A Brief Analysis of the Adaptive Algorithm and Optimization of the BP Neural Network

Yichen Dai

School of Artificial Intelligence, China University of Mining & Technology,Beijing,

Beijing, 100083, China

2110402213@student.cumtb.edu.cn

Abstract. This article first introduces the development of adaptive algorithms, especially the application of deep learning and backpropagation algorithms in adaptive algorithms. Then the backpropagation (BP) neural network including its basic structure and mathematical modeling is analyzed. Special emphasis is placed on the advantages of BP neural network in adaptive algorithms, such as self-regulation ability and ability to handle complex problems. This paper compares the advantages of various algorithms through the performance indicators of various algorithms in predicting early neurological deterioration (END) in patients with acute cerebral infarction (ACI) after intravenous thrombolysis. In addition, the paper discusses the problems that BP neural networks may face in practical applications and optimizes the model by introducing methods such as zero-norm regularization, which can reduce the computational burden in training and inference and can effectively avoid overfitting questions.

Keywords: BP Neural Network, adaptive algorithms, machine learning, zero-norm regularization.

1. Introduction

The concept of adaptive algorithms dates to the middle of the last century, when scientists began exploring systems that could adjust and improve themselves. During this period, early neural network models and simple adaptive control systems were born. From the 1990s to the early 21st century, the development of machine learning methods such as support vector machines (SVM), decision trees, and random forests provided new tools for adaptive algorithms, making their applications more widespread and effective.

Since 2010, the rise of deep learning has greatly promoted the development of adaptive algorithms. Deep neural networks, especially convolutional neural networks (CNN) and recurrent neural networks (RNN) have demonstrated unprecedented development potential in image recognition, language processing, and other aspects.

In recent years, scientists have been exploring more combinations of different types of adaptive algorithms and models to deal with more complex problems. For example, adaptive algorithms are widely used in autonomous driving technology - enabling vehicles to handle complex environments on their own and make correct decisions quickly. There is also the medical and health field - adaptive algorithms are widely used in medical image analysis, disease prediction, personalized treatment, etc., greatly reducing the misdiagnosis rate [1].

The adaptive algorithm has the following advantages. The first is elasticity and adaptability - its parameters can be automatically adjusted according to changes in input data, which makes the algorithm more suitable for application environments where data characteristics are constantly changing [2]. At the same time, adaptive algorithms have good fault tolerance - they can learn from errors and adjust themselves, and they can handle imperfect input data.

Adaptive algorithms have important practical significance in all aspects of modern society [3]. This article focuses on related methods suitable for adaptive algorithms and compares these methods. In particular, the BP neural network was studied, and its performance indicators were optimized to improve the generalization ability and reduce the error through the backpropagation algorithm.

2. The analysis of the BP Neural Network

2.1. Basic Structure

The basic structure of the BP neural network can be expressed as $n\times1\times m$, where n, l, and m are the neurons of the input layer, hidden layer, and output layer respectively. Yuan quantity. Denote the neurons of the input layer, hidden layer, and output layer as $\{I_1, I_2, ..., I_n\}$, $\{H_1, H_2, ..., H_n\}$ and $\{O_1, O_2, ..., O_n\}$ respectively as shown in Figure 1.

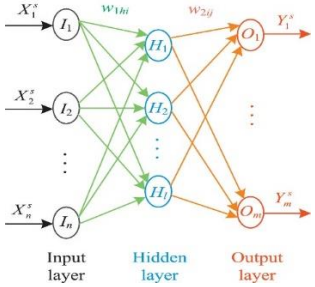

Fig. 1 Basic structure of a 3-layer BP neural network (Original)

In such a network structure, the input layer is responsible for receiving external input signals and passing these signals to the hidden layer. The hidden layer is the internal processing layer of the network, which processes the signal from the input layer and then passes it to the output layer. The output layer is responsible for producing the final output signals, which represent the network's response or processing results to the input data.

