Comparison of Neural Network Training Algorithms for Classification of Heart Diseases

Hesam Karim, Sharareh R. Niakan, Reza Safdari
Department of Health Information management, Tehran University of Medical Sciences, Iran

Article Info

Article history:

Received Jul 11, 2018
Revised Oct 10, 2018
Accepted Oct 2, 2018

Keyword:

Heart Disease
Machin Learning
Medical Informatics
Neural Network
Training Algorithms

ABSTRACT

Heart disease is the first cause of death in different countries. Artificial neural network (ANN) technique can be used to predict or classification patients getting a heart disease. There are different training algorithms for ANN. We compared eight neural network training algorithms for classification of heart disease data from UCI repository containing 303 samples. Performance measures of each algorithm containing the speed of training, the number of epochs, accuracy, and mean square error (MSE) were obtained and analyzed. Our results showed that training time for gradient descent algorithms was longer than other training algorithms (8-10 seconds). In contrast, Quasi-Newton algorithms were faster than others (<=0 second). MSE for all algorithms was between 0.117 and 0.228. While there was a significant association between training algorithms and training time ($p<0.05$), the number of neurons in hidden layer had not any significant effect on the MSE and/or accuracy of the models ($p>0.05$). Based on our findings, for development an ANN classification model for heart diseases, it is best to use Quasi-Newton training algorithms because of the best speed and accuracy.

Corresponding Author:

Reza Safdari,
Department of Health Information Management,
Tehran University of Medical Sciences,
Enghelab Ave, Ghods Ave, Farredanesh Alley, No 17, Tehran, Iran.
Email: rsafdari@tums.ac.ir

1. INTRODUCTION

In recent decades, a large amount of data is produced in healthcare industry about patients. These data are a good resources to be analyzed for knowledge extraction that enables best decision making [1, 2]. In order to conduct data analyzing in the medical domain, there are various approaches containing statistics, data mining and machine learning methods. One popular method of these approaches is the artificial neural network (ANN).

ANNs provide a powerful tool to analyze and model the data across a broad range of medical applications. Most applications of ANNs in medicine are classification problems which assign an input data to one of a set of classes in output level [3, 4]. A neural network has to be configured such that the application of a set of inputs produces the desired set of outputs [5, 6]. The use of ANN has three important steps for any purposes including training, testing and validation [7]. For configuring the ANN, it must train the neural network by teaching patterns through changing their weights according to some learning rules. Training of the neural networks can be done by various suggested algorithms [4, 8]. Different types of training algorithms were compared in various fields and their pros and cons have been analyzed [9-12]. However, no studies have been conducted in the cardiovascular domain. One of the areas of healthcare where the data are growing up is the cardiovascular field. Heart disease is the first cause of death in different countries and accounts for approximately 80% of all deaths. Based on WHO report, about 12 million deaths per year occur in the world due to the heart diseases. The term heart disease comprises the various diseases that affect the heart [1, 13, 14]. Efforts to improve lifestyles and control risk factors will definitely contribute to heart disease prevention.

Indeed, the predictive and diagnosis of heart diseases in the early stage should be done to reduce the risk of heart disease and is vital for the prevention of patient's deaths [1, 13, 14].

In order to diagnose heart diseases, there are various ways including physical examination, echocardiogram, cardiac nuclear scan, and angiography. However, physicians diagnose heart disease by learning and experience. Because of human mistakes, diagnostic methods might be less accurate and lead to errors, false presumptions and unpredictable effects [1]. Thus mathematical algorithms such as ANNs have been used to classify heart diseases [15]. Among all applied data mining methods, ANNs have had an acceptable performance and known as a valuable algorithm for heart disease classification [16]. In the learning process, understanding the best structure and function to obtain the best result is crucial; otherwise, there would be time and cost consuming if they are found by try and error. For ANNs algorithm application in the area of heart disease, the best method and structure is not known yet. This study is aimed to compare some ANN training algorithms and find out the best method for classification of heart diseases.

