A time simulated annealing-back propagation algorithm and its application in disease prediction

Article *in* Modern Physics Letters B · August 2018

DOI: 10.1142/S0217984918503037

Fang Hu
Hubei University of Chinese Medicine
18 PUBLICATIONS **90** CITATIONS
SEE PROFILE

Yanhui Zhu
University of West Florida
5 PUBLICATIONS **6** CITATIONS
SEE PROFILE

Jia Liu
University of West Florida
4 PUBLICATIONS **76** CITATIONS
SEE PROFILE

Yalin Jia
4 PUBLICATIONS **4** CITATIONS
SEE PROFILE

Modern Physics Letters B
Vol. 32, No. 25 (2018) 1850303 (12 pages)
© World Scientific Publishing Company
DOI: 10.1142/S0217984918503037

World Scientific
www.worldscientific.com

A time simulated annealing-back propagation algorithm and its application in disease prediction

Fang Hu[*,†,‡], Mingzhu Wang[*], Yanhui Zhu[*], Jia Liu[†] and Yalin Jia[*]

[*] *College of Information Engineering,*
Hubei University of Chinese Medicine, Wuhan 430065, China
[†] *Department of Mathematics and Statistics,*
University of West Florida, Pensacola 32514, USA
[‡] *naomifang@mails.ccnu.edu.cn*

Received 1 June 2018
Revised 8 July 2018
Accepted 16 July 2018
Published 16 August 2018

In this paper, based on the Back Propagation (BP) neural network algorithm, we introduce the idea of the Simulated Annealing (SA), and then propose a new neural network algorithm: Time Simulated Annealing-Back Propagation (TSA-BP) algorithm. The proposed algorithm can improve the convergence rate and numerical stability. By using this proposed algorithm, the learning rates and initial weights in the BP neural network could be easily adjusted. We show that the TSA-BP algorithm could reduce the errors caused by human-made factors. Several numerical experiments have been tested by using different disease data. Furthermore, we compared the TSA-BP algorithm to the other existing, well-known algorithms. Numerical results show higher accuracy and efficiency of the TSA-BP algorithm.

Keywords: Neural network; TSA-BP algorithm; simulation experiment; contrastive analysis; disease prediction.

1. Introduction

Recently, with the rapid development of different advanced technologies including artificial intelligence,[1] deep learning,[2] complex networks,[3] neural network (NN),[4] etc., the potential relationships among things are being increasingly explored. The NN, an effective approach of prediction, is widely used in various research fields such as computer science,[5] medicine,[6,7] physics,[8,9] and has made many significant breakthroughs.[10,11] How to design high-powered NN algorithms to improve the accuracy and efficiency of prediction has drawn great attention of many researchers.

[‡] Corresponding author.

The performance of one NN algorithm is evaluated by the accuracy rate of prediction and training time. Despite various kinds of NN approaches, achieving both high accuracy of prediction and less training time is still challenging. Unfortunately, most algorithms which claim to have high accuracy usually require high training time. Enhancing the performance of the NN algorithm is the main goal in different research fields. In the field of medical applications, multiple approaches/algorithms, such as artificial intelligence and machine learning techniques, have been used for disease prediction.[12–14] In Refs. 6, 7, 15 and 16 NN algorithms have been applied to coronary heart disease or breast cancer. Research data shows that NN algorithms provide both accurate and efficient methods to solve disease prediction problems.[17] Naushad *et al.*[15] proposed an Artificial Neural Network (ANN) model to predict breast cancer risk. The ANN model demonstrated 94.2% variability in breast cancer prediction. Naushad *et al.* also introduced another ANN model to predict coronary artery disease. The accuracy of the disease prediction could be improved to 81%. Ha *et al.*[18] presented a language model-like method. Their model predicts high-risk prognoses from the diagnosis histories of patients using deep recurrent neural networks (RNNs). This new method could improve the accuracy of disease prediction to 78.3%.

