Research on Optimization of BP Neural Network

Qiuyun He

Research School of Computer Science,
Australian National University
u7003977@anu.edu

Abstract. In view of the inherent limitations of the BP neural network and the problems that arise when it is applied to performance prediction, this article explores the influence of the depth of the neural network on the prediction, then uses Genetic Algorithms to optimize the weights and thresholds of the BP neural network. We prove that the effect of the model will not get better as the model deeper. Deeper models are suitable for larger amounts of data, but some small models are suitable for smaller amounts of data. The comparison with the basic BP neural network and the optimized BP neural network shows the effectiveness and feasibility of the optimization. The analysis results show that the genetic neural network model has high accuracy in performance prediction and has certain practical value.

Keywords: Deep Learning · Genetic Algorithm · BP Neural Network

1 Introduction

It is an exploratory and feasible method to predict the final student's grades based on daily grades, especially in the case of COVID-19, which becomes even more important. Most scholars widely used BP neural network model to achieve predict students' performance. Comparing the results of different models, the linear regression model achieves the best effect on all features, and Linear regression can be regarded as a special neural network model without hidden layers. In terms of small important features, the neural network achieves the best results [1]. This article selects the COMP1111 grade dataset [2] and extends the above conclusion. While achieving certain results, the BP neural network algorithm has been exposed to the disadvantages of slow convergence, low efficiency, and inadaptability in the field of course performance prediction. The choice of network structure, initial connection weights, and thresholds have a great impact on network training, but they cannot be accurately obtained. In view of these characteristics, Genetic Algorithms can be used to optimize the neural network and exploring the influence of the depth of neural network.

1.1 Data description

Before proceeding to the next step of the method description, this article first gives a brief introduction to the data set, and its example diagram is shown in Fig. 1. From Figure1, we can find that there are a total of 153 samples, including 16 dimensions, where 'Regno' is the number and it does not contain data information, 'final' is the final grade result, which is the model needs to predict based on the features. From the data, we can find that there are two types of features, one is category type, such as' Crse/Prog','S','ES','Tutgroup', and the other is numerical type features, such as 'lab2','h1','mid','f1', etc. and there are missing values in the data. Since the labels of eight samples in the data are missing, this article deletes these eight samples.In order to have a more intuitive understanding of the data, this article first visualizes the data.From the visualization results of the scatter plot Fig. 2, we can find that the linear correlation between mid and final final is the greatest. Further, we solve the covariance matrix of the data as Fig. 3. Since there are missing values in the array features in the data, this article uses the column mean filling method to fill in the missing values.On the other hand, this article also visualized the Categorical characteristics of the data as Figure4.We can find that the final distributions of different classification features are inconsistent. Based on classification features, this article constructs its final statistical features, including maximum, minimum, mean, and standard deviation. In addition, this paper constructs a new feature 'sum score' by summing all the score features. To correctly evaluate the effect of the model, this paper uses the Holdout model to evaluate the effect of the model, randomly splitting 80% of the data as the training set for model training, and the remaining 20% of the data as the test set for evaluation.

	Regno	Crse/Prog	S	ES	Tutgroup	lab2	tutass	lab4	h1	h2	lab7	p1	f1	mid	lab10	final
0	168826	3971	2	FS	T1-yh	2	3	2.5	19.5	.	2.5	11	.	33	2.5	71
1	168883	3971	2	F	T9-ko	3	5	2	20	17	.	8	.	27	2.4	67
2	168907	3400	1	F	T6-no	3	3	3	10	.	2	.	.	9.5	2.4	30
3	169379	3970/1061	2	F	T3-ku	2	3	2	20	19.5	2	15.5	.	21	.	62
4	169717	3970/1061	2	F	T10-yh	2	3.5	1.5	19	15.5	2	17.5	.	13	2.5	58
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
148	8111648	3970/1000	2	F	T8-no	2.5	0	1.5	19.5	5.5	1.5	12.5	.	29	2.4	58
149	8126247	3484	3	F	T9-ko	3	5	3	18	20	3	20	18	24	2.4	79
150	8126809	3400	1	F	?	.	.	.	.	.	.	.	.	.	.	.
151	8130182	3420	3	F	?	.	.	.	.	.	.	.	.	.	.	.
152	8130678	3420	1	F	T2-no	3	2	2.5	18.5	18.5	0	13	15	8.2	2.4	32