2.2. Mathematical Modeling

The following is a mathematical model of a typical three-layer feedforward neural network.

$$\begin{aligned} H_{Ii}^{s} &= \sum_{h=1}^{n} w_{1hi} X_{h}^{s}, i = 1,2,...,l \\ H_{Oi}^{s} &= g\left(H_{Ii}^{s}\right) = g\left(\sum_{h=1}^{n} w_{1hi} X_{h}^{s}\right), i = 1,2,...,l \end{aligned} \tag{1}$$

$$\begin{aligned} Y_{Ij}^{s} &= \sum_{j=1}^{m} w_{2ij} H_{Oi}^{s},\ j = 1,2,...,m \\ Y_{Ij}^{s} &= f\left(Y_{Ij}^{s}\right) = f\left(\sum_{j=1}^{m} w_{2ij} H_{Oi}^{s}\right) = f\left(\sum_{j=1}^{m} w_{2ij} \cdot g\left(\sum_{h=1}^{n} w_{1hi} X_{h}^{s}\right)\right) \\ & j = 1,2,...,m \end{aligned} \tag{2}$$

X^{s} is the input vector where s represents the s-th sample and N is the maximum number of samples. $Y^{s} = \left(Y_1^{s}, Y_2^{s}, ...Y_m^{s}\right)$ is the output vector which corresponds to the output vector of the output layer.

The connection weight between the input layer and the hidden layer is expressed as $\{w_{1hi}\}$;

The connection weight between the hidden layer and the output layer is expressed as $\{w_{2ji}\}$;

Input to hidden layer: $H_{Ii}^{s} = \sum_{h=1}^{n} w_{1hi} X_{h}^{s}$ where i is the index of the hidden layer neuron.

Output of hidden layer: $H_{Oi}^{s} = g\left(H_{Ii}^{s}\right)$;

Input to the output layer: $Y_{Ij}^{s} = \sum_{i=1}^{l} w_{2ji} H_{Oi}^{s}$ where j is the index of the output layer neuron.

Output of the output layer: $Y_{Oj}^{s} = f\left(Y_{Ij}^{s}\right)$.

The activation function $g(\cdot)$ and $f(\cdot)$ are usually assumed to be a bounded and infinitely differentiable function over the real number domain.

2.3. Exhaustive Analysis of Back Propagation Algorithm

2.3.1 Overview

The Backpropagation algorithm is a fundamental mechanism for training neural networks, essential for the development of deep learning [4]. It allows for efficient optimization of network weights by calculating the gradient of the loss function, enabling networks to learn from data and improve their performance on tasks such as image and speech recognition. Its universal applicability and the ability to facilitate automatic feature learning from raw data have made it an indispensable tool in the advancement of artificial intelligence [5].

The guiding principle behind the learning rules of a Backpropagation (BP) network is that changes to the network's weights and thresholds should be made in the direction of the negative gradient, which corresponds to the steepest descent of the function.

$$x_{k+1} = x_k - \eta_k g_k \tag{3}$$

The equation referenced specifies that x_k denotes the current matrix of weight and threshold values; g_k denotes the gradient of the current function; and η_k represents the learning rate.

2.3.2 Calculation of output nodes

The output of each node of the output layer z_l is obtained by applying the activation function f to the weighted sum of the inputs to that node. This weighted sum is expressed as net_1. The sum of the products of the outputs of the hidden layer nodes y_j and the corresponding weights v_{lj} minus the threshold θ_j.

Output of the node of the hide layer:

$$\begin{aligned} y_j &= f\left(\sum_i w_{ji} x_i - \theta_j\right) = f\left(net_j\right) \\ net_j &= \sum_i w_{ji} x_i - \theta_j \end{aligned} \tag{4}$$

Computational output of the output node:

$$\begin{aligned} z_l &= f\left(\sum_j v_{lj} y_j - \theta_l\right) = f\left(net_l\right) \\ net_l &= \sum_j w_{lj} y_j - \theta_l \end{aligned} \tag{5}$$

2.3.3 Error of output node

The error E of the output layer node is determined by calculating half the sum of the squares of the difference between the expected output t_l and the calculated output z_l.

Here is the mathematical form:

$$E = \frac{1}{2}\sum_l (t_l - z_l)^2 = \frac{1}{2}\sum_l \left(t_l - f\left(\sum_j v_{lj} y_j - \theta_j\right)\right)^2 \tag{6}$$

Expand the formula further:

$$E = \frac{1}{2}\sum_l \left(t_l - f\left(\sum_j v_{lj} f\left(\sum_i w_{ji} x_i - \theta_j\right) - \theta_j\right)\right)^2 \tag{7}$$

This method can be used to calculate the gradient of the network error for each weight very efficiently, especially for some relatively complex network training [6]. Another advantage is its

universality, which is suitable for many different neural network structures including deep neural networks.