In 1986, Rumelhart *et al.*[19] presented the idea of using the Back Propagation (BP) to solve the NN problems efficiently. Now, the BP algorithm can be applied to a wide range of problems because it can solve complicated random nonlinear mapping issues.[20] Xu *et al.*[21] proposed a multi-parameter fusion model by BP NN to estimate the blood pressure and acquired a high accuracy result. Zhang *et al.*[22] diagnosed congenital heart disease and colorectal cancer based on BP NN and obtained 75% diagnostic accuracy. However, the slow convergence rate and poor numerical stability of traditional BP algorithm are always disturbing, and the training time will become lengthy as the sample size increases.[20] We aim to develop some prediction algorithms that have better performance than most existing state-of-the-art algorithms in different characteristics.

In this paper, we will introduce the idea of the simulated annealing algorithm to the BP algorithm, and then propose a Time Simulated Annealing-Back Propagation (TSA-BP) algorithm based on a combination of the BP algorithm and the simulated annealing algorithm, named as TSA-BP algorithm hereafter. We will explain the steps of the algorithm, and then explore the complexity of TSA-BP. We also investigate the performance of the TSA-BP algorithm through using several numerical experiments, in which results are promising and encouraging. We further test the algorithm with a variety of datasets such as breast cancer data, coronary heart disease data and other adult disease data. The results, compared to the other current high-performance algorithms, achieve high accuracy rates with less training time in most disease predictions.

2. Realization of the TSA-BP Algorithm

2.1. *Algorithm idea*

In this section, we will briefly introduce the TSA-BP algorithm. We normalized the testing data by using the min–max standardized algorithm to ensure the range of data is between 0 and 1. By normalizing, we avoided the limitations of slow convergence and the high amount of training time for the inconsistency of data unit. Then we selected the BP algorithm to optimize the NN model. Based on the values of inversion parameters and the characteristics of the NN structure, this algorithm confirms and adjusts the number of neuron nodes in the processing of inversion in the NN. After comparing several activation functions, we could respectively process the hidden layer and the output layer by using different activation functions. For example, this algorithm uses the hyperbolic tangent S function for the hidden layer and the general linear function for the output layer. By using this method, this algorithm optimizes the NN by layers, speeds up the convergence rate and ensures the accuracy of the results. Meanwhile, we introduce the cross-entropy cost function such as the energy function (which is the objective function). The cross-entropy cost function could speed up the convergence of the algorithm.

We further introduce the simulated annealing algorithm to implement the weight adjustment and the inversion phase after optimizing the structure of the NN model by using the BP algorithm. In general, the traditional NN algorithms usually use the gradient descent method (GDM). However, the GDM often leads to a local optimal solution instead of the global one. Under the condition of suboptimal solutions, we used the SA algorithm. This combination of the SA and BP algorithms enabled us to receive the new values based on a specific probability. Therefore, we were able to produce the optimal solution with the minimum error. The TSA-BP method can avoid falling into the local minimum. At the same time, we did not need to find the inverse of the large matrix which is an expensive operation. Therefore, the proposed algorithm has advantages in terms of simplifying codes, controlling objective functions flexibly and acquiring the global optimal solutions.

2.2. *Algorithm steps*

The steps of the new NN algorithm, TSA-BP, are as follows.

The weight matrix of the NN is set as w; the minimum error accuracy is e; the initialized parameters are set as: the current iteration time $m = 1$, the counter LocalCount $= 0$, and the learning rate $\eta = 0.1$; $E(m)$ is the sum of squared errors of the mth iteration in the network; $\mathrm{SA}(w)$ is the SA algorithm.

Step 1. Initialize the weight matrix; avoid the limitations of slow convergence and high training time by using the min–max standardized algorithm

$$N = \frac{(X_{i(\max)} - X_{i(\min)}) * (X_{ij} - X_{j(\min)})}{(X_{j(\max)} - X_{j(\min)})} + X_{i(\min)}, \tag{1}$$

where X is the training dataset, N represents the normalized result matrix, $X_{i(\max)}$ and $X_{i(\min)}$, respectively, denote the maximum and minimum of the ith property in X.

Step 2. Set the input as $x_1, x_2, \ldots$, the weight values as $w_1, w_2, \ldots$, the bias as b, and the output of neuron as

$$a = \sigma(z) = \frac{1}{1 + e^{-z}}, \tag{2}$$

where $z = \sum_j w_j x_j + b$.

Step 3. Define the cross-entropy cost function C of a neuron as

$$C = -\frac{1}{n} \sum_x [y \ln a + (1 - y) \ln(1 - a)], \tag{3}$$

where n is the total number of samples, y is the expected output and a is the predicted output of a neuron.