153 rows × 16 columns

Fig. 1. The COMP1111 grade data set

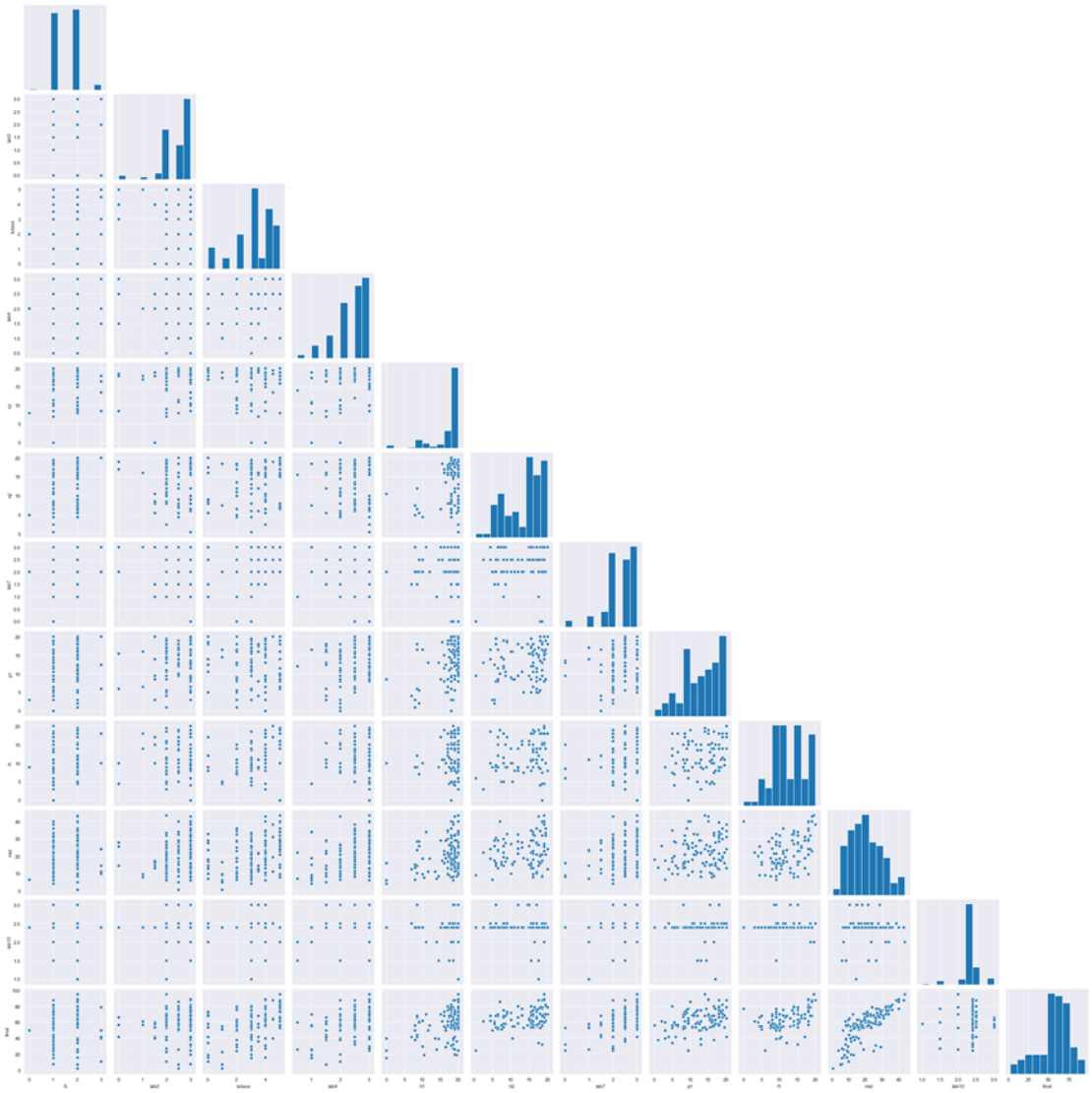

Fig. 2. The scatter plot of data

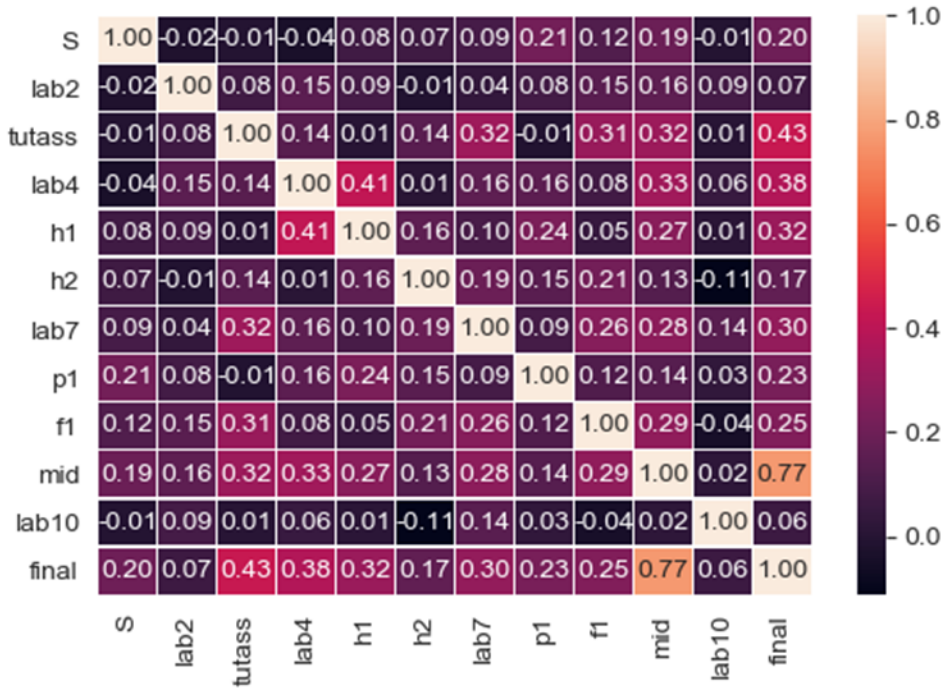

Fig. 3. the covariance matrix of the data

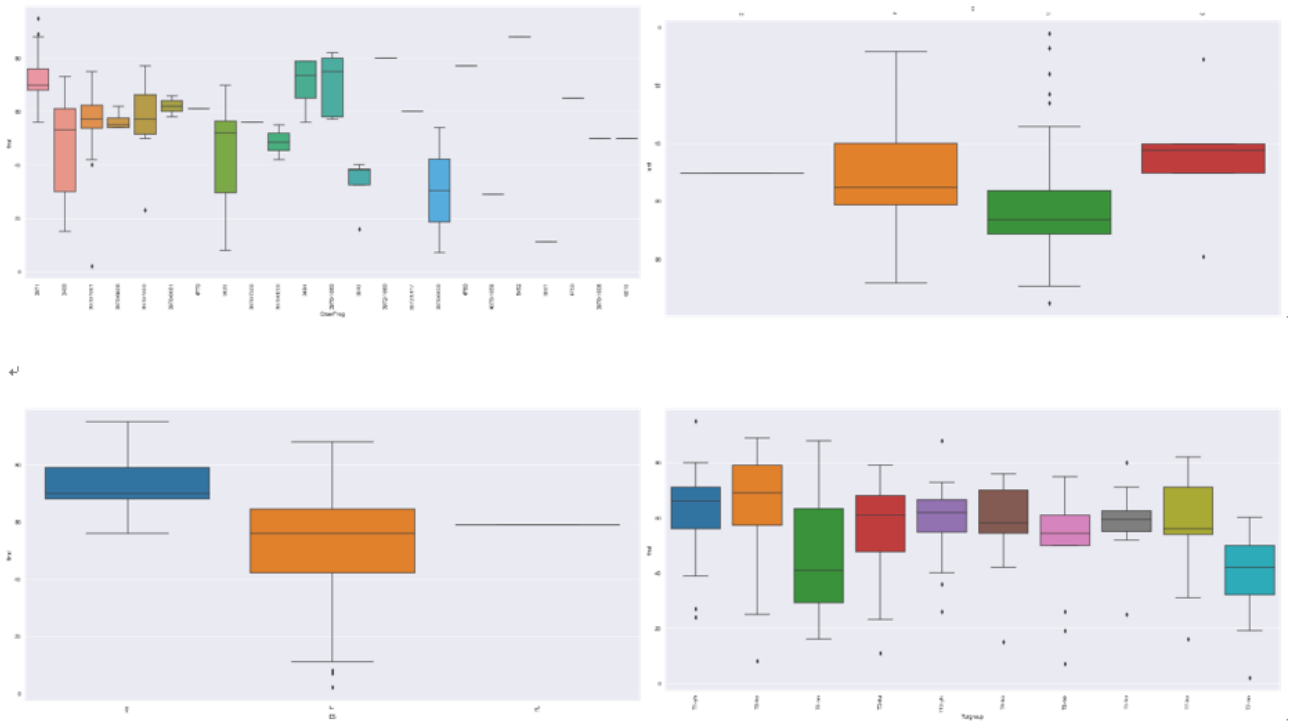

Fig. 4. The Categorical features visualization

2 Model

2.1 Network Layer

Comparing the structure of two hidden layer model to one fully connected layer. The implementation of the fully connected layer is implemented by nn.Linear In Pytorch. The reason is that the amount of network parameters that need to be trained is proportional to the amount of data in deep learning. The amount of data we use is a little small, so when using more than one layer of neural network for training, there will be a risk of overfitting.

