An Improved BP Neural Network Based on Adaptive Genetic Algorithm

Jiale Zhang, Lianshuan Shi(✉), Jiaxing Zhao, and Shuangyu Duan

School of Information Technology Engineering, Tianjin University of Technology and Education, Tianjin 300222, China
shilianshuan@sina.com

Abstract. In order to improve the optimization effect of genetic algorithm on BP neural network, this paper proposes an improved BP neural network based on adaptive genetic algorithm. Firstly, the selection operation introduces the elite retention strategy. Secondly, adaptive operation is introduced into the crossover mutation operator, and the crossover mutation mode is optimized to adjust the population diversity, avoid the algorithm falling into the local optimal, and prevent the algorithm from precocious. Finally, the effectiveness of the proposed algorithm in reducing the time cost, improving the network fitting and improving the neural network's tendency to fall into the local optimal is verified by comparing with the two common algorithms.

Keywords: genetic algorithm · Adaptive · Algorithm optimization · BP neural network

1 Algorithm Introduction

Genetic algorithm (GA) was a computational model simulating Darwinian biological evolution [1]. By abstracting the biological evolution process, this iterative algorithm for global optimization search is obtained [2].

BP neural network is one of the most mature and widely used neural network models at present, with excellent self-learning ability and a very wide range of applications [3]. It still has some defects, for example, its learning convergence speed is slow [4], and the initial weight threshold selection has a great impact on the network training [5]. In view of these defects, this paper proposes an adaptive genetic algorithm to improve the BP neural network, improve the accuracy of the neural network and solve the problem that the neural network is prone to fall into the local optimal [6].

Y. Cao and X. Shao (Eds.): MONAMI 2022, LNICST 474, pp. 282–292, 2023.
https://doi.org/10.1007/978-3-031-32443-7_20

2 Algorithm Introduction

2.1 Genetic Algorithm Optimization

2.1.1 Initialize the Population

In this paper, the population size of 50 was determined by the fitting effect of different populations and neural network. The encoding method of this paper adopts real number encoding [7].

2.1.2 Fitness Function

Genetic algorithm is used to optimize BP neural network. After determining the topology of BP neural network, the chromosome length corresponding to the population individual can be obtained. The known training data can be used to train BP neural network. The sum of the absolute value of the error between the predicted result of the test sample and the expected output is taken as the fitness value of the individual.

2.1.3 Select Operations

The selection operation of genetic algorithm is to retain excellent individuals in the population to the next generation with a greater probability, and at the same time eliminate crossover individuals, so that the population can evolve in a more excellent direction [8]. Common selection methods include sorting selection, random selection without putting back, proportional selection. This paper introduces the elite retention strategy on the basis of proportional selection.

The basic idea of elite retention strategy is to preserve elite individuals, namely the best individuals, directly to the next generation without genetic manipulation. The introduction of elite retention strategy can ensure that excellent individuals will not be destroyed in the process of evolution, and the convergence ability of the population has been improved.

2.1.4 Adaptive Cross-Mutation

The main idea of crossover operator is to select two individuals in the population for local crossover to obtain new individuals and increase the diversity of the population [9]. It is an important part of genetic algorithm and its distinctive feature.

The main idea of mutation operator is to carry out local mutation of individual chromosomes, which is also an important part of genetic algorithm [10]. The main purposes of the algorithm are two: one is to make the algorithm have local search ability, the other is to keep the diversity of the population without destroying the good individuals of the population as much as possible.

Arithmetic crossover operator: now set two individuals X_t^m and X_t^n; arithmetically cross at t; then the two new individuals generated at after crossing are:

$$X_{t+1}^m = X_t^m - r * (X_t^m - X_t^n) \tag{1}$$

$$X_{t+1}^{n} = X_{t}^{n} - r * (X_{t}^{m} - X_{t}^{n}) \tag{2}$$

In formula 1 and 2, when parameter r is set to constant, it is uniform arithmetic crossing. When r is a variable, it is a non-uniform arithmetic crossover.