Step 4. Define the hyperbolic tangent S function as the activation function

$$f(z) = \frac{1}{1 + e^{-z}} \equiv \frac{1}{1 + \exp(-\sum_i w_i x_i - b)}. \tag{4}$$

Step 5. Calculate $E(m)$ in the training network based on BP algorithm. If $E(m) < e$, the condition of convergence is satisfied, then proceed to the Step 11.

Step 6. Compare the equation $E(m) - E(m-1)$; adjust the learning rate η according to the activation function $f(z)$.

Step 7. If $|E(m) - E(m - 1)| < p$, where p is a selected threshold, increase the parameter LocalCount = LocalCount + 1; otherwise, set LocalCount = 0.

Step 8. If LocalCount $> C$, the NN has fallen into the local minimum, then record the error at this time. Make the local minimum LocalMin $= E(m)$ by using the SA algorithm

$$\mathrm{SA}(i+1) = \begin{cases} \mathrm{SA}(i), & \text{if } (J(\mathrm{SA}(i+1)) \geq J(\mathrm{SA}(i))), \\ \mathrm{SA}(i), & \text{if } (e^{J(Y(i+1)) - J(Y(i))} > \mathrm{rand}(1)), \\ r * \mathrm{SA}(i), & \text{others}, \end{cases} \tag{5}$$

where $r = \mathrm{rand}(1)$, $\mathrm{rand}(n)$ is a $n \times n$ random matrix; the value range of every element is $(0, 1)$.

Step 9. If $E(m) < e$, the condition of convergence is satisfied again, then proceed to Step 11.

Step 10. If $E(m) <$ LocalMin, this algorithm jumps out of the local minimum, then begin another iteration of BP training. Otherwise, return to Step 7.

Step 11. Stop the network training.

The flowchart of the TSA-BP algorithm is in Fig. 1.

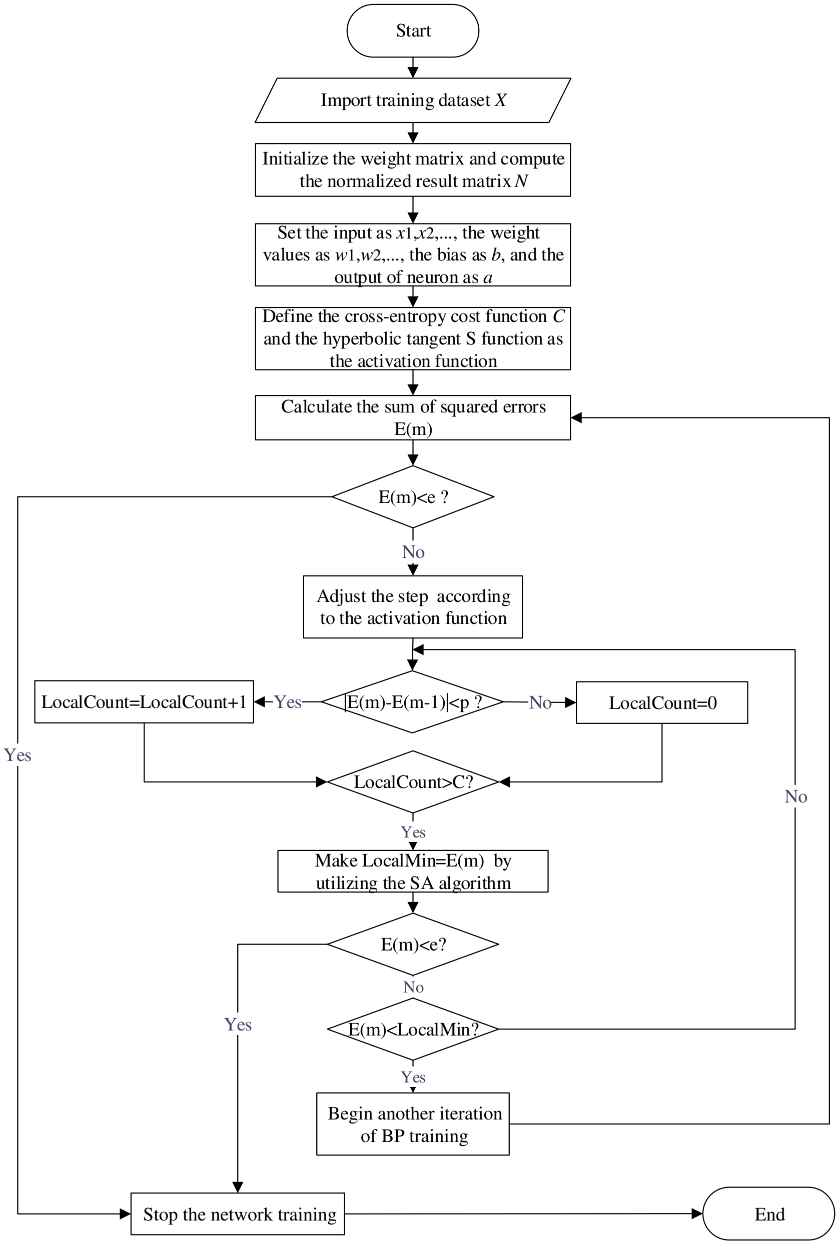
Start
Import training dataset X
Initialize the weight matrix and compute the normalized result matrix N
Set the input as x1,x2,..., the weight values as w1,w2,..., the bias as b, and the output of neuron as a
Define the cross-entropy cost function C and the hyperbolic tangent S function as the activation function
Calculate the sum of squared errors E(m)
E(m)<e ?
No
Yes
Adjust the step according to the activation function
LocalCount=LocalCount+1
Yes
\|E(m)-E(m-1)\|<p ?
No
LocalCount=0
LocalCount>C?
No
Yes
Make LocalMin=E(m) by utilizing the SA algorithm
E(m)<e?
No
Yes
E(m)<LocalMin?
Yes
Begin another iteration of BP training
Stop the network training
End