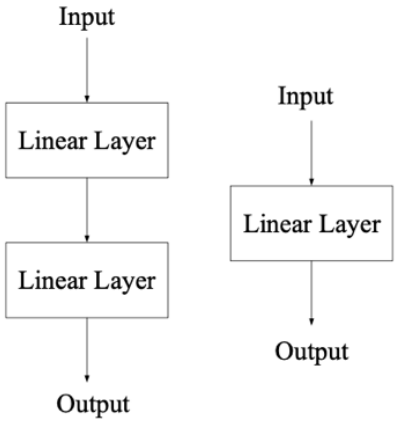

Fig. 5. The structure of two types of network

2.2 Design of genetic algorithm combined with BP neural network

Genetic algorithm has strong global search ability and strong robustness, which is very suitable for optimizing BP neural network [3]. Combining the two can greatly reduce the probability of the network falling into a local minimum, and at the same time further improve the convergence speed of the network, which can quickly obtain the global optimal solution to the problem. The basic principle of genetic algorithm optimization of BP neural network is to The weights of the BP neural network are controlled by the genetic algorithm, that is, the node weights and thresholds of the hidden layers of the BP neural network are used as the input of the genetic algorithm, and they are coded to generate the initial population, and the selection crossover and mutation of the genetic algorithm are used To produce new offspring, that is, the new neural network weights and thresholds, and then return them to the BP neural network, and the neural network will perform the subsequent solution process. From the point of view of the algorithm, the genetic algorithm is used to search in the solution space of the target problem. When a better network form is searched, the BP algorithm is used for positioning to find the exact solution in the better solution space. Optimal solution or satisfactory solution. The main process of genetic algorithm combined with BP algorithm is shown in Figure 2.

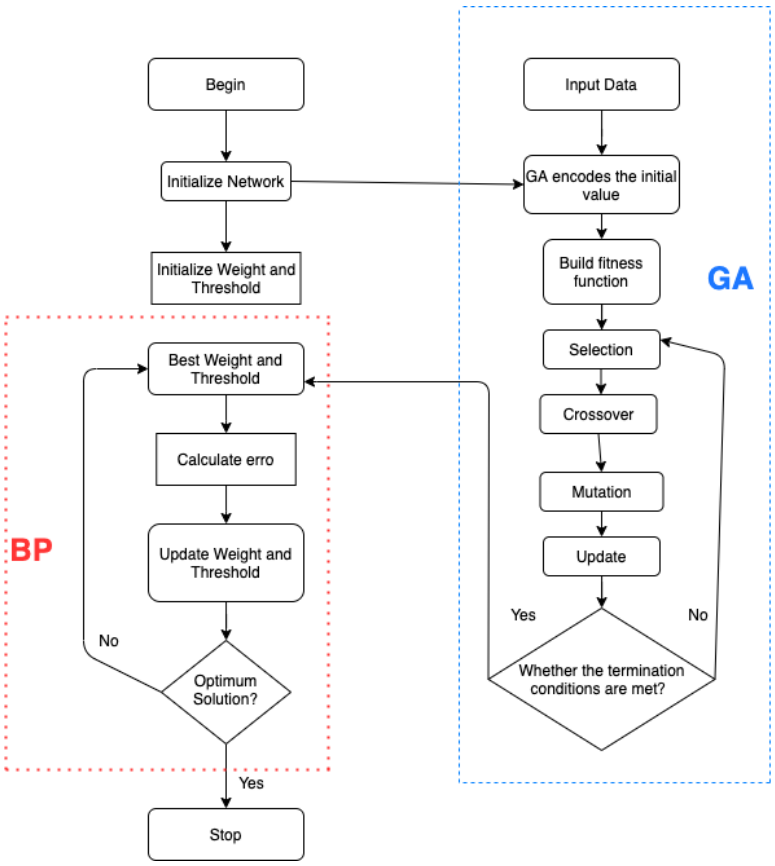

Fig. 6. Flow chart of genetic algorithm combined with BP neural network

2.2.1 BP Neural Network Design

1. Design of the number of input and output layer nodes

The factor entered in the text is the 5 courses with the highest correlation with the course to be predicted, so the number of nodes in the input layer is 5. The output layer is the output result of the neural network, that is, the grade of the target course, so the number of nodes in the output layer is 1.