The mutation operation uses real variation:

$$X_{t+1} = X_t - u * rand \tag{3}$$

In the formula, u is a parameter, $rand$ is the corresponding to a random number.

In the evolution of population, the probability of crossover operator and mutation operator will greatly affect the result of the algorithm. In the crossover operation, if the crossover probability is too large, the good individuals of the population are easy to be destroyed. On the contrary, if the value is too small, it can not promote the diversity of the population well. In the mutation operation, too large mutation probability will make the algorithm similar to random search, making it lose the most distinctive characteristics of biological evolution of genetic algorithm [11], which cannot ensure the diversity of the population and obtain the optimal value.

Based on the above analysis, the crossover and mutation probabilities are optimized in this paper. On the basis of the original, evolutionary algebra and individual fitness value of the population are introduced, and a new adaptive crossover and mutation probability formula optimization algorithm is proposed. After adaptive adjustment, the population evolution effect can be adjusted in real time according to the fitness value and the current evolution algebra, which helps the algorithm to jump out of the local optimal.

The adaptive crossover probability formula is as follows:

$$P_c = \begin{cases} 0.8 - 0.3 * (f_{avg} - f_s)/(f_{max} - f_{min}), f_s \leq f_{avg} \\ 0.9 - 0.7 * g/G, f_s > f_{avg} \end{cases} \tag{4}$$

The adaptive mutation probability formula is as follows:

$$P_m = \begin{cases} 0.01 + 0.09 * (f_{avg} - f)/(f_{max} - f_{min}), (f \leq f_{avg}) \\ 0.08 + 0.02 * g/G, (f > f_{avg}) \end{cases} \tag{5}$$

Among them:

f_s is an individual with small fitness value in a cross parent;

f_{avg} is the average of the current population fitness value;

f_{max} is the maximum fitness value of individuals in the current population;

f_{min} is the minimum fitness value of individuals in the current population;

f is the fitness value of the mutated individual;

g is the current evolutionary algebra;

G is the total iteration number of the algorithm.

2.2 BP Neural Network

BP neural network is divided into the following parts. Firstly, the number of hidden layers of the neural network is determined, and the topology structure of the current neural network is determined jointly with the nodes of the input layer and output layer [12].

As shown in Fig. 1, in the topological structure of BP neural network, X_1, X_2 is the input value of the neural network, Y_1, Y_2 is the predicted value of the BP neural network, and is the weight of the BP neural network.

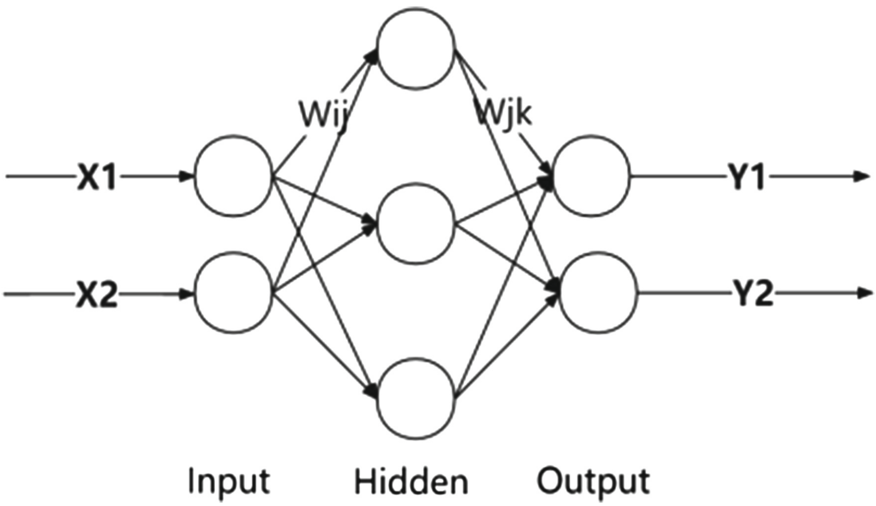

Fig. 1. Topology of BP neural network.

Secondly, the initial weights of the neural network input layer, output layer and hidden layer and the initial thresholds of hidden layer and output layer are determined. Finally, through training, the weight and threshold of the neural network can be dynamically updated through the error obtained by training. After repeated iterative evolution of the algorithm, the final neural network can be obtained, and finally the test data is used for evaluation.

