(IJIRSE) International Journal of Innovative Research in Science & Engineering
ISSN (Online) 2347-3207

Improving The Fault Prediction In Oo Systems Using ANN With Firefly Algorithm

Bhavya G
Asst. professor,Dept.of ISE,BMSIT
Bangalore,India.
bhavya20.g@gmail.com

***Abstract*— Assessing the quality of the software is very difficult since most of the software quality characteristics like Capability, Reliability, Security, Performance etc. are not directly measurable. Several approaches had been proposed to predict the quality of the software. In this paper the quality of object oriented system is predicted. The prediction model is constructed using Artificial Neural Network. To train the ANN various algorithms has been proposed. To provide maximum accuracy within limited number of iterations, ANN is optimized using firefly algorithm. Firefly algorithm is the evolutionary algorithm based on the flashing light characteristics of the fireflies. The prediction accuracy of ANN with Firefly algorithm, is compared with the accuracy of ANN trained using Gradient descent algorithm. The result shows that ANN trained using Firefly algorithm predicts maximum accuracy.**

***Keywords*— Artificial neural network, Firefly algorithm, Software metrics**.

I. INTRODUCTION

The Software quality is the degree to which a system meets specified requirements and specifications. As the dependency on the software is increased there is a need for quality software [1]. Software quality could be affected by factors like faults, approaches, tools, schedule pressure, time limit, technology etc. To achieve a good software quality the faults in the software system as to be identified and removed as early as possible. To predict the faults resides in the system a prediction model is been developed. Software fault prediction model is used to predict the faults in software modules. There are various techniques for fault prediction which depends on historical data. Some of these techniques are logistic regression by Basili et al.1996, classification trees by Gokhale and Lyu 1997, Khoshgoftaar and Seliya 2002 [2], Selby and Porter 1988, neural networks by Khoshgoftaar and Lanning 1995 effective for predicting faults have a large input data, genetic algorithms by Azar et al.2002.Software quality prediction model is based on the basic components of the systems. Software metrics is the basic component that is used in fault prediction models to predict faults in object oriented systems in this paper. Form various researches it is found that Artificial neural network (ANN) provide best accuracy than other prediction model [3] [4] [5] [6]. Artificial neural network is a type of network where the node of that network sees as the artificial neurons which are a computational model inspired in the human brain. These neurons solve the complex problem by work together and highly interconnected to each other. The Parameters of ANN are Inputs, Interconnection weights, summing function, activation function. The accurate prediction model is obtained by training the artificial neural network using training algorithms which is based on adjusting the parameters of ANN. The Key parameter to well train the ANN is the weight associated with the connection between the layers. In this paper weight is optimized to train the ANN.

ANN can be trained using many training algorithms. The basic training algorithm is the gradient based algorithm such as Back Propagation (BP) algorithm which is widely accepted training algorithm because of ease of implementation and its simplicity. The concept of the BP algorithm is to minimize the error function. But the disadvantages of the BP algorithm are it get struck in the local optimum and converge in more number of iterations [7].

Genetic algorithm is the another technique used for optimizing the ANN. Genetic algorithm is introduced by John Holland, which is a type of evolutionary algorithm inspired by the concepts of genetics and evolution. This algorithm converges faster than the BP algorithm [8] but it is still slow and complex because of its operators like crossover and mutation [9]. Our focus in this paper is to optimize the weight parameter of ANN using Firefly algorithm which is a family of Evolutionary algorithm.

The aim of our research is to train the ANN using firefly algorithm to get a prediction model and compare the prediction accuracy with the model obtained from ANN trained by Gradient descent algorithm. This paper is organized as follows: section 2 gives introduction about the artificial neural network and presents training the ANN, section 3 describes firefly algorithm, section 4 comparison results of ANN-GD and ANN-FA, section 5 gives the conclusion and the future work can be done.

II. ARTIFICIAL NEURAL NETWORK

ANN is the simplified model of human nervous system. The key element of this ANN is the large number of neurons present and act as basic processing elements. The artificial neural networks (ANN) consist of a number of interconnected neurons, each of which has a number of inputs, outputs and a transformation function. These neurons are arranged in the form of three types of layers: input layer, output layer and hidden layer(s). The input layer receives the values from the input variables and the output layer gives the output of the network. Between the input and output layer is a hidden layers. There can be any number of hidden layers between input and output layers. Each neuron is connected to another neuron through weights. These neurons work together to solve complex problems. Similarly, an Artificial Neural Network can be configured to solve a number of difficult and complex problems. ANNs used for wide variety of applications in diverse areas including functional approximation, nonlinear system identification and control, pattern recognition and pattern classification, optimization , English text pronunciation, protein secondary structure prediction and speech.