2.4. Advantages Applied to Adaptive Algorithms

The first is that it can adjust itself. The backpropagation algorithm allows a neural network to automatically adjust its internal parameters based on changes in input data, thereby improving its adaptability to new data. And this algorithm can update its parameters in real time. Thanks to its gradient-based optimization approach, the network can be continuously updated to respond to environmental changes in real-time [7].

Secondly, it is very robust and can well solve the modeling of complex problems [8]. Because the backpropagation algorithm can help neural networks learn complex function mappings, it is very important for adaptive control systems and predictive models. Models trained through backpropagation through the network have better fault tolerance and can handle noise and incomplete input data.

In addition, this algorithm also has powerful nonlinear fitting capabilities. The backpropagation algorithm can transform linear relationships into nonlinear relationships through activation functions, allowing neural networks to learn complex nonlinear relationships, which is essential for many adaptive algorithms [9].

3. Performance Comparison with Other Related Algorithms

3.1. Build Predictive Models

With the advancement and development of adaptive algorithms, more and more methods are being applied to disease diagnosis, especially some diseases with complex causes and many interference items.

The following is a comparison of the energy efficiency of BP neural network, random forest and decision tree algorithms in predicting early neurological deterioration (END) in patients with acute cerebral infarction (ACI) after receiving intravenous thrombolysis [10].

3.1.1 BP Neural networks

The input layer includes a total of 10 nodes as shown in Figure 2 (plus an offset term B1): Transient Ischemic Attack (TIA), smoking, time from onset to intravenous thrombolysis, TOAST classification, hyperattenuating middle cerebral artery sign (HMCAS), systolic blood pressure before thrombolysis, fasting blood glucose before thrombolysis, white blood cell count before thrombolysis, before thrombolysis D-dimer (D-D) and the monocyte to high-density lipoprotein ratio (MHR) before thrombolysis [10].

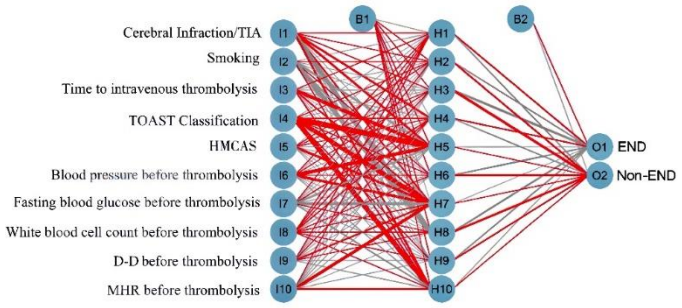

Fig. 2 BP neural network illustrating the risk of END in patients with ACI after receiving intravenous thrombolytic therapy [10].

3.1.2 Random forest

Establish a random forest training model with mtry=3. When n tree>100, the variation range of the model error rate begins to decrease and becomes stable as shown in Figure 3. By observing the

importance of input variables, the sequence of variables that affects the occurrence of END in ACI patients after intravenous thrombolysis can be obtained.

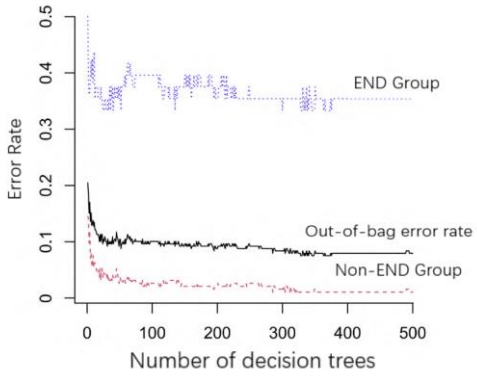

Fig. 3 Visual analysis of the relationship between model error rate and the number of trees in a random forest [10].

3.1.3 Decision tree algorithm

The idea of the decision tree algorithm (as shown in figure 4) for this problem is to construct a decision tree for END in ACI patients after intravenous thrombolysis, and screen out a total of 3 explanatory variables, namely D-D before thrombolysis, time from onset to intravenous thrombolysis, and white blood cell count before thrombolysis.