2.3 Algorithm Process

Algorithm steps:

Step 1: According to the input and output of the system to determine the topology of the neural network;
Step two: parameter setting and fitness function selection;
Step 3: Initialize the population randomly;
Step 4: train the neural network, get the error and calculate the fitness value;

Step 5: Adaptive genetic algorithm iteration is carried out to get the final neural network;
Step 6: Test the final neural network, and evaluate the final neural network according to the predicted results and expected output.

The overall flow of the algorithm is shown in Fig. 2 below.

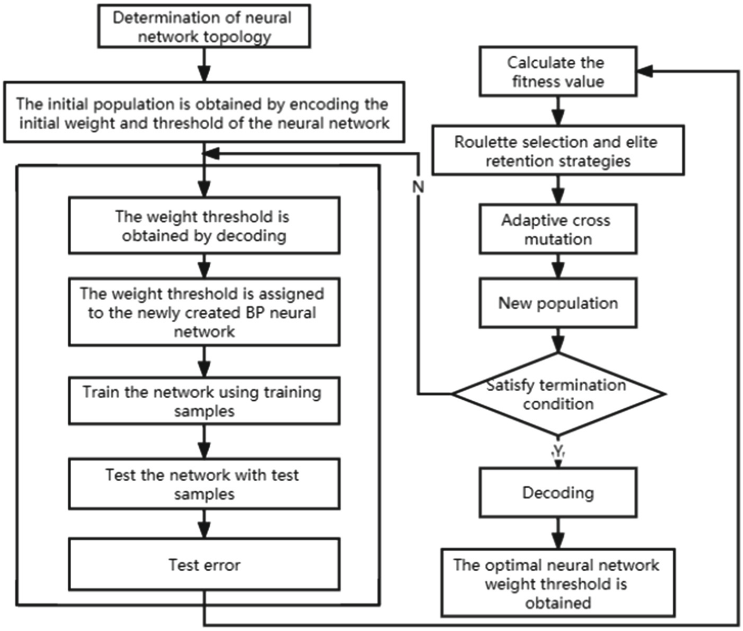

Fig. 2. Algorithm flow chart.

3 Experimental Analysis

In this paper, non - fitting linear function $y = x_1^2 + x_2^2$ is used to test the neural network; Experimental data: 4000 sets of data, among which 3900 sets of data were used for training network and 100 sets of data were used for neural network test; Population size: 50; Iterative algebra: 50.

3.1 Experimental Analysis

In this paper, non - fitting linear function is used to test the neural network; Experimental data: 4000 sets of data, among which 3900 sets of data were used for training network and 100 sets of data were used for neural network test; Population size: 50; Iterative algebra: 50.

3.2 BP Neural Network Simulation

Figure 3 shows BP neural network simulation optimized by standard particle swarm optimization algorithm.

Figure 4 shows the simulation of BP neural network optimized by simple genetic algorithm;

Figure 5 shows the BP neural network simulation optimized by adaptive genetic algorithm proposed in this paper.

In the diagram, the abscissa represents the test sample and the ordinate represents the output. Figure 6 shows the comparison of sample errors of the three algorithms in the 100 groups of test data.

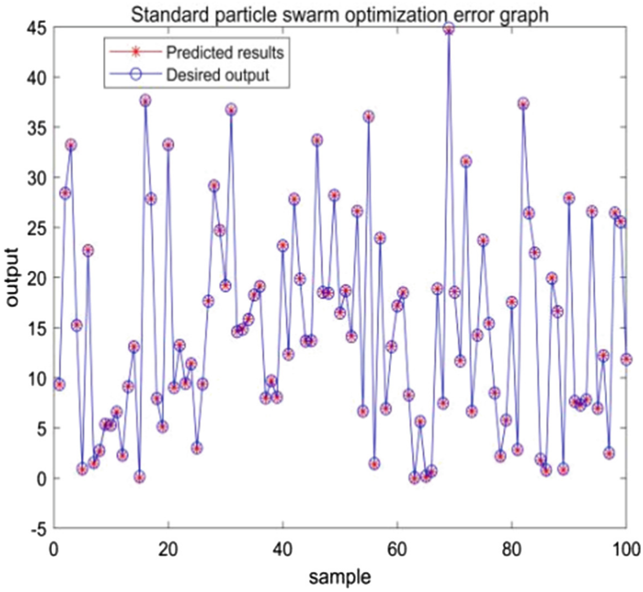

Fig. 3. Standard particle swarm optimization.