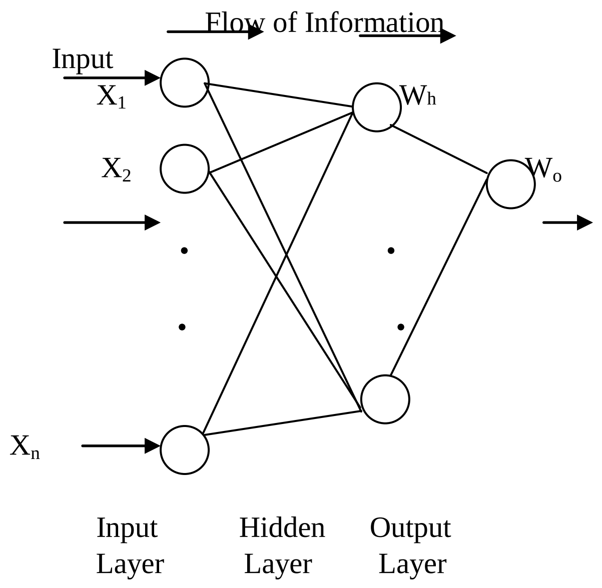

Figure 1. ANN Architecture

ANN’s can be divided into two major categories based on their connection topology

- Feed forward neural network
- Feedback neural networks

Feed-forward neural networks allow the signal to flow in the forward direction only. Feed-forward networks, based on their architectures, are further classified into different categories i.e. Multilayer Perceptrons (MLPs), Counter-propagation Networks (CPN), Cerebellar Model Articulation Controller (CMAC) and Radial Basis Function Networks (RBF Nets).Feedback neural networks do not impose any such constraint on the flow of the signal in the network. The signal from a neuron in a layer can flow to any other neuron whether it be preceding or succeeding layers. The interconnections between neurons in the same layer are called lateral connections. Such neural nets are very much suitable for systems where output of the system under consideration is a function of not only the inputs but past outputs and inputs also. However, the drawback with these networks is that these are complex and are difficult to implement.

There are three layers in ANN architecture: input layer, hidden layer, output layer as shown in Figure 1.The input to the input layer is raw information which is then processed and sent to hidden layer. In the hidden layer the data is multiplied with the weight associated with that link and all the results is summed and sent to output layer. The output layer again processes it same as hidden layer and give a prediction model. The nodes in the input layer are passive because they just replicate and pass the information to the nodes in the hidden layer. The hidden nodes are active, since they do some process on the information it gets from the input nodes and map it to the other domain and sent to output layer nodes. The output nodes are also active since it also performs some computation on the data which it gets.

Table 1: Metrics List

S. No.	METRICS	ABBREVATION
1	WMC	Weighted method per class
2	DIT	Depth of inheritance tree
3	NOC	No. of classes
4	CBO	Coupling between objects
5	RFC	Response for class
6	LCOM	Lack of cohesion between methods
7	Ca	Afferent couplings
8	Ce	Efferent couplings
9	NPM	No. of methods
10	LCOM3	Lack of cohesion between method
11	LOC	Line of code
12	DAM	Data access metrics
13	MOA	Measure of aggregation
14	MFA	Measure of functional abstraction
15	CAM	Cohesion among methods
16	IC	Inherent coupling
17	CBM	Coupling between methods
18	AMC	Average method complexity
19	MAX_CC	Maximum McCabe's Cyclomatic Complex
20	AVG_CC	Average McCabe's Cyclomatic Complex

The input to the input layer in our work is the object oriented metrics. Input or the Data sets are obtained from publicly available software engineering data repositories (Metric Data Program NASA) .The Metrics in the datasets describe projects which vary in size and complexity, development process, etc. Each dataset contains 20 software metrics, which describe product's size, complexity and some structural properties [10]. List of the metrics used in this project is shown in Table 1.

The representation of the active node is shown in Figure 2. Active nodes used in hidden and output layers of the ANN. Each input to the active nodes is multiplied by a weight associated with that link and all results are then added produces a single value. The result then passed to the sigmoid function which performs computation on the input and gives the response.