2. RESEARCH METHOD

This was a prospective cross-sectional study that measured and compared performance and functionality of artificial neural network training algorithms for classification of heart diseases. Dataset taken from UCI machine learning repository [17] was used to develop the ANN-based models. The database contains 303 samples with 76 attributes. However, we used only 13 most important attributes listed in Table 1. The predict attribute was diagnosis of heart disease in which its value is '0' if diameter narrowing =<50% (no heart disease) and is '1' if this parameter is >50% (positive heart disease). For ANNs learning process, data was divided into three sets for training (60%), validation (20%) and testing (20%). To avoid possible bias in the presentation order of the sample patterns to the ANN, these sample sets were randomized.

Table 1. Attributes of heart diseases data used in developing ANN

Variable	Variable Definition	Categories of Values
Age	Age of patient	[29-77]
Sex	Gender of patient	(1 = male; 0 = female)
CP	Chest pain type	[1-4]
RBP	Resting blood pressure	[94-200]
SC	Serum cholesterol in mg/dl	[126,564]
FBS	Fasting blood sugar > 120 mg/dl	[0-1]
RER	Resting electrographic results	[0-2]
MHRA	Maximum heart rate achieved	[71-202]
EIA	Exercise induced angina	[0-1]
Old-peak	ST depression induced by exercise relative to rest	[0-6.2]
Slope	Slope of the peak exercise ST segment	[1-3]
NUM	Number of major vessels colored by fluoroscopy	[0-3]
Def-t	Defect type (normal, fixed, reversible defect)	[3,6,7]
Diagnosis	Class of heart disease	0 (no heart disease) or 1 (has heart disease)

Table 2. All training functions for conducting ANN

Training Algorithm	Training Function	Description
Gradient Descent	GD	Gradient descent back-propagation
	GDM	Gradient descent with momentum back-propagation
	RP	Resilient back-propagation (Rprop)
Conjugate Gradient	SCG	Scaled conjugate gradient back-propagation
	CGP	Conjugate Gradient back-propagation with Polak-Rieber Updates
	CGF	Fletcher-Powell conjugate gradient back-propagation
Quasi-Newton	BFG	BFGS quasi-Newton back-propagation
	LM	Levenberg-Marquardt back-propagation

In order to develop Multilayer Perceptron Neural Networks (MLPNN), we used three main training algorithms (GD: Gradient Descent, CG: Conjugate Gradient, Quasi-Newton) containing eight training functions described in table 2. The sigmoid transfer function is used for the hidden layer. Basic system training parameters are max_epochs=1000, show=5, performance goal=0, time=Inf, min_grad=1e-010, max_fail=6 are fixed for each training function. Finally, performance evaluation of each training function conducted with measuring and comparing the speed of training (time), number of epoch at the end of training, correct classification percentage (accuracy), regression on training, regression on validation and mean square error (MSE) as the evaluation criteria of each function. All these parameters were checked for 10, 20 and 30 number of neurons in the hidden layer. One-way analysis of variance (ANOVA) was used to determine whether there are any statistically significant differences between the means of performance measures for all training algorithms. ANN toolbox in MATLAB 2010 was used to construct neural networks for diagnosing of the heart disease. SPSS (version 2015) also used for statistical data analysis. All these experiments were carried out on Windows 7 (32-bit) operating system with Intel(R) Core(TM) i5 2.50GHz processor and 6 GB RAM.

3. RESULTS AND DISCUSSION

In this study, we compared the performance of eight ANN training function for heart disease classification. The result of this evaluation is shown in table 3. As shown in table 3, training time ranges between 8 and 10 seconds for GD and GDM (gradient descent with momentum) respectively. Time measurement for remain algorithms was in a rage of 0-2 seconds. Training process ended in epoch 1000 for GD and GDM algorithms. All other algorithms ended in epoch 2-22. Average of accuracy for Quasi-Newton algorithms (86.06%), GD (83.13) and CG (83.14) were obtained. Maximum and minimum regression value on training were 0.999 (LM: Levenberg-Marquardt back-propagation) and 0.173 (CGF: Conjugate Gradient back-propagation with Fletcher-Reeves Updates), respectively. MSE for all algorithms was between 0.117 and 0.228. Based on results of variance analysis showed in table 4, statistically, there was no significant difference between MSE/Accuracy in groups of algorithms and number of hidden layers ($p>0.05$). Between training algorithms and training time, there was a significant association ($p<0.05$). The mean training time for GD and GDM was 9.3 and 8.3 seconds respectively. In return, the mean training time for RP (resilient back-propagation) (0 sec.), LM and CGF (0.33 sec.) were obtained and reported in Table 3.

