Software defect prediction based on CS-BP neural network

Jiawang Ma
College of Information Engineering, Shanghai Maritime University, Shanghai, 201306, China
e-mail: 21104963@qq.com

Abstract. Software defect prediction can detect whether there are defects in the program module so as to effectively reduce the unnecessary cost of software development and maintenance. In this paper, the limitation of the traditional BP neural network in the field of defect prediction leads to the inaccuracy of the prediction results. By using the global optimization ability of cuckoo search, the BP neural network is improved, the important initial parameters of the network are optimized, and the software defect prediction method of CS-BP is proposed. The experimental results show that compared with traditional machine learning algorithms such as BP neural network, J48 and SVM, CS-BP method has a better effect on the prediction of software defects.

1. Introduction
With the increasing complexity of software system, there are inevitably some defects in software. The potential harm caused by these defects brings trouble to software users and developers. Generally speaking, software reliability affects the cost and quality of software development[1]. Software defect prediction can predict the current module that may be wrong through the historical defect data in the code writing stage. Correcting defects before software release can not only reduce the cost of software development and maintenance, but also effectively avoid the risk of defective software in the process of operation. Therefore, software defect prediction is very important to improve the reliability of software[2].

At present, the main algorithms used in the field of software defect prediction are Markov Model, Naive Bayes, Support Vector Machine, Linear Discriminant Analysis, etc[3]. Although these methods have certain ability of defect prediction, they also have some limitations. For example, in the case of many parameters and complex model structure, the efficiency of naive Bayes is low; Linear discriminant is easy to over fit the data, and the prediction samples should obey Gaussian distribution; Markov model needs to realize many premises, so it has great limitations in practical application.

Based on the research and imitation of human brain neurons, neural network algorithm proposed by scholars for the realization of artificial intelligence has been widely used in the field of software defect prediction in recent years. The neural network using back propagation algorithm is the most, but because it is easy to fall into local optimum, it is difficult to find the global optimal solution in the application of prediction. So there are still some drawbacks. Cuckoo search[4] refers to cuckoo's parasitic oviposition strategy and fly's special Levy flight mode, which can quickly and effectively find the optimal solution of the problem. CS is a general algorithm, which can be combined with other algorithms to get better performance.

In order to effectively solve the over dependence of BP algorithm on initial weight and threshold, this paper combines CS and BP neural network to establish CS-BP software defect prediction method.
2. Related work

2.1. Prediction of project defects by BP neural network

The essence of neural network is a nonlinear system that uses a large number of computing units to simulate human brain neurons[5], and finally realizes the intelligent processing behavior of human brain. The process is: for the input vector $X = \{x^1, x^2, \cdots, x^n\}$, the corresponding output vector is known $Y = \{y^1, y^2, \cdots, y^m\}$, the network outputs according to the actual situation $Z = \{z^1, z^2, \cdots, z^n\}$. The error between $Z$ and $Y$ modifies the weight to make $Z$ as close as possible to the expected $Y$, that is, the output layer error is trained to the minimum. In addition to the input and output layers, BP neural network can approach any nonlinear function with only one hidden layer[6]. The process of defect prediction is shown in figure 1:

![Flow chart of software defect prediction based on BP network](image)

2.2. Cuckoo search algorithm

Cuckoo search algorithm was first proposed by Yang and Deb in 2009. It is a random optimization bionic algorithm based on cuckoo parasitic oviposition strategy and Levy flight behavior of Drosophila. The algorithm has the advantages of easy implementation, few parameters, strong optimization ability and easy combination with other algorithms. In order to simplify the simulation of cuckoo parasitic behavior, the algorithm has three basic assumptions:

1) Each cuckoo produces only one egg in a random nest;
2) Among the parasitic nests, the nests with better fitness will be preserved to the next generation;
3) The number of nests in the flight range remained the same, and the host birds chose to abandon or search for new nests after sensing the parasitic eggs.

Cuckoo looking for a parasitic nest is an optimization process. The specific steps are as follows:

1) Initialize the nest sequence and randomly generate N nests, in which M eggs represent the dimension of the solution;
2) The fitness value of each nest after initialization was calculated;
3) After the cuckoo eggs were found by the host bird, a new nest was found by formula (1), and the fitness value of the new nest sequence was calculated. Compared with the best nest, the better nest was reserved. The location $x$ of cuckoo's new nest is as follows:

$$x_i (k+1) = x_i (k) + \alpha \odot \text{Levy}(\lambda)$$ (1)

In the formula 1, the algebra of cuckoo is $i$, and the number of nests of cuckoo is $k$, which is used to define the flight range of cuckoo, $\odot$ represent the matrix multiplication, $\text{Levy}(\lambda)$ represent the random flight path of cuckoos, as shown in formula (2):

$$\text{Levy} \sim \lambda \sim \lambda \sim \lambda \sim \lambda \sim \lambda \sim \lambda \sim \lambda \sim \lambda \sim \lambda$$ (2)