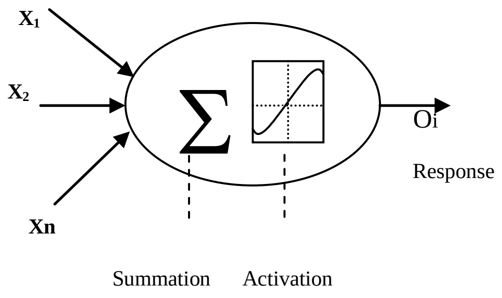

Figure 2.Activation node

2.1. Training ANN

Training is a process by which the parameters of a neural network are adjusted through a process of stimulation by the environment in which network is embedded. Training rule is a prescribed set of well-defined rules for the solution of learning. The training rule in the neural network is to minimize the error that is the different between the expected and the observer output of the neural network. To minimize the error function weight parameter of the ANN is adjusted using weight training. The nodes of the network in one layer are interconnected with nodes in other layer to perform some task and this connection is associated with weights.

III. FIREFLY ALGORITHM

Firefly algorithm is a bioluminescence process which is based on the flashing light behavior of the fireflies which was proposed by Yang. Firefly algorithm uses the random numbers corresponds to the random movement of the fireflies which is one of the main advantage of the firefly algorithm.

There are three rules that should be used while implementing the firefly algorithm.

i) All fireflies attract towards other fireflies which is brighter regardless of their sex.
ii) The attractiveness increases as the brightness with decrease in the distance. That is attractiveness is proportional to the brightness. If no brighter firefly then move randomly.
iii) The Objective function's value of the given problem determines the brightness. For maximization problems, the light intensity is proportional to the value of the objective function.

The pseudo code of the firefly algorithm [11] is given below:

```
Begin
        1) Objective function  $f(x)$;
        2) Initial population of fireflies is gnerated
        3)Light intensity $I$ is formulated so that it is associated with  $f(x)$
         (for example, for maximization problems, $I \alpha f(x)$ or $I=f(x)$);
        4) Define absorption coefficient $\gamma$
        While (t<MaxGeneration)
        for i=1:n (all n fireflies)
                for j=1:n (n fireflies)
                        if (Ij > Ii),
                        move firefly i towards j;
                        end if
                Vary attractiveness with distance r via exp($-\gamma r$);
                  new solutions obtained is evaluated and accordingly update light intensity;
                end for j
        end for i
        give the rank to the fireflies and based on rank find the current best;
        end while
        Post-processing the results and visualization;
End
```

3.1 ANN-FA

Train the ANN using Firefly algorithm involves the following steps. The weight parameter of the ANN is to be optimized so Weights associated with the links corresponds to the position the fireflies. Consider there are n number of weights to be optimized so search space is of n-dimensions.the fireflies move around this search space and come up with a optimized weights. The optimized weights are then assigned to the ANN to get the accurate prediction model. After a desired number of iterations the position of the firefly is considered as the best solution and yields an optimized weight. The steps for a ANN-FA for optimizing this ANN is as follows:

Pseudo code of ANN-FA algorithm

- Define the architecture of ANN i.e., number of input, hidden and output nodes of each layers.

- Identify the objective function i.e., the light intensity in the maximization problem
- Initiate a number of fireflies with n dimension where n is the number of weights that need to be optimized for the ANN.
- For each iteration repeat these steps for each firefly.
- Move the fireflies towards the brighter one. If no brighter firefly then all fireflies take a random movement.
- Update the weights of all moved firefly and find whether this is the best.
- Update the weight of the firefly and check if it is a global best.
- Repeat these steps till iterations are completed.
- The global best weights of firefly are final result and map this with the weights of ANN to get better prediction accuracy. The overall system architecture is as shown in Figure 3.