Fig. 1. The flowchart of the TSA-BP algorithm.

2.3. *Computational complexity*

Based on the steps of this algorithm, one of the most significant key factors of the BP algorithm is the capacity of the constructed network X_{m*n}, which will be close to $X_{n*m} \geq (n+1)H$ if there is only one output node and the computational complexity is approximately $O(e*[(n+1)H])$, where n is the number of samples, m is the number of the properties of samples, and H is the number of hidden nodes.

According to the SA algorithm, the time complexity of one internal loop is $O(n)$ and the external loop is $O(m)$. Thus, the total time complexity is $O(n*m)$. Similarly, the space complexity of one internal loop is $O(n)$ with consideration to the external loop. Therefore, the total space complexity is $O(n*n)$. The time complexity of TSA-BP is $O(n*m)$. Considering the space complexity, the internal complexity is $O(m)$ and the external is $O(n)$, the total space complexity of TSA-BP then is $O(n*n)$.

3. Simulations and Analysis

In this section, to verify the accuracy and efficiency of the TSA-BP algorithm, we test it by using three types of disease networks: breast cancer network,[23] coronary heart disease network,[24] and adult disease network.[25] We also compared the TSA-BP algorithm to the other popular algorithms.

3.1. *Datasets testing and numerical experiments*

3.1.1. *Test on the breast cancer network*

One of the most common cancers in females is breast cancer. This cancer threatens one-eighth of women around the world.[15] The five-year relative survival rate for women with Stage 0 or Stage I breast cancer was close to 100% between 2007 and 2013 according to the National Cancer Institute's SEER database cancer.[26] In 1988, Michalski *et al.*[23] collected the dataset of breast cancer from the clinical detection results in the International Clinical Sciences Support Center, University of Wisconsin, USA. Through years of clinical verifications, researchers selected only 10 properties: radius, texture, perimeter, area, smoothness, compactness, concavity, concave points, symmetry, and fractal dimension.[27,28] These properties are closely related to the breast cancer from the dataset with 32 properties. This dataset contains 459,347 records.