The generation of random step $S$ is shown in formula (3):
In formula 3, $u \sim N\left(0, \sigma_u^2\right), \nu \sim N\left(0,1\right)$, \[ \sigma_u = \left( \frac{\Gamma\left(1+\beta\right) \sin\left(\pi\beta / 2\right)}{\beta \Gamma\left(\left(1+\beta\right)/2\right) 2^{(\beta-1)/2}} \right)^{1/\beta} \] 

(4)

4) Probability $\rho$ is used to improve the poor fitness nest, and then random walk is used to generate the same number of solutions instead of the discarded ones:

\[ x_i(k+1) = x_i(k) + r \odot \text{Heaviside}(\rho - \varepsilon) \odot \left( x_i(k) - x_m(k) \right) \]  

(5)

5) The fitness of the new generation solution generated by step 2) - 4) is calculated and the optimal solution is selected;

6) Repeat steps 2-5 until the maximum number of iterations is met, and the global optimal nest position is output.

3. Cs optimized bp neural network

In order to improve the shortcomings of BP neural network in application, and improve the prediction accuracy and generalization ability, this paper uses CS algorithm to search better initial connection weights and thresholds to achieve the purpose of optimization. The steps are as follows:

1) Determine the structure of BP neural network, initialize the number of nests $N$, $\rho$ and the maximum number $N_{\text{max}}$ of iterations and other parameters;

2) Cuckoos randomly parasitize $n$ nest position sequences $P_i = [x_1^{(i)}, x_2^{(i)}, \ldots, x_n^{(i)}]^T$. The nest position represents the initial weights and thresholds of the network. The fitness values of the positions are calculated and compared to find the optimal parasitic nest position $x_0^{(i)}$;

3) After the nest position $x_0^{(i)}$ is obtained, the other nests are updated according to formula (1) to get a new set of position sequences. After calculation and comparison with $P_{t-1} = [x_1^{(t-1)}, x_2^{(t-1)}, \ldots, x_n^{(t-1)}]^T$, the poor position $P_{t-1}$ will be replaced to get the better nest position $K_t = [x_1^{(t)}, x_2^{(t)}, \ldots, x_n^{(t)}]^T$;

4) By comparing the random numbers $r$ and $\rho$, the location $K_t$ with lower detection probability of cuckoo eggs is reserved, and other nests are updated at the same time. After calculation and comparison, the poor location is replaced to get the better nest location $P_t = [x_1^{(t)}, x_2^{(t)}, \ldots, x_n^{(t)}]^T$;

5) Calculate the fitness value in $P_t$ and determine the optimal nest position. If the maximum number of iterations or the optimal nest position is not reached, return to step 3 continue to search, otherwise output the optimal nest position $x_b^{(t)}$;

6) After obtaining the corresponding value of the optimal position $x_b^{(t)}$, it is input into the BP neural network to establish a prediction model, and the model is used to test the defect data set to be predicted.

The overall flow chart of CS-BP is shown in Figure 2:
4. Experimental results and analysis

4.1. Experimental data

The experimental data used in this paper is the MDP[7] data package provided by NASA. Four data subsets of MDP: PC1, CM1, JM1, KC3 were used in this experiment, as shown in Table 1:

| Data source | Language | Size  | Number of modules | Number of defects |
|-------------|----------|-------|-------------------|------------------|
| CM1         | C        | 47K   | 505               | 48               |
| JM1         | C        | 722K  | 10878             | 2102             |
| KC3         | Java     | 29K   | 2107              | 325              |
| PC1         | C        | 99K   | 1107              | 76               |

Taking PC1 as an example, there are 1107 modules, 41 software metrics attributes and one-dimensional labels of whether it is defect {y, n}. Use the CFS algorithm in Weka tool to extract key attributes. The key attributes extracted by PC1 are shown in Table 2:

| Data source | Key attributes                               |
|-------------|----------------------------------------------|
| PC1         | LOC_BLANK, LOC_COMMENTS, LOC_CODE_AND_COMMENT, |
|             | HALSTEAD_CONTENT, NORMALIZED_CYLOMATIC,      |
|             | LOC_EXECUTABLE, PARAMETER_COUNT              |

In order to facilitate the input and output of the network, this paper transforms the defective {y, n} tag into a numerical type, and sets y as 1 for defects, and N as 0 for no defects and then normalize other attributes, as shown in formula (6):

\[
\tau'(A) = \frac{\tau - \tau_{\text{min}}}{\tau_{\text{max}} - \tau_{\text{min}}}
\]  
(6)

In formula 6, \(\tau'(A)\), \(\tau\), \(\tau_{\text{max}}\), \(\tau_{\text{min}}\) represent the normalized value, original value, maximum value and minimum value of attribute A respectively.