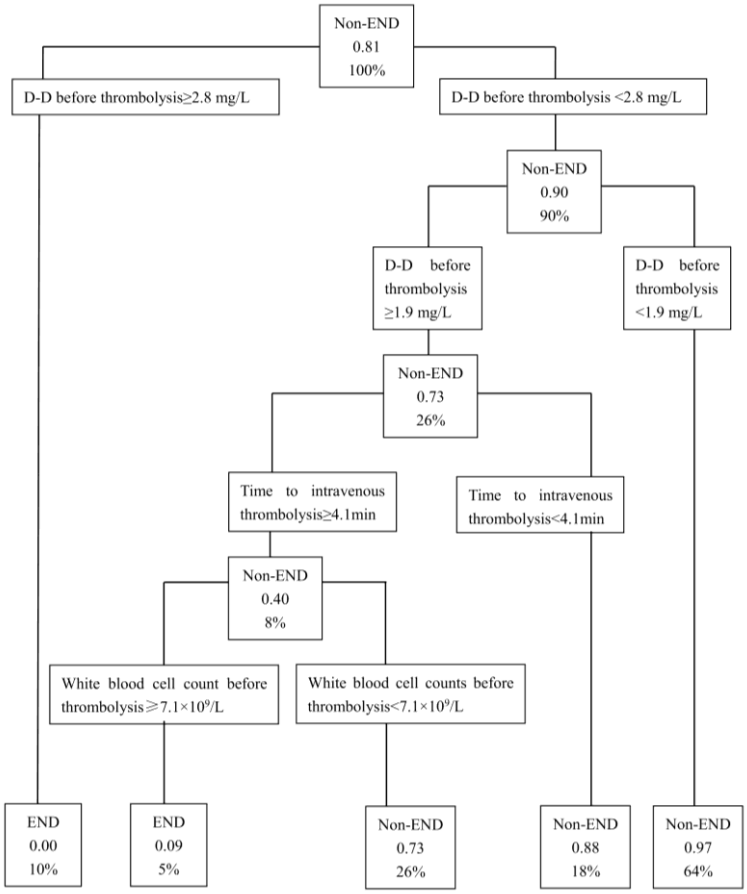

Fig. 4 Decision tree for END in patients with ACI after intravenous thrombolysis [10].

3.2. Research Result

After establishing the prediction model, the effectiveness of the above algorithms can be compared more intuitively through their performance on the training set and test set. The following evaluation process is implemented through the machine learning library in python (mainly the scikit-learn

library), and the prediction process is performed to calculate related indicators such as AUC, accuracy, recall and specificity as shown in Tabel 1, Tabel 2 and Figure 5.

Tabel 1. The effectiveness of BP neural network, random forest and decision tree in predicting END in ACI patients in the training set.

Model	AUC	Accuracy	Recall rate	Sensitivity	Specificity	Correct rate
BP Nerual Network	0.963	1.000	0.935	0.935	1.000	0.987
Random Forest	1.000	1.000	1.000	1.000	1.000	1.000
Decision Tree	0.916	0.971	0.739	0.995	0.739	0.946

Table 2. The effectiveness of BP neural network, random forest and decision tree in predicting END in ACI patients in the training set.

Model	AUC (95%CI)	Accuracy	Recall rate	Sensitivity	Specificity	Correct rate
BP Nerual Network	0.957(0.918, 0.995)	0.682	0.882	0.882	0.912	0.912
Random Forest	0.969(0.913, 1.000)	0.948	0.989	0.989	0.925	0.947
Decision Tree	0.848(0.737, 0.959)	0.750	0.883	0.750	0.914	0.883

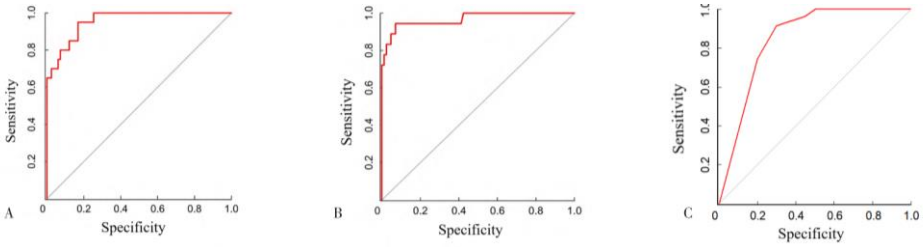

Fig. 5 BP neural network, random forest, and decision tree predict the ROC curve of END in ACI patients in the test set after intravenous thrombolysis. A. BP Neural Network; B. Random Forest; C. Decision Tree [10].