We tested the TSA-BP algorithm in the breast cancer network, which has 300 cases of training data and 400 cases of testing data. The simulation results (Fig. 2) represent that the breast disease is divided into two categories: *benign tumor* and *malignant tumor*. The *blue* and *red circles* denote the expected outputs and predicted outputs, respectively. In addition, the result of the *benign tumor* and *malignant tumor* are valued as "−1" and "1", indicating the patient without and with the disease, respectively. The training time is an intuitive reflection of the algorithm efficiency, and the number of the training time of the TSA-BP algorithm for this dataset is 92. The accuracy is the most significant index to verify the results of the algorithm. We normalized the predicted result as "1" or "−1" based on whether it is greater than 0 or not. Through comparing the predicted results to the expected results with the formats "1" or "−1", we can acquire the accuracy (0.9361) in this dataset.

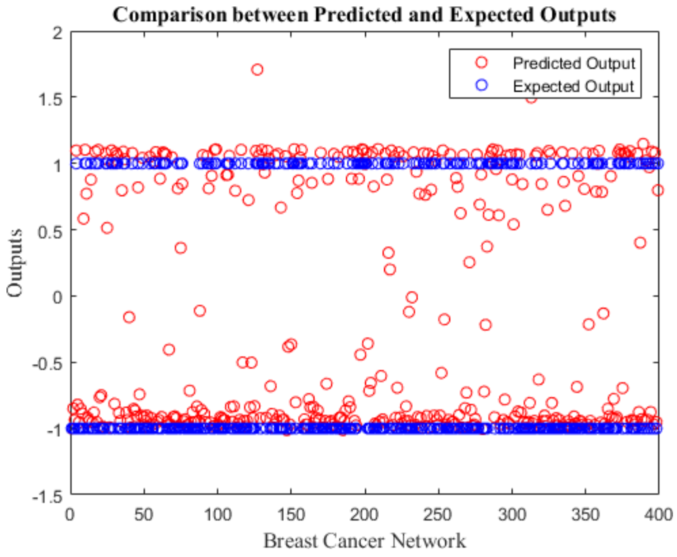

Fig. 2. (Color online) The predicted and expected outputs based on TSA-BP in breast cancer network.

3.1.2. *Test on the coronary heart disease network*

In 1988, Detrano *et al.*[24] collected the dataset of coronary heart disease from the clinical detection results in Institute of Coronary Heart Disease, Hungary. This dataset has 11 properties including age, gender, chest pain type, resting blood pressure, serum cholesterol, fasting blood-glucose, resting electrocardiogram, max heart rate, exercise-induced angina, the colored perspective of the blood vessel, and thalassemia level. This dataset contains 270 records.

We tested the TSA-BP algorithm in the coronary heart disease network with 200 cases of training data and 70 cases of testing data. According to Fig. 3, the predicted results denote that the coronary disease is divided into two categories: the *sickness* and the *non-sickness*. The *blue* and *red circles* present the expected outputs and predicted outputs, respectively. The training time of the TSA-BP algorithm for this dataset is 16; the accuracy reaches up to 0.6615.

3.1.3. *Test on the adult disease network*

Some adult diseases including diabetes, high blood pressure, and hyperlipidemia can be controlled well by the patients and the doctors if there is early detection and prevention.[29–31] In 1996, Kohavi,[25] collected the dataset of adult disease from the result of the population census in 1994. This dataset has 14 properties including age, gender, weight, job specification, educational status, marital status, social relationship, race, capital surplus, capital deficit, occupation, exercise time per week, country, and income. We select 5000 records from this dataset.

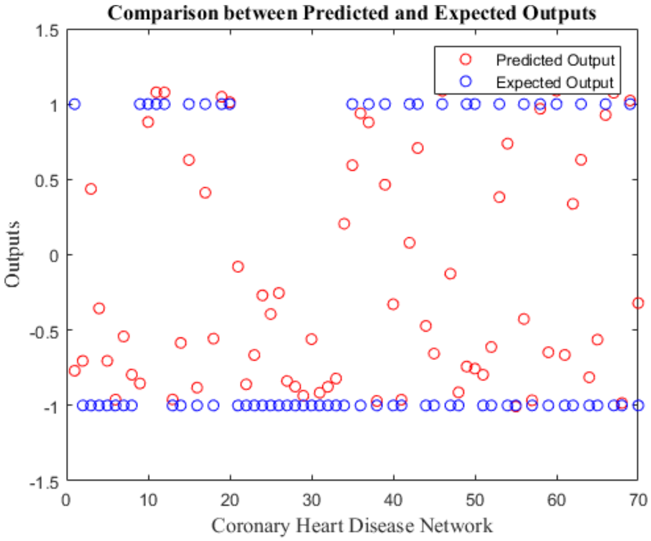

Fig. 3. (Color online) The predicted and expected outputs based on TSA-BP in coronary heart disease network.

