), 431 – 441, DOI: 10.1137/0111030.

[27] M.F. Møller, A scaled conjugate gradient algorithm for fast supervised learning, *Neural Networks* **6**(1993), 525 – 533, DOI: 10.1016/S0893-6080(05)80056-5.

[28] A. Payal, C.S. Rai and B.V.R. Reddy, Comparative analysis of Bayesian regularization and Levenberg-Marquardt training algorithm for localization in wireless sensor network, *IEEE 15th International Conference on Advanced Communications Technology (ICACT)*, (2013), 191 – 194, URL: https://ieeexplore.ieee.org/abstract/document/6488169.

[29] Perona Lab, http://www.vision.caltech.edu (Photos/Pictures), Caltech Vision Lab, (2022).

[30] M.J.D. Powell, Restart procedures for the conjugate gradient method, *Mathematical Programming* **12**(1) (1977), 241 – 254, DOI: 10.1007/BF01593790.

[31] M.S. Pratiwi and A. Aditsania, Cancer detection based on microarray data classification using Genetic Bee Colony (GBC) and conjugate Gradient Backpropagation with modified Polak Ribiere (MBP-CGP), *IEEE International Conference on Computer, Control, Informatics and its Applications* (IC3INA), (2018), 163 – 168, DOI: 10.1109/IC3INA.2018.8629538.

[32] M. Riedmiller and H. Braun, A direct adaptive method for faster backpropagation learning: the RPROP algorithm, *IEEE International Conference on Neural Networks*, San Francisco, CA, USA, (1993), (1), 586 – 591, DOI: 10.1109/ICNN.1993.298623.

[33] B. Roberto, First- and second-order methods for learning: between steepest descent and Newton's method, *Neural Computation* **4**(2) (1992), 141 – 166, DOI: 10.1162/neco.1992.4.2.141.

[34] S. Särkkä and L. Svensson, Levenberg-Marquardt and line-search extended Kalman smoothers, *IEEE International Conference on Acoustics, Speech and Signal Processing* (ICASSP), (2020), 5875 – 5879, DOI: 10.1109/ICASSP40776.2020.9054686.

[35] L.E. Scales, *Introduction to Non-Linear Optimization*, Macmillan International Higher Education (1985), DOI: 10.1007/978-1-349-17741-7.

[36] The MathWorks, Inc., *Deep Learning Toolbox* https://www.mathworks.com/help/deeplearning, (2022).

[37] P.K. Upadhyay and C. Nagpal, SCG backpropagation based prediction of stressed EEG spectrum, *IEEE Advances in Science and Engineering Technology International Conferences* (ASET), (2020), 1 – 5, DOI: 10.1109/ASET48392.2020.9118324.

[38] P.K. Upadhyay, A. Pandita and N. Joshi, Scaled conjugate gradient backpropagation based SLA violation prediction in cloud computing, *IEEE International Conference on Computational Intelligence and Knowledge Economy* (ICCIKE), (2019), 203 – 208, DOI: 10.1109/ICCIKE47802.2019.9004240.

[39] J. Wang, B. Liu, C. Zhang, F. Yang, T. Zhang and X. Miao, Partial discharge type identification in GIS based on SCG algorithm, *IEEE 3rd International Electrical and Energy Conference* (CIEEC), (2019), 511 – 515, DOI: 10.1109/CIEEC47146.2019.CIEEC-2019213.

[40] U.N. Wisesty, Comparative study of conjugate gradient to optimize learning process of neural network for Intrusion Detection System (IDS), *IEEE 3rd International Conference on Science in Information Technology* (ICSITech), (2017), 459 – 464, DOI: 10.1109/ICSITech.2017.8257156.

[41] F. Yang, Z. Li, H. Li, H. Huang and Z. Pan, Method of neural network modulation recognition based on clustering and Polak-Ribiere algorithm, *Journal of Systems Engineering and Electronics* **25**(5) (2014), 742 – 747, DOI: 10.1109/JSEE.2014.00085.

[42] Z. Yumei, B. Shulin, C. Ping and Q. Shiru, A novel second-order DFP-based volterra filter and its applications to chaotic time series prediction, in: *Proceedings of the IEEE 31st Chinese Control Conference*, (2012), 1036 – 1039, URL: https://ieeexplore.ieee.org/abstract/document/6390076.

[43] Y. Zhang and S Bai, A novel adaptive filter algorithm based on DFP technique, in: *Proceedings of the IEEE 30th Chinese Control Conference*, (2011), 1688 – 1691, URL: https://ieeexplore.ieee.org/abstract/document/6001590.

[44] Y. Zhang, S. Bai, G. Lu and X. Wu, Kernel estimation of truncated Volterra filter model based on DFP technique and its application to chaotic time series prediction, *Chinese Journal of Electronics* **28**(1) (2019), 127 – 135, DOI: 10.1049/cje.2018.04.014.

[45] R. Zhao, W.B. Haskell and V.Y. Tan, Stochastic L-BFGS: Improved convergence rates and practical acceleration strategies, *IEEE Transactions on Signal Processing* **66**(5) (2018), 1155 – 1169, URL: http://www.auai.org/uai2017/proceedings/papers/86.pdf.

[46] N.S.A. Zubir, M.A. Abas, N. Ismail, N.A.M. Ali, M.H.F. Rahiman, N.K. Mun and N.T. Saiful, Pattern classifier of chemical compounds in different qualities of agarwood oil parameter using scale conjugate gradient algorithm in MLP, *IEEE 13th International Colloquium on Signal Processing & its Applications* (CSPA), (2017), 18 – 22, DOI: 10.1109/CSPA.2017.8064917.