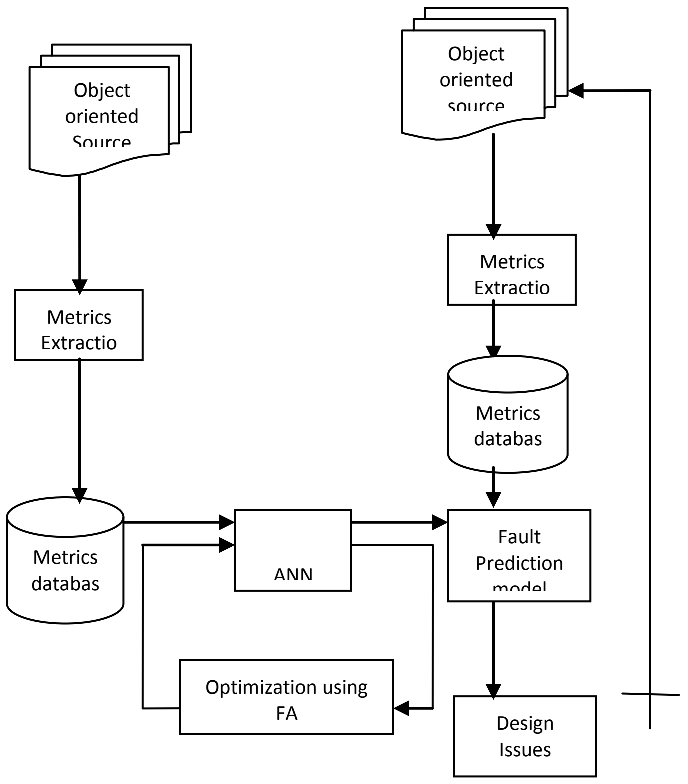

Figure 3.System architecture

IV. PERFORMANCE EVALUATION

The ANN-FA is tested with various dataset, and the result of it is analyzed with the ANN-GD. Table 3 provides the Maximum(max.),Minimum (min.), Mean, median, standard deviation (Std.Dev.),Percentile values for each object oriented metrics.

The firefly algorithm parameters are the amount of fireflies (n), the number of generations (G), the light absorption coefficient (γ), the randomization parameter (α) and the attractiveness value (β_0).Generally, the combination factor(nG) determine the amount of search (candidate solutions) in the solution space conducted by the firefly algorithm. This factor is directly related with the size of the problem considered. The high value of this combination usually increases the probability of getting the best solution but requires longer computational time and resources. Table 4 gives the settings of firefly algorithm parameters.

Table 4: FA parameters setting

FA parameters	Values
nG	20*100
Γ	1.0
A	0.5
β0	0.2

Table 5 gives the prediction accuracy values of ANN-GD and ANN-Firefly. The accuracy of the ANN-Firefly for the four different projects is compared with the accuracy of the ANN-GD for the same projects. Table 8.3 shows that ANN-Firefly has high accuracy compared to ANN-GD given the same dataset.

Table 5: Comparison of Prediction accuracy

Project	ANN-GD	ANN-Firefly
Arc	64.529	88.462
Camel	93.510	94.690
Intercafe	77.778	85.185
Tomcat	74.475	84.033

The graph as in Figure 4 shows improved fault prediction accuracy using the ANN-Firefly than the ANN-GD for the four different projects thus showing maximum improvements in the accuracy rate.

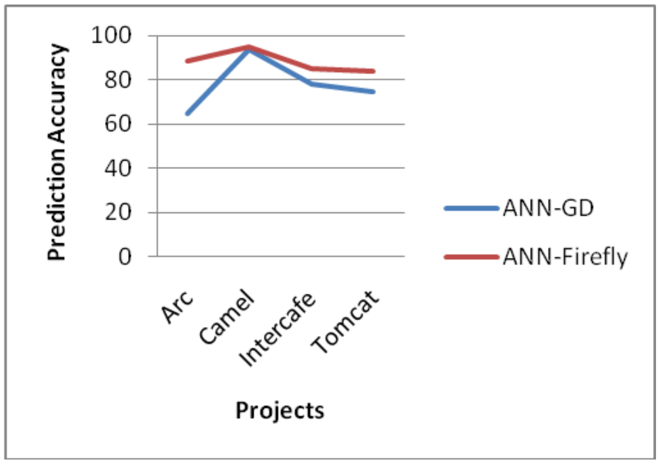

Figure 4: Comparison graph for ANN-GD and ANN-Firefly

Table 3:Statistical distribution values of metrics and bug data

Metric	**Min**	**25Q**	**Mean**	**Median**	**75Q**	**Max**	**SD**
WMC	0	3	11.34	6	13	252	15.79
DIT	0	1	1.76	1	2	6	1.15
NOC	0	0	0.38	0	0	31	2.04
CBO	0	2	8.12	5	10	185	11.89
RFC	0	7	28.50	16	33	511	38.43
LCOM	0	0	129.96	6	40.75	29258	906.75
CA	0	0	3.99	1	3	184	10.36
CE	0	0	2.15	0	2	36	4.52
NPM	0	2	9.43	5	11	231	14.02
LCOM3	0	0.63	1.10	0.88	2	2	0.67
LOC	0	23	258.61	83	244.50	7956	528.35
DAM	0	0	0.60	1	1	1	0.47
MOA	0	0	0.88	0	1	24	1.87
MFA	0	0	0.30	0	0.70	1	0.39