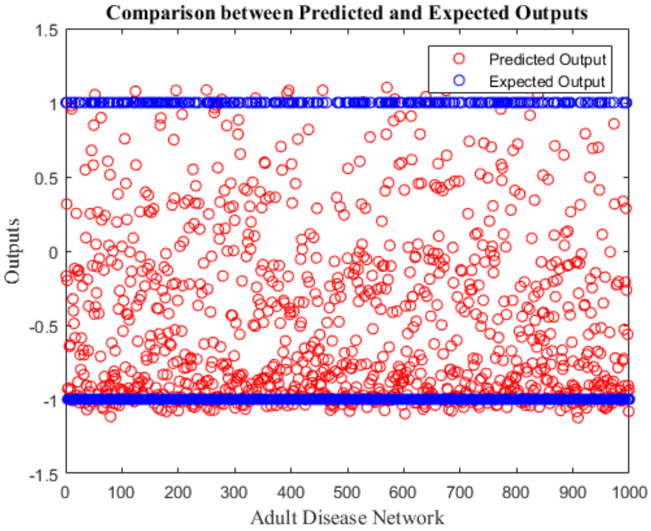

Fig. 4. (Color online) The predicted and expected outputs based on TSA-BP in adult disease network.

We tested the TSA-BP algorithm in the adult disease network, which has 4,000 cases of training data and 1,000 cases of testing data. The predicted results (Fig. 4) denote that the adult disease is divided into two categories: *sickness* and *non-sickness*. The *blue* and *red circles* indicate the expected outputs and predicted outputs, respectively. The training time of the TSA-BP algorithm for this dataset is 71, and the accuracy is up to 0.5717.

3.2. *Simulation results and contrastive analysis*

3.2.1. *Contrastive analysis of accuracy*

The accuracy of the predicted results is one of the most significant indices to verify the performance of NN algorithms. Therefore, we tested the proposed NN algorithm in three disease networks including breast cancer network, coronary heart disease network, and the adult disease network. Furthermore, we compared our algorithm with seven representative neural algorithms. The most popular algorithms in the NN are GDM,[32] Self-Adaptive Gradient Descent Method (SA-GDM),[33,34] BP Elastic Algorithm (RPROP),[35] Scaled conjugate gradient (SCG),[36] Broyden–Fletcher–Goldfarb–Shanno (BFGS),[37] One-step Secant Algorithm (OSS),[38] and Levenberg-Marquardt (LM).[39] The simulation results calculated by these eight algorithms in three disease networks are in Table 1.

Accuracy is an essential performance parameter to test the NN algorithms. The higher the value of the accuracy is, the better the property of the algorithm is. According to Table 1 and Fig. 5, we can see that the TSA-BP algorithm and the

Table 1. The accuracy comparison in three disease networks.

Disease networks	Training data	Testing data	Accuracy							
			GDM	SA-GDM	RPROP	SCG	BFGS	OSS	LM	TSA-BP
Coronary heart	300	400	0.479	0.4553	0.4879	0.5813	0.5466	0.4655	0.4826	0.6615
Breast cancer	200	70	0.3520	0.4047	0.4702	0.3700	0.5691	0.711	0.5701	0.9361
Adult disease	4000	1000	0.4214	0.4049	0.5253	0.5095	0.5598	0.4262	0.5605	0.5717

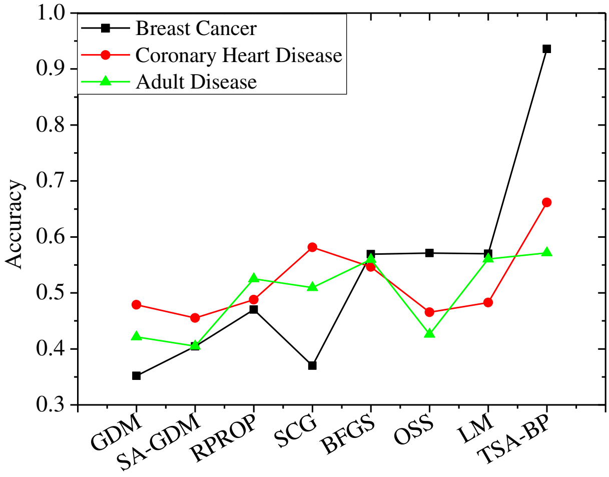

Fig. 5. (Color online) The accuracy comparison chart of GDM, SA-GDM, RPROP, SCG, BFGS, OSS, LM, and TSA-BP in three disease networks.