In the above figure, it can be clearly seen that the improved adaptive genetic algorithm proposed in this paper has obvious optimization effect on the neural network, the error is the smallest of the three algorithms, and its fitting function is also closer to the actual expected output.

It can be concluded that the improved adaptive genetic algorithm proposed in this paper has a great role in optimizing the accuracy of BP neural network.

3.3 Comparative Analysis

As shown in the figure above, Fig. 7 is the iterative evolution curve of the total error of the sample.

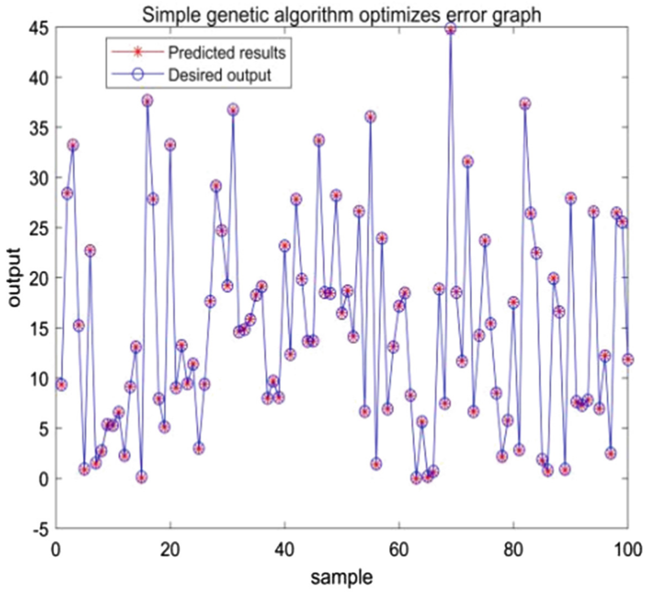

Fig. 4. Simple genetic algorithm.

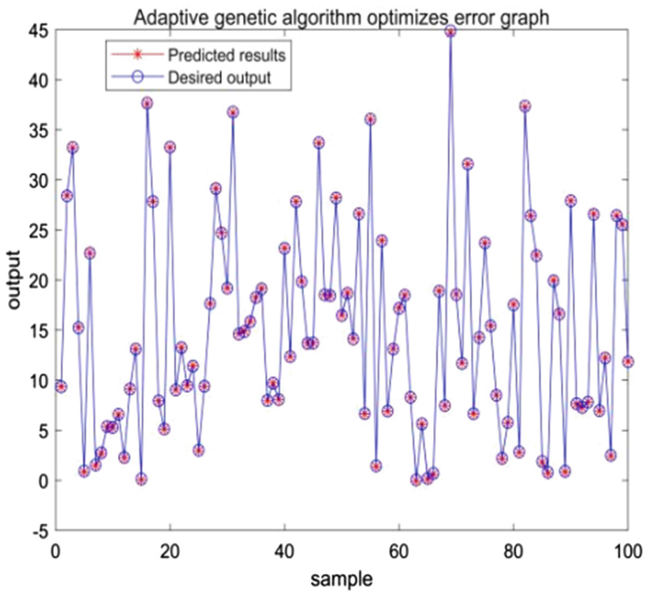

Fig. 5. Adaptive genetic algorithm.