CAM	0	0.29	0.48	0.44	0.67	1	0.26
IC	0	0	0.30	0	1	4	0.56
CBM	0	0	0.53	0	1	19	1.47
AMC	0	4.34	19.39	10.40	22.96	894.5	37.10
MAX_CC	0	1	3.49	1	4	95	5.74
AVG_CC	0	0.67	1.14	1	1.33	10	0.89
BUG	0	0	0.12	0	0	6	0.45

V. CONCLUSION AND FUTURE WORK

There can be many number of faults resides in the software project. It is very important to detect and remove the faults as early as possible in the lifecycle of the project development. Various techniques are available for this purpose in the literature, but previous research has shown that the object oriented metrics are useful in predicting the fault proneness of classes in object oriented software systems.

To predict the quality of the object oriented software the application of artificial neural networks will be an efficient method to estimate software quality in such systems. Among the different soft computing techniques ANN possesses advantage of minimal number of iterations. In addition, it is among one of the emerging technique in software development and plays a crucial role in predicting software quality. Predicting the software quality using ANN requires large set of data so algorithm is required to train ANN. Firefly algorithm is an evolutionary optimization used to optimize such model. Accurate predictions obtained from such a good model leads to improve the quality of the software.

The following enhancements can be made in the future one is Neural network integrated with other artificial intelligence technologies, can be used to predict the faults in object oriented software. Second is Integration of firefly and artificial neural network can be applied to other problems with different datasets like Iris dataset, Wine dataset, medical dataset etc,.

REFERENCES

[1] Bellini P. (2005), “Computing Fault-Proneness Estimation Models”, 10th IEEE International Conference on Engineering of Complex Computer Systems (ICECCS’05), China, pp. 205-214.
[2] Khosgoftaar TM, Gao K,Szabo RM (2001) An Application of zero-inflated poisson regression for software fault prediction, Proceedings of 12th international symposium on software reliability engineering,pp:63-73.
[3] R.C.Lacher,P.K.Coats, S.C.Sharma, and L.F.Fant, “A neural network for classifying the financial healt of a firm,” Eur.J.Oper.Res.,vol.85,pp. 53-65,1995.
[4] K.Y.Tan and M.Y.Kiang, “Managerial application of neural networks: The case of bank failure prediction, “manage.Sci., vol.38,no.7,pp.926-947,1993.
[5] I.Guyon, :Application of neural networks to character recognition”, Int. J. Pattern Recognit.Artif. Intell., vol.5,pp.353-383,1991
[6] S.Knerr, L.Personnaz, and G.Dreyfus, “Handwritten digit recognition by neural networks with single-layer training”,IEEE Trans. Neural Networks,vol.3,pp. 962-968,1992.
[7] T.Petsche,A.Marcantonio,C.Darken,S.J.?Hanson,G.M.Huhn, and I. Santoso, :An autoassociator for online motor monitoring,” in industrial Applications of Neural Network,F.F.Souline an p.Gallinari,Eds,Singapore:word scientific,1998,pp.91-97.
[8] E.B.Barlett and R.E.Uhrig,”Nuclear power plant status diagnostics using artificial neural netwoks,”Nucl.Technol.,vol.97,pp.272-281,1992.
[9] J.C.Hoskins,K.M.Kaliyur, and D.M.Himmelblau, “incipient fault detection and diagnosis using artificial neural networks,” in proc. Int. Jint Conf. Neural Networks,1990,pp.81-86.
[10] Promise dataset for object oriented systems-metrics and bug data,http://www.Promizedata.org.
[11] Aphirak khadwilard,Sirikaran chansombat,Thatcai Thepphakorn,”Application of Firefly Algorithm and its Parameter setting for job shop scheduling.