other seven representative algorithms GDM, SA-GDM, RPROP, SCG, BFGS, OSS, and LM perform well on these three disease networks. The TSA-BP algorithm has better performance (higher values of accuracy) than the others.

3.2.2. *Contrastive analysis of training time*

We calculated the training time, the other index to test the performance of NN algorithms, by using the proposed algorithm TSA-BP and the other seven different neural algorithms. The training time calculated by these eight algorithms in three disease networks are in Table 2.

Another critical performance parameter is the training time of the NN algorithms. The less the amount of the training time is, the better the convergence is. In this experiment, we set the maximum of training time as 1,000. Based on Table 2 and Fig. 6, these eight neural algorithms can use less training time to acquire the results in three disease networks. However, the results of training time calculated by the TSA-BP algorithm proved that the proposed algorithm is more efficient than the other neural algorithms GDM and OSS.

Table 2. The training time comparison in three disease networks.

Disease networks	Training data	Testing data	Training time							
			GDM	SA-GDM	RPROP	SCG	BFGS	OSS	LM	TSA-BP
Coronary heart	300	400	397	89	14	11	13	8	11	16
Breast cancer	200	70	1000	96	60	19	54	106	13	92
Adult disease	4000	1000	1000	103	60	66	35	79	17	71

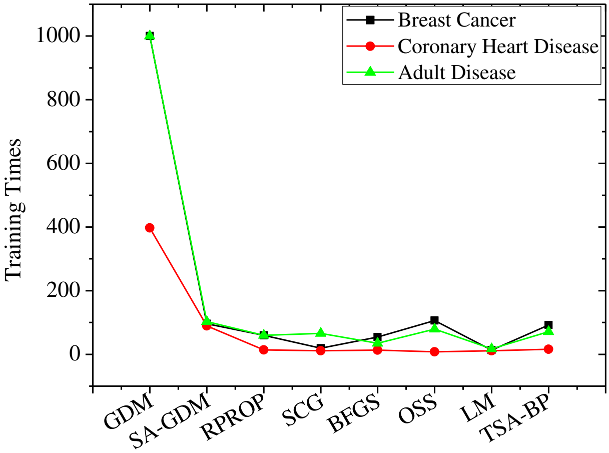

Fig. 6. (Color online) The training time comparison chart of GDM, SA-GDM, RPROP, SCG, BFGS, OSS, LM, and TSA-BP in three disease networks.

4. Conclusions

In this paper, based on the BP algorithm, we designed a new NN algorithm TSA-BP. We optimized this algorithm by using the SA algorithm to improve the weight adjustment and the inversion phase. The proposed TSA-BP algorithm enables better performance regarding accuracy rate and training time. We have tested the TSA-BP algorithm in multiple datasets to observe the accuracy and the training time. We used the disease datasets including breast cancer, coronary heart disease, and the other adult disease to verify the accuracy and the efficiency of the proposed algorithm. All three experiments show high accuracy rates and efficiency of the TSA-BP algorithm. Furthermore, we compared the proposed algorithm to current popular algorithms such as GDM, SA-GDM, RPROP, SCG, BFGS, OSS, and LM. The experimental results prove that the TSA-BP algorithm could produce fewer errors than most available algorithms, and it also reduces the amount of training time according to the GDM and OSS algorithms.

In conclusion, NN becomes more critical for the prediction in different research fields, especially in the detection of various diseases in the early stages. Getting accurate and efficient algorithms is the primary research goal in the NN area. We proposed an efficient algorithm with high accuracy in this paper. The experimental results are promising and robust.

Acknowledgment

We acknowledge the funding support by the Natural Science Foundation of Hubei Province (2018CFB259).