2. Design of the number of hidden layers and the number of nodes

BP neural network needs to set the number of hidden layers and the unit of each hidden layer Count. Studies have shown that the BP neural network with double hidden layers is sufficient to deal with most of the complex problems, so it only needs to analyze the performance of no hidden layer, single hidden layer and double hidden layer. The final topology setting for the BP network is: input layer + double Hidden layer + output layer. Each hidden layer has 20 nodes.

2.2.2 Genetic Algorithm Design

The input layer of the neural network has 5 neurons, the first hidden layer has 20 neurons, the second hidden layer has 20 neurons, and the output layer has 1 neuron. The ownership value and threshold value of the BP neural network are used as the parameters to be coded, so there are $(5+1)\times 20+(20+1)\times 20+(20+1)\times 1=561$ parameters to be optimized. These parameters make up a chromosome. The sequence of genes in the chromosome is input layer to first hidden layer weight, first hidden layer to second hidden layer weight, second hidden layer to output layer weight, first hidden layer threshold, second hidden layer threshold, Output layer threshold. The genetic algorithm has designed a parameter that can reflect the adaptability of chromosomes. It reflects how well each chromosome compares with the optimal solution. How to make the choice of the fitness function is to compress the fitness and expand the fitness. A balance should be sought between them, and at the same time, smaller calculations, stronger versatility and consistency should be achieved [4].

If the population is too small, it will not provide enough sampling points, resulting in poor algorithm performance; if the population is too large, although optimization information can be added to prevent premature convergence, it will undoubtedly increase the amount of calculations and cause the convergence time to be too long, which manifests as convergence. slow. According to experience, the population size is generally selected between 20 and 100.
The greater the crossover probability (Pc), the faster the generation of new individuals. However, if Pc is too large, the genetic model is likely to be destroyed, and it is easy to destroy the structure of individuals with high fitness. If the crossover probability is too small, the search process will be slow or even stagnant. The mutation operation is a disturbance to the population model, which is conducive to increasing the diversity of the population. If the mutation probability (Pm) is too small, it will be difficult to generate a new individual structure. If the value is too large, the algorithm becomes a pure random search. According to experience, the crossover probability (Pc) is generally selected as a number between 0.4 and 0.99, and the mutation probability (Pm) is generally selected as between 0.0001 and 0.1[5].
If the genetic algebra is too small, the algorithm is not easy to converge, and the population is not mature; the genetic algebra is too large, the algorithm has converged, or the population is premature and cannot converge again. It is meaningless to continue evolution, and waste time and computing resources. According to experience, the genetic algebra is generally selected from 100 to 500 generations.

According to the training sample set, the total error E obtained by training the neural network is:

$$E = \sum_{p=1}^{p} E_p = \frac{1}{2}\sum_{p=1}^{p}\sum_{l=1}^{k}(t_{pl} - o_{pl})^2 \quad (1)$$

In the formula: p is the number of nodes in the input layer, k is the number of nodes in the output layer, $t_{pl} - o_{pl}$ represents the difference between the actual output and the expected output of the l-th node of the p-th sample of the output layer.
The fitness function is:

$$f = \frac{1}{1+E} \quad (2)$$

3 Results and Discussion

3.1 Network Layer

The article compares the difference between the two models and the loss curve. In addition, we chose the mean square error function as the evaluation method of the model. From Fig.7 We can see that the effect of the model is better when one fully connected layer is used. The number of layers of a deep neural network is not as many as possible but should be consistent with the amount of data. When the amount of data is larger, there is enough data to fit the model. When the amount of data is small, fewer parameters of the model are required.