Training of the neural networks can be done by different optimization algorithms [7, 8]. In this study, we compared three main classes of training algorithms containing eight methods for classification of heart diseases. One of the main measurements for evaluation of each algorithm was accuracy. Based on our results the maximum accuracy was for Quasi-Newton algorithms (91.75%). Quasi-Newton methods exploit gradient information to approximate the Hessian matrix of the error function with respect to the parameters of the network. This approximation matrix is subsequently used to determine an effective search direction and update the values of the parameters [18]. The effectiveness of training algorithms was measured by mean squared error (MSE). Although some studies believe that networks are sensitive to the number of neurons in their hidden layers [19], we did not find any significant association between the number of neurons in hidden layers and models accuracy, and MSE. We used regression analysis function in order to compare the actual outputs the algorithms with the desired outputs. Maximum regression value on training was for LM algorithm. Our results about regression values is similar to the result of Sharma’s study [9]. It shows that the correlation coefficient (R) between actual and desired output in LM algorithm is acceptable, so, this algorithm is proper to classification task.

Another performance measure evaluated in this study was computation time of training algorithms. Based on our findings, simple GD and GDM algorithms run slower than others. GD algorithm is known as steepest descent start with a random weight vector. The weight vector will be modified iteratively until a minimum in the error surface is found [20-22]. GD takes many small steps to reach the minimum error; therefore, its relatively slow and inefficient [22]. Although some algorithms such as the GDM and RP have been proposed for improving the speed of convergence of GD algorithms, our results showed a lower execution time for GDM. The momentum variation is usually faster than simple GD because it allows higher learning rates [19]. However, RP execution time was faster than GD and GDM (near 0 sec). RP training algorithm known as Rprop changes the weight vector according to separate update value. This algorithm is easy to compute local learning scheme and easy to implement; it is due to no choice of parameters requirement at all process to obtain optimal convergence times. The number of learning steps is significantly reduced in comparison to the original gradient-descent procedure [23] thus RP is faster than GD and GDM.

Our finding showed low execution time for SCG (Scaled conjugate gradient), CGP (Conjugate Gradient back-propagation with Polak-Rieber Updates), and CGF as CG algorithms. CG algorithm implemented as an iterative algorithm. It starts out by searching in the negative of the gradient and then performs a line search to determine the optimal distance to move along the current search direction. Searching along with conjugate directions leads to faster convergence than steepest descent directions [24, 25]. The SCG method was designed to avoid the time-consuming line search in CG algorithms. This algorithm requires more iterations to converge rather than the other CG algorithms; however, the number of computations in each step is significantly reduced as no line search is performed [19]. CGF is an updated version of CG which computes new search direction as the ratio of the norm squared of the current gradient to the norm squared of the previous gradient[26-28]. CGP calculates new search direction as the ration of the inner product of the previous change in the gradient with the current gradient divided by the norm squared of the previous gradient [9, 28]. Generally, the execution time of Quasi-Newton algorithms was similar to CG algorithms. In Newton methods, a quadratic approximation is used instead of a linear approximation of the error function. The main advantage of the

Newton methods is that it has a quadratic convergence rate while the steepest descent has a much slower linear convergence rate. However, each step of this method requires a large amount of computation [29]. A variety of algorithms were designed base on Newton methods. BFGS (Broyden–Fletcher–Goldfarb–Shanno) algorithm is an iterative method for solving unconstrained nonlinear optimization problems that uses an approximate Hessian matrix in computing the search direction [29]. LM algorithm was designed to approach second-order training speed without having to compute the Hessian matrix. This algorithm appears to be the fastest method for training moderate-sized feed-forward neural networks[19, 30] but is not suitable for a large number of data [31]. The main drawback of the LM is that it requires the storage of some matrices that can be quite large for certain problems[19]. CG algorithms are characterized by low memory requirements, fast and strong local and global convergence properties [32]. Thus, it can be used to sparse systems that dimension are too large and to solve unconstrained optimization problems [25, 33]. The storage requirements for CGP (four vectors) are slightly larger than for CGF [9, 28].