References

1. J. Weng *et al.*, *Science* **291** (2001) 599.
2. Y. Lecun *et al.*, *Nature* **521** (2015) 436.
3. F. Hu *et al.*, *Phys. Lett. A* **381** (2017) 3604.
4. L. A. Campfield *et al.*, *Science* **269** (1995) 546.
5. L. G. Chen *et al.*, *Mod. Phys. Lett. B* **31** (2017) 1750262.
6. J. K. Kim and S. Kang, *J. Healthc. Eng.* **2017** (2017) 1.
7. O. Y. Atkov *et al.*, *J. Cardiol.* **59** (2012) 190.
8. G. Long and G. Xu, *SPE J.* **22** (2017) 645.
9. B. Xiao *et al.*, *Fractals* **26** (2018) 1840015.
10. Z. Che *et al.*, *Sci. Rep.-UK* **8** (2018) 6085.
11. E. Cardelli *et al.*, *J. Appl. Phys.* **117** (2015) 17D129.
12. S. Langley-Evans, *J. Hum. Nutr. Diet.* **28** (2015) 1.
13. C. Barbieri *et al.*, *Comput. Biol. Med.* **61** (2015) 56.
14. B. Robson and S. Boray, *Comput. Biol. Med.* **66** (2015) 82.
15. S. M. Naushad *et al.*, *Gene* **580** (2016) 159.
16. C. Milani and N. M. Jadavji, *J. Young Invest.* **33** (2017) 122.
17. R. Wu *et al.*, *J. Health. Inf. Manag.* **16** (2002) 50.
18. J.-W. Ha *et al.*, in *IEEE Int. Conf. on Big Data and Smart Computing* (*BigComp*) (2017), p. 394.

19. D. E. Rumelhart *et al.*, *Nature* **323** (1986) 533.
20. J. Cao *et al.*, *PLoS One* **11** (2016) e0157551.
21. Z. Xu *et al.*, *Comput. Ind.* **89** (2017) 50.
22. B. Zhang *et al.*, *Genet. Mol. Res.* **15** (2016) 15028643.
23. R. S. Michalski *et al.*, The multipurpose incremental learning system AQ15 and its testing application to three medical domains, in *AAAI National Conf. on Artificial Intelligence*, Philadelphia, Pennsylvania (1986), p. 1041–1045.
24. R. Detrano *et al.*, *Am. J Cardiol.* **64** (1989) 304.
25. R. Kohavi, Scaling up the accuracy of Naive–Bayes classifiers: A decision-tree hybrid, in *Proc. Second Int. Conf. on Knowledge Discovery and Data Mining* (1996), p. 202–207.
26. A. M. N. N. Howlader *et al.*, *National Cancer Institute* (2017), https://seer.cancer.gov/csr/1975_2014/2017.
27. B. Xiao *et al.*, *Fractals* **25** (2017) 1750030.
28. B. Xiao *et al.*, *J. Nanosci. Nanotechno.* **17** (2017) 6811.
29. E. J. Benjamin *et al.*, *Circulation* **135** (2017) e146.
30. J.-C. Li *et al.*, *Physica A* **495** (2018) 1.
31. R. Narain *et al.*, *Patient Prefer. Adher.* **10** (2016) 1259.
32. P. Tseng and S. Yun, *Math. Program.* **117** (2009) 387.
33. H. Abbasi *et al.*, Type-2 fuzzy wavelet neural network controller design based on an adaptive gradient descent method for nonlinear dynamic systems, in *Applied Methods and Techniques for Mechatronic Systems* (2014), p. 229–247.
34. C. Burges *et al.*, Learning to rank using gradient descent, in *Proc. 22nd Int. Conf. on Machine Learning* (2005), pp. 89–96.
35. M. Riedmiller and H. Braun, RPROP — A fast adaptive learning algorithm, in *Int. Symp. on Computer and Information Science* (1992), pp. 279–286.
36. M. F. Møller, *Neural Networks* **6** (1993) 525.
37. D. C. Liu and J. Nocedal, *Math. Program.* **45** (1989) 503.
38. A. K. A. Gangwar and D. K. Sharma, *Int. J. Adv. Res. Sci. Eng.* **4** (2015) 239.
39. J. J. Moré, *Lect. Notes Math.* **630** (1978) 105.