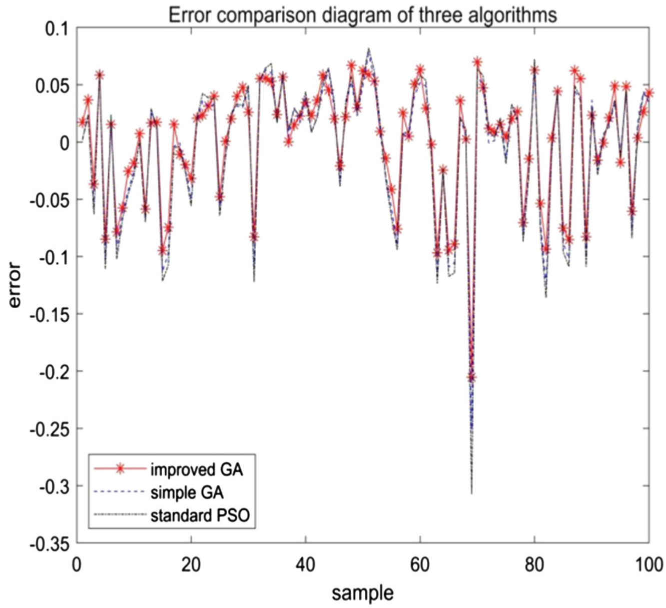

Fig. 6. Error comparison diagram of three algorithms.

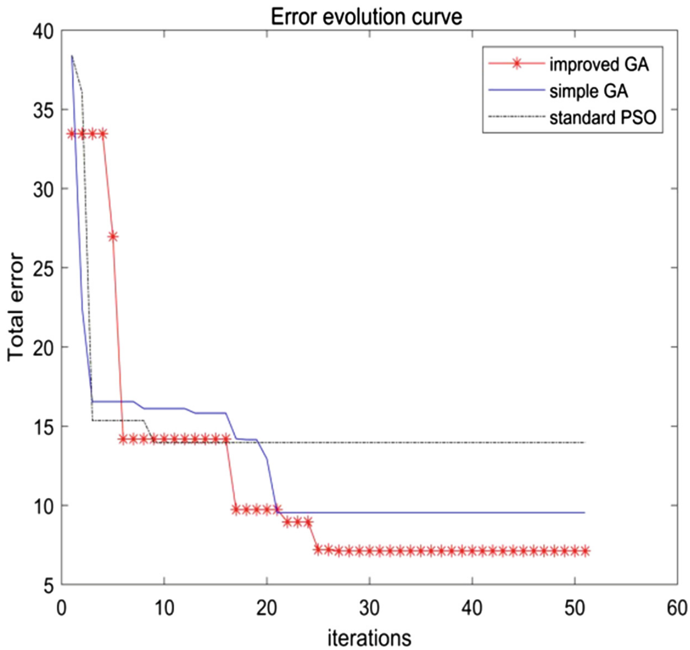

Fig. 7. Error evolution curve.

The figure shows that the initial errors of the improved adaptive genetic algorithm, simple genetic algorithm optimization and standard particle swarm optimization proposed in this paper are 33.4589, 38.402 and 38.4122, respectively. The algorithm proposed in this paper is far superior to the other two algorithms in the initial population, with an increase of 12.87% and 12.89% respectively. At the same time, it can be seen from the figure that the improved algorithm in the iterative process has better performance in jumping out of the local optimal value.

After the completion of iteration, the final errors of the improved adaptive genetic algorithm, simple genetic algorithm optimization and standard particle swarm optimization proposed in this paper are 7.1223, 9.5396 and 13.9649, respectively. The improved algorithm proposed in this paper is also far superior to the other two algorithms, improving by 25.33% and 48.99%, respectively. From the above analysis, it can be seen that the improved adaptive genetic algorithm proposed in this paper has a better performance in terms of error, and its optimization of neural network is more significant.

Table 1. Comparison table of algorithm performance comparison items

Compare the item and Algorithm	The mean	The variance	Sum of squares	The elapsed time
Optimization of BP neural network by standard particle swarm optimization	−0.0057	0.0645	4.8935	353.636903 s
Optimization of BP neural network by simple genetic algorithm	−0.0049	0.0580	4.4439	219.534010 s
Optimization of BP neural network by adaptive genetic algorithm	0.0014	0.0510	4.0535	205.918277 s

As shown in the above table, Table 1 is the comparison table of algorithm performance comparison items. The training error and evolution curve have been analyzed in the previous paper. To further explore the performance of the algorithm, several common comparison terms are used. Are the mean of errors, the variance of errors, the sum of squares of errors and the running time of the program.