The error square formula of neural network used in this paper is also the fitness function of CS algorithm, as shown in formula (7):
In the formula 7, \( q \) is the dimension of output, \( h \) is the \( h \)-th learning sample, \( E_{h,p} \) represents the expected output value, \( O_{h,p} \) represents the actual value, and \( \mu_h \) represents the square of error in network learning.

### 4.2. Experimental evaluation criteria

In order to verify the learning effect of the optimized network model, this paper randomly selects 500 records from each data set, and uses the commonly used 10 fold CV method. In this paper, accuracy, recall, precision and F value[8] are used to evaluate the prediction ability of the model. These evaluation indexes need cross matrix, which is shown in Table 3:

| Actually Wrong situation | Defect prediction results | Module with error | Module without error |
|--------------------------|--------------------------|-------------------|---------------------|
| Module with error        | T1(The right example)    | F0(Negative examples of mistakes) |
| Module without error     | F1(Positive examples of mistakes) | T0(Correct negative examples) |

#### Table 3. Binary classification cross matrix

#### Table 4. Evaluating indicator

| Indicator name                                      | Expression                                      |
|-----------------------------------------------------|-------------------------------------------------|
| Accuracy refers to the ratio of the number of records correctly predicted by the network to the total number of records. | \[ \text{accuracy} = \frac{T1 + T0}{All} \] |
| The precision rate represents the ratio of the number of records predicted to be defective and actually defective to the total number of records predicted to be defective. | \[ \text{precision} = \frac{T1}{T1 + F1} \] |
| Recall represents the ratio of the number of records predicted to be defective and actually defective to the number of records actually defective | \[ \text{recall} = \frac{T1}{T1 + T0} \] |
| \( F \) is the harmonic mean of precision and recall | \[ F = \frac{2}{\frac{1}{\text{precision}} + \frac{1}{\text{recall}}} \] |

### 4.3. Experimental results and analysis

The experimental results are shown in figure 3-6:
4.4. Conclusion

The experimental results show that the evaluation indexes of CS-BP algorithm are better, and the prediction effect of other traditional prediction methods is relatively poor due to some limitations. For example, the structure of BP neural network is selected by experience and depends too much on the initial parameters, so the prediction ability is relatively weak; the evaluation indexes of SVM algorithm are relatively balanced, but the generalization ability is not ideal, which may lead to data over fitting and rely on effective parameter selection methods, so the overall prediction effect is not ideal compared with CS-BP algorithm; J48 algorithm is sensitive to the number of learning samples, and needs to traverse the data set many times in the process of running, when there are many training samples, the efficiency is low, and the overall prediction effect is not as good as CS-BP. In this paper, the global optimization ability of CS algorithm is used to improve BP algorithm. The experimental results show that the evaluation indexes of CS-BP algorithm are higher than those of BP, SVM and J48 algorithm. However, the basic CS algorithm is slow in the late search, so the future research work is to further improve the efficiency of CS in the late search.

References

[1] Yang Bo, Wu Ji, Xu Luo, Bi Kao, Liu Chao. An Approach of Modeling Software Testing Requirements and Generating Test Case[J].Journal of Software,2014,37(03):522-538.
[2] Zhang Ce, Cui Gand, Liu Hongwei, Meng Fanchao,Fu Zhangchuan. Journal of Harbin Institute of Technology [J].Jou,2014,46(05):51-58.
[3] Wang Qing,Wu Shujian,Li Mingshu. Software Defect Prediction[J]. Journal of Software,2008(07):1565-1580.Xin-She Yang and Suash Deb. Engineering optimisation by cuckoo search[J]. Int. J. of Mathematical Modelling and Numerical Optimisation, 2010, 1(4) : 330-343.
[4] Xin-She Yang and Suash Deb. Engineering optimisation by cuckoo search[J]. Int. J. of Mathematical Modelling and Numerical Optimisation, 2010, 1(4) : 330-343.
[5] M. Fast and M. Assadi and S. De. Development and multi-utility of an ANN model for an industrial gas turbine[J]. Applied Energy, 2008, 86(1) : 9-17.
[6] Abhishek Pandey et al. General Regression Neural Network and Radial Basis Neural Network for the Estimation of Crop Variables of Lady Finger[J]. Journal of the Indian Society of Remote Sensing, 2012, 40(4) : 709-715.
[7] Chapman M,Callis P,Jackson W. Metrics data program [EB/OL]. http://mdp.ivv.nasa.gov.
[8] Yu Anlei, Pi Dechang. Software defect prediction model based on PSO-BP [J]. Computer Engineering and Applications,2013,49(07):64-67.