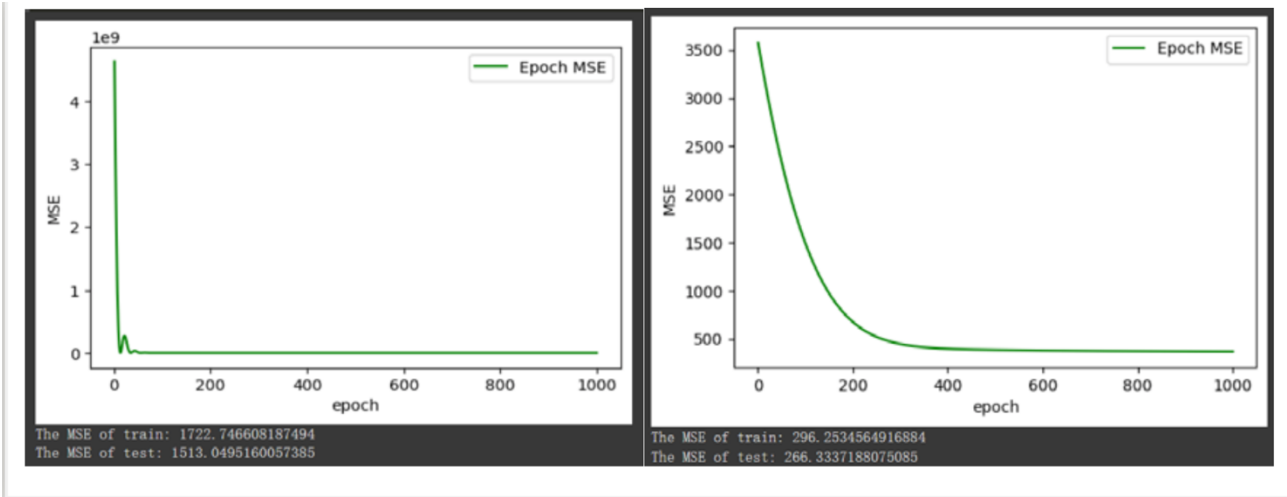

Fig. 7. The model convergence status: Loss vs epoch.

3.2 Use GA-BP algorithm to predict performance

After preprocessing the data, only 5 related scores "H1" "H2""lab7""P1""Mid" are retained.
It finds that if use “Final” grade as output, accuracy is only 43.2%. But if only predict the rate of pass, the accuracy would improve much higher. In this case, create a new column as “Pass”, if final grade is over 51, the data of this column would be 1, or it would be 0 instead. Using “Pass” as output. This article compares accuracy with GA-BP algorithm and pure BP algorithm.

After training, the accuracy with GA_BP is about 91%

```
loss:  tensor(0.0150, grad_fn=<MseLossBackward>)
epoch:  2 - 4 - 136
GABP Accuracy:  0.9084967320261438
```

Fig. 8. Loss and Accuracy of GA-BP

The accuracy with BP is about 30%

```
loss tensor(1., grad_fn=<MseLossBackward>)
BP Accuracy:  0.3006535947712418
```

Fig. 9. Loss and Accuracy of BP

Compare the results, GABP algorithm accuracy is much better than BP. The loss is also smaller than BP. Based on the comparison of the two algorithm models, it can be concluded that the reason for this gap is that the number of input samples is relatively small, and the performance data is characterized by discrete but high repetition, and the effect of training is more dependent on hidden layer nodes and weights. This is the core of the genetic algorithm's optimization of BP neural network, so the effect is more significant. This shows that genetic algorithm optimization of BP neural network does have certain advantages when applied to similar fields such as performance prediction.

4 Conclusion and Future work

This paper discusses the various factors of neural network optimization. Compared the mean squared error of different depths of neural networks, and the influences of using GA to find the optimum weights and thresholds on BP neural networks. The neural network model combined with genetic algorithm has good accuracy and smaller error value. Compared with the basic BP neural network, the effectiveness and feasibility of the optimization are shown. However, whether the application of RBF neural network to the field of performance prediction will achieve better results than the BP neural network needs to be verified through experiments in the next step.

References

1. Qiuyun He, Research on Regression Forecast of BP Neural Network (2021)
2. Choi, Ecy , and T. D. Gedeon . ”Comparison of extracted rules from multiple networks.” IEEE International Conference on Neural Networks IEEE, (1995).
3. Yin Ran et al. "Research on Software Defect Prediction Model Based on SA-BP Neural Network." Journal of Southwest China Normal University: Natural Science Edition 38.008(2013):147-152.
4. Chen Yong. Research and Implementation of Performance Forecast Based on Genetic Neural Network[J]. Modern Electronic Technology (5 Issue): 96-100.
5. Optimization of BP neural network based on genetic algorithm https://blog.csdn.net/tuqitimi/article/details/107234231
6. NeuralGenetic: Training Neural Networks using the Genetic Algorithm https://github.com/ahmedfgad/NeuralGenetic