Compared with standard particle swarm optimization and simple genetic algorithm, the mean error of the proposed algorithm is improved by 75.44% and 71.43% respectively. The error variance was increased by 20.93% and 12.07%, respectively. The sum of error squares increased by 17.18% and 8.81%, respectively. The program running time was increased by 41.77% and 6.20%, respectively. From the above analysis, it can be seen that the algorithm proposed in this paper has excellent performance in the four comparison terms, which proves that it has obvious advantages in the stability and accuracy of the neural network. In summary, the improved algorithm proposed in this paper reduces the time cost, reduces the network error, and improves the fitting of the neural network.

4 Conclusion

This paper presents a new adaptive genetic algorithm to improve the BP neural network. In the algorithm optimization part, the selection operator is optimized and the elite retention strategy is introduced. At the same time, adaptive operation is introduced in crossover and mutation operators to adjust the population diversity, so as to avoid the algorithm falling into local optimal and precocious algorithm.

Through the comparison experiment of the three groups of algorithms, the results show that the BP neural network optimized by adaptive genetic algorithm is obviously superior to the BP neural network optimized by simple genetic algorithm and standard particle swarm optimization. It has better fitting and effectively solves the shortcomings of the neural network which is easy to fall into the local optimal. The improved algorithm proposed in this paper is of great significance in the research and improvement of neural networks to improve the accuracy of neural network prediction.

References

1. Li, M., Xu, B., Kou, J.: Combination of genetic algorithm and neural network. Syst. Eng. Theory Pract. (02), 66–70+113 (1999)
2. Li, W., Song, D., Chen, B.: Artificial neural network based on genetic algorithm. Comput. Eng. Design (02), 316–318 (2006). https://doi.org/10.16208/j.issn1000-7024.2006.02.043
3. Meng, D., Fan, C., Wang, J.: Research on improvement of BP neural network based on chaotic genetic algorithm. Math. Theory Appl. **34**(01), 102–110 (2014)
4. Tian, X., Song, T., Liu, Y.: Optimization of structure and parameters of BP neural network based on genetic algorithm. Comput. Appl. Softw. (06), 69–71 (2004)
5. Zhai, Y., Li, H., Liu, H., Yuan, X.: Initial weights of neural network are optimized with genetic algorithm method. J. Jilin Univ. (Eng. Sci.) (02), 45–50 (2003). https://doi.org/10.13229/j.carolcarrollnkijdxbgxb2003.02.011
6. Wang, Z., Liu, Z., Gao, C., Lu, Y.: Application of genetic algorithm in BP network weight learning. J. Gansu Univ. Technol. (02), 20–22 (2001)
7. Li, P., Fan, D.: Research on improved genetic algorithm based on real coding. J. Astronaut. Meas. Technol. (06), 61–64 (2007)
8. Wu, W., Yue, T.: A review of selection operators in genetic algorithms. Fujian Comput. **28**(06), 43–44 (2012)
9. Deng, C.Y.: Crossover operator analysis of genetic algorithm. Agric. Netw. Inf. (05), 124–126 (2009)
10. Shen, C., Yue, T.: Review of mutation operator of genetic algorithm (GA). J. Horizon Sci. Technol. (23), 107–108+188 (2012). https://doi.org/10.19694/j.carolcarrollnkiissn2095-2457.2012.23.043

11. Xiong, S., Dong, L.: Adaptive genetic algorithm for labeled planar stereograms. J. Graph. **39**(06), 1105–1111 (2018)
12. Wang, J., Hu, Y., Zhang, X., Wu, B.: Based on GA - BP neural network to determine the water leakage of inevitable. J. Water Resour. Hydropower Technol. (Both Engl. Chin.) **52**(12), 201–211 (2021). https://doi.org/10.13928/j.carolcarrollnkiwrahe.2021.12.020