Some important factors such as training time, memory need and accuracy must be considered in order to choose the best training algorithm. According to the finding of this study, GD and GDM algorithms are too slow; in contrast, training algorithms based on Newton method converge in less iteration and are faster and more accurate. In addition, the CG algorithms require more storage than the other algorithms. It is better to use LM training for small and medium-size networks if there is enough memory. For large networks, SCG or RP algorithms are a suitable choice[19, 24, 33]. Finally, Quasi-Newton methods are generally considered more powerful compared to other training algorithms [18].

Table 3. Comparison of ANN Training Functions based on the values of Accuracy, time and neuron number in hidden layer

Training Algorithm	Training Function	H	MSE	Epoch	R Train	R Validation	Accuracy	Execution Time (Sec)
Gradient Descent	GD	10	0.117	1000	0.606	0.735	83.50	9
		20	0.173	1000	0.717	0.573	81.85	9
		30	0.131	1000	0.692	0.681	81.85	10
	GDM	10	0.175	1000	0.755	0.578	84.49	9
		20	0.175	1000	0.734	0.608	81.19	8
		30	0.200	1000	0.695	0.564	81.52	8
	RP	10	0.138	6	0.787	0.632	84.16	0
		20	0.154	14	0.815	0.638	84.82	0
		30	0.202	19	0.862	0.489	84.82	0
Conjugate Gradient	SCG	10	0.122	17	0.838	0.677	86.90	1
		20	0.121	14	0.817	0.657	84.49	1
		30	0.191	13	0.837	0.526	83.50	0
	CGP	10	0.189	4	0.191	0.324	80.86	1
		20	0.148	5	0.346	0.634	81.85	0
		30	0.155	16	0.359	0.583	87.09	1
	CGF	10	0.112	10	0.532	0.705	83.50	1
		20	0.136	11	0.507	0.654	84.16	0
		30	0.228	3	0.173	0.231	75.91	0
Quasi-Newton	BFG	10	0.111	22	0.401	0.504	86.47	1
		20	0.160	15	0.333	0.405	85.15	2
		30	0.124	8	0.371	0.665	86.80	2
	LM	10	0.139	5	0.888	0.788	82.51	0
		20	0.165	2	0.952	0.875	83.83	0
		30	0.153	8	0.999	0.858	91.75	1

H: Number of neurons in hidden layer.
MSE: Mean of Square Error.
R: Regression.

Table 4. One-way ANOVA result for comparing means of performance measures in any training algorithms

Performance	Training algorithm		Number of hidden layer	
measures	F	P-value	F	P-value
MSE	0.725	0.654	2.840	0.081
Accuracy	1.228	0.344	0.135	0.875
Time	11.378	<=0.001	0.214	0.809
Epoch	2.985E4	<=0.001	0.000	1.000

4. CONCLUSION

In conclusion, for ANN classification model development for heart diseases, it is best to use Quasi-Newton training algorithms because of best speed and accuracy. Also, the number of neurons in hidden layer has no significant effect on the performance model.

REFERENCES

[1] Dangare CS, et al. "A data mining approach for prediction of heart disease using neural networks," *International Journal of Computer Engineering & Technology (IJCET),* vol. 3, pp. 30-40p, 2012.

[2] Spathis D, et al. "Diagnosing asthma and chronic obstructive pulmonary disease with machine learning," *Health Informatics Journal,* vol. 0, pp. 146.

[3] Khan IY, et al. "Importance of Artificial Neural Network in Medical Diagnosis disease like acute nephritis disease and heart disease," *International Journal of Engineering Science and Innovative Technology (IJESIT),* vol. 2, pp. 210-7, 2013.

[4] Kröse B, et al. "An introduction to neural networks," vol. pp. 1993.

[5] Al-Bahrani R, et al. "Survivability prediction of colon cancer patients using neural networks," *Health Informatics Journal,* vol. 0, pp. 146.

[6] Selma B, et al. "Neural network navigation technique for unmanned vehicle," *Trends in Applied Sciences Research,* vol. 9, pp. 246, 2014.

[7] Hashim MN, et al. "A Comparison Study of Learning Algorithms for Estimating Fault Location," *Indonesian Journal of Electrical Engineering and Computer Science,* vol. 6, pp. 2017.

[8] Yadav N, et al. An Introduction to Neural Network Methods for Differential Equations: *Springer*; 2015.