The above prediction results show that the AUC of random forest predicting END in ACI patients in the test set is greater than the decision tree. There was no statistical difference in the AUC between BP neural network and decision tree, BP neural network and random forest in predicting END in ACI patients in the test set. It is suggested that BP neural network, decision tree and random forest have good prediction performance in predicting END in ACI patients, among which the random forest is better than the decision tree in distinguishing END in ACI patients.

The prediction performance of the BP neural network and random forest algorithm on this problem is better than that of the decision tree. The random forest algorithm is slightly better than the BP neural network. However, all three can be used in adaptive algorithms to deal with complex problems.

3.3. Analysis of the Reasons

Compared with decision trees, random forests have random characteristics and do not suffer from overfitting. The main advantage of random forests and the BP neural network is to analyze data with complex non-linear relationships, and there is no limit to the number of predictor variables. In addition, it can sort feature variables according to their importance and handle both continuous and categorical variables.

All in all, the BP neural network still has a lot of room for improvement. The learning criteria can be further optimized, or regularization and other methods can be added to improve performance.

4. Further Optimization of the BP Neural Network

4.1. Shortcomings of BP Neural Network in Application

The first is that like other deep learning algorithms, the BP neural network can also perform well on training data, but due to overfitting problems, the performance of the algorithm decreases when generalizing to new data sets. And in deep networks, gradients may decrease rapidly, resulting in slow weight updates.

In addition, the BP algorithm may find a local minimum instead of a global minimum, and the convergence speed is slow. To solve or reduce the impact of its structure on the prediction results, methods such as regularization technology, small-batch stochastic gradient descent, hyperparameter optimization and adaptive learning rate can be used.

4.2. Neural Network Pruning Method Based on Zero-norm Regularization

4.2.1 Overview of zero-mode regularization

Zero-norm regularization (L0 regularization) is a regularization technique used in machine learning models and neural networks to increase the sparsity of the model [11]. Mathematically, it is usually implemented by adding a term that penalizes the number of non-zero elements in the model weight matrix.

$$\text{Regulation term} = \eta \|W\|_0 \tag{8}$$

This mathematical expression is expressed as the L0 norm of the weight matrix W. And η is a positive regularization parameter that controls the contribution of the regularization term to the entire optimization problem.

Unlike L1 regularization (which tends to shrink coefficients to zero, thus promoting sparsity) and L2 regularization (which tends to shrink all coefficients uniformly and does not promote sparsity), L0 regularization enhances the sparsity of the model directly by minimizing the number of non-zero parameters [12].

4.2.2 Mathematical expression

Network pruning is to delete connections with zero weight between neurons to achieve model compression as shown in Figure 6. To achieve the purpose of pruning, the zero-norm regularization loss minimization problem is given below.

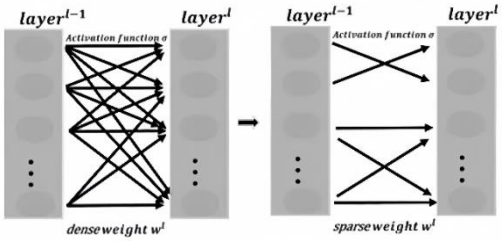

Fig. 6 Dense and sparse connections in the network [13].

$$\min_{W\in}\{\frac{1}{N}\sum_{i=1}^{N}\mathrm{L}\left(y^{(i)}, f\left(w; x^{(i)}\right)\right)+\frac{\mu}{2}\|W\|_F^2+\eta\sum_{i=1}^{L}\|W_l\|_0\} \tag{9}$$

Among them, $\eta > 0$ is a regular parameter, and $\mu > 0$ is a very small parameter. The introduction of the regularization term $\frac{\mu}{2}\|W\|_F^2$ is mainly to ensure that the problem has an optimal solution.