[9] Sharma B. "Comparison of neural network training functions for Hematoma classification in brain CT images," *IOSR Journals (IOSR Journal of Computer Engineering),* vol. 1, pp. 31-5, 2014.

[10] Aggarwal K, et al. "Evaluation of various training algorithms in a neural network model for software engineering applications," *ACM SIGSOFT Software Engineering Notes,* vol. 30, pp. 1-4, 2005.

[11] Zhou L, et al. "Training algorithm performance for image classification by neural networks," *Photogrammetric Engineering & Remote Sensing,* vol. 76, pp. 945-51, 2010.

[12] Al-Hasanat A, et al. "Experimental Invesitigation of Training Algorithms used in Backpropagation Artificial Neural Networks to Apply Curve Fitting."

[13] Soni J, et al. "Predictive data mining for medical diagnosis: An overview of heart disease prediction," *International Journal of Computer Applications,* vol. 17, pp. 43-8, 2011.

[14] Santulli G. "Epidemiology of cardiovascular disease in the 21st century: updated numbers and updated facts," *JCvD,* vol. 1, pp. 1-2, 2013.

[15] Abdar M, et al. "Comparing Performance of Data Mining Algorithms in Prediction Heart Diseases," *International Journal of Electrical and Computer Engineering (IJECE),* vol. 5, pp. 1569-76, 2015.

[16] Das R, et al. "Effective diagnosis of heart disease through neural networks ensembles," *Expert systems with applications,* vol. 36, pp. 7675-80, 2009.

[17] Aha DW. UCI Machine Learning Repository. 2016.

[18] Likas A, et al. "Training the random neural network using quasi-Newton methods," *European Journal of Operational Research,* vol. 126, pp. 331-9, 2000.

[19] Beale MH, et al. "Neural network toolbox 7," *User's Guide, MathWorks,* vol. pp. 2010.

[20] Wilson DR, et al. "The general inefficiency of batch training for gradient descent learning," *Neural Networks,* vol. 16, pp. 1429-51, 2003.

[21] Qian N. "On the momentum term in gradient descent learning algorithms," *Neural networks,* vol. 12, pp. 145-51, 1999.

[22] Bishop CM. Neural networks for pattern recognition: Oxford university press; 1995.

[23] Riedmiller M, et al., "*A direct adaptive method for faster backpropagation learning: The RPROP algorithm*," in 1993. Neural Networks, 1993, IEEE International Conference on, 1993, pp. 586-91.

[24] Birgin EG, et al. "A spectral conjugate gradient method for unconstrained optimization," *Applied Mathematics and optimization,* vol. 43, pp. 117-28, 2001.

[25] Dai Y-H, et al. "New conjugacy conditions and related nonlinear conjugate gradient methods," *Applied Mathematics and Optimization,* vol. 43, pp. 87-101, 2001.

[26] Fletcher R, et al. "Function minimization by conjugate gradients," *The computer journal,* vol. 7, pp. 149-54, 1964.

[27] Masood S, et al., editors. *Analysis of weight initialization routines for conjugate gradient training algorithm with Fletcher-Reeves updates*. Computing, Communication and Automation (ICCCA), 2016 International Conference on; 2016: IEEE.

[28] Al-Bayati A, et al. "Conjugate Gradient Back-propagation with Modified Polack–Rebier updates for training feed forward neural network," *Iraqi J Statis Sci,* vol. 20, pp. 164-73, 2011.

[29] Saini L, et al. "*Artificial neural network based peak load forecasting using Levenberg–Marquardt and quasi-Newton methods*," IEE Proceedings-Generation, Transmission and Distribution, vol. 149, pp. 578-84, 2002.

[30] Hagan MT, et al. "Training feedforward networks with the Marquardt algorithm," *IEEE transactions on Neural Networks,* vol. 5, pp. 989-93, 1994.

[31] Ilonen J, et al. "Differential evolution training algorithm for feed-forward neural networks," *Neural Processing Letters,* vol. 17, pp. 93-105, 2003.

[32] Hager WW, et al. "A survey of nonlinear conjugate gradient methods," *Pacific journal of Optimization,* vol. 2, pp. 35-58, 2006.

[33] Fletcher R. Conjugate gradient methods for indefinite systems. Numerical analysis: *Springer*; 1976. p. 73-89.