Formula (1) is an objective function containing a zero-mode regularization term, which attempts to minimize the prediction error (via the loss function L) and the model complexity (via the L0 norm of the weight matrix W).

$$F_{\mu}(w) \text{ referred to as: } = \frac{1}{N}\sum_{i=1}^{N} \mathrm{L}\left(y^{(i)}, f\left(w; x^{(i)}\right)\right) + \frac{\mu}{2}\|W\|_F^2 \tag{10}$$

Ergo, question (1) can be simply written as:

$$\min_{W\in}\{F_{\mu}(w) + \eta\sum_{i=1}^{L}\|W_l\|_0\} \tag{11}$$

Formula (2) is a modification of formula (1), which mainly emphasizes the loss function and regularization terms. The purpose is to find an optimal solution that can both minimize the prediction error and keep the weights sparse.

4.3. Advantages After Model Optimization

First, zero-norm regularization helps identify and eliminate unimportant connections in BP neural networks by making the weight matrix sparse. This not only reduces the complexity of the model, but also reduces the computational burden in training and inference and avoids overfitting problems.

At the same time, a sparse weight matrix can speed up forward and backward propagation because there are fewer multiplication operations involved. And because each activated neuron has a more direct connection to a specific input feature, the optimized neural network is easier to analyze and interpret.

5. Conclusion

This article first introduces the basic theory and mathematical model of BP algorithm. Then by comparing the accuracy of various adaptive algorithms when dealing with specific problems, it is easy to find out the advantages, disadvantages and applicable issues of various algorithms. The disadvantage of the original BP neural network is that it is easy to fall into gradient disappearance and local minima, resulting in slow training or even stagnation. Then the causes of the problems exposed by the BP neural network are analyzed and the zero-norm regularization method is used to optimize the BP algorithm. The optimized model has significantly improved its ability to handle complex problems and avoid overfitting problems.

References

[1] Wei A, Jian Y, Hongqiang L, et al. An R-adaptive algorithm based on self-organizing maps for solving incompressible flows with high-order discontinuous Galerkin methods. Computers and Fluids, 2024, 270106160.

[2] Guobing Q, Sifan H, Junzhu L, et al. Robust adaptive algorithm for widely-linear Hammerstein system and its application. Signal Processing, 2024, 217109354.

[3] Yong W, Kuichao L, Gaige W, et al. A fuzzy-guided adaptive algorithm with hierarchy mechanism for solving dynamic multi-objective optimization problems. Knowledge-Based Systems, 2024, 284111227.

[4] Anmol B, Vivek S, Udayan G. A temporally and spatially local spike-based backpropagation algorithm to enable training in hardware. Neuromorphic Computing and Engineering, 2023, 3(3).

[5] Ziwei S, Shuiying X, Shihao Z, et al. A multi-layer photonic spiking neural network with a modified backpropagation algorithm for nonlinear classification. Optics Communications, 2023, 546.

[6] Editors P. O. Retraction: Designing an intelligent monitoring system for corn seeding by machine vision and Genetic Algorithm-optimized Back Propagation algorithm under precision positioning. PloS one, 2023, 18(11): 0294884-e0294884.

[7] Anmol B, Vivek S, Udayan G. A temporally and spatially local spike-based backpropagation algorithm to enable training in hardware. Neuromorphic Computing and Engineering, 2023, 3(3).

[8] Ziwei S, Shuiying X, Shihao Z, et al. A multi-layer photonic spiking neural network with a modified backpropagation algorithm for nonlinear classification. Optics Communications, 2023, 546.

[9] Lihua W, Wenjing Y, Yanjuan Z, et al. Optimal parameters selection of back propagation algorithm in the feedforward neural network. Engineering Analysis with Boundary Elements, 2023, 151575-596.

[10] Shouquan X, Guowen T, Wbiao H., et al. Comparison of the efficacy of BP neural network, random forest and decision tree in predicting early neurological deterioration after intravenous thrombolysis in patients with acute cerebral infarction. Journal of Practical Heart, Brain and Pulmonary Vascular Disease, 2023, 31(02): 16-21.

[11] Hao M, Hu Y. L0 regularized logistic regression for large-scale data. Pattern Recognition, 2024, 146.

[12] Alice D, Anne F, Pascal L. L0 Regularization parameter for sparse DOA estimation of coherent signals with modeling errors. Signal Processing, 2023, 209.

[13] Zhi L. Neural network pruning method based on zero-mode regularization. Operations Research and Management, 2023, 32(10): 102-107.