Received: March 1, 2018

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

Automization of Object Oriented Design Using Back Propagation Neural Network for Better Maintainability of Software

Mallaiah Vorsu[1]* **Vinaya Babu Arramraju[2]** **Sujatha Bhanothu[3]**

[1]*Acharya Nagarjuna University, India*
[2]*Stanley College of Engineering and Technology for Women, India*
[3]*University College of Engineering Osmania University, India*
* Corresponding author's Email: mallaiah.vorsu@gmail.com

Abstract: Object Oriented Program (OOP) provides the way to reuse the program by pre-implementing functionalities in the software. It is difficult to develop the object oriented software, which is important in the computer programing. For OOP, modularization mainly depends on the class. There are many methods present for the assigning responsibilities, but most of them are based on the human for decision making. In this research, back propagation neural network (BPNN) was used to provide the solution to object oriented design of the software. The Cinema Booking System (CBS) was taken as the input documentation and Formal Concept analysis (FCA) then found the relationship of the element in the lattice manner, after that the relationship was set with each other. The result showed that the proposed system outperformed the existing system and also the design made manually.

Keywords: Back propagation neural network, Cinema booking system, Formal concept analysis, Object oriented program.

1. Introduction

This is the difficult task to find the class responsibilities in the OOP and this is crucial task during early analysis, design phases, and maintaining phase, when new responsibilities have to be allocated to classes, or present responsibilities have to be changed [1, 2]. There are lots of existing methods for the manual interpretation and decision making. A lot of time is needed to maintain the existing software to analyse and comprehend the available source code [3]. Some tools are available to support Class Responsibility Assignment (CRA) which can be used for analysing and designing OOP, was designed to provide a cognitive toolkit for designers and developers [4]. Designing the object-oriented software (OOS) is the complex process and in the initial steps it's included analysing the class candidates and allocating responsibilities of a system to them [5]. This type of initial design is applied in more advanced Object-oriented mechanisms like interfaces, design patterns, inheritance or even architectural styles [6].

The poor allocation of responsibilities to classes hardly overcome by inheritance or design patterns. Successful accomplishment of software maintenance is based on the how much information taken by the software maintains [7]. There are some metaheuristic algorithms like Particle swarm optimization (PSO), Ant colony optimization (ACO), Genetic algorithm (GA) used by the researchers for CRA problems [8], [9]. Most of the other methods on software remodelling is based on the two genetic algorithms namely, Non-dominated Sorting Genetic algorithm (NSGA-II) and hill-climbing algorithm [10]. In this context, FCA was applied to the program to extract the initial solution from documents and model. The BPNN helped to improve the initial solution and gave the best possible solution through iteration. The BPNN uses the delta rule or gradient descent technique to find the weight value with minimum error and the weight value having the minimum error is considered as a solution. BPNN compute every

layer in the network and also the hidden layer, then comparing those errors with real value to find the best solution. Then the dependency between the elements of design model was measured and this method was compared with the existing methods. The result clearly showed that BPNN performed better compared to the existing methods.

The paper is organized as follows. In section 2, the latest research paper of OOP is surveyed. In section 3, the proposed BPNN method is briefly explained along with pseudo code. The experimental comparison between existing and proposed methodology is presented in the section 4 and Conclusion is given in section 5.

2. Literature review

J. de AG Saraiva, M.S. De França, S.C. Soares, J. C.L. Fernando Filho, and R.M. de Souza [11] studied the OOS maintainability (OOSM) through years and they proposed the metrics categorization to improve the process. The seven categories and seventeen subcategories were found and the family of OOSM was generated based on this metrics categorization. These categorize also represents the scenario of OOSM metrics adoption. The classification was obtained as 90% with the 99% level of confidence using the Wilcoxon Test. The depth investigation like finding the other categories, is needed for better performance.

Gupta and Chhabra [12] developed a dynamic analyser tool using aspect-oriented programming, collecting a run-time data to analyse of Java software dynamically for the processing of calculate the dynamic cohesion of the proposed system. This technique is executed in the 20 Java software's and this shows that this method is more accurate and useful, while compared to the existing cohesion metrics. The change-proneness of classes are identified very effectively by this method, which is found by evaluating the proposed technique by using the source code APIs of Java Development Kit (JDK). The alternate method has to be used to select the weight assigned to the different relations. The weight selection technique is not efficient, so alternative weight technique is needed.

L. Kumar and S.U. Rath [13] designed a predicting maintainability model with the help of three AI techniques such as particle swarm optimization (PSO), hybrid approach of functional link artificial neural network (FLANN) with genetic algorithm (GA), and clonal selection algorithm (CSA), i.e., FLANN-CSA (FCSA), FLANN-Genetic (FGA and AFGA), FLANN-PSO (FPSO and MFPSO). The maintainability on the two case studies were predicted by this AI techniques, and the two study case are namely; (1) User Interface System (UIMS) and (2) Quality Evaluation System (QUES). The feature reduction techniques such as principal component analysis (PCA), and rough set analysis (RSA), used for predicting maintainability. The features reduction techniques were very effective while using FLANN-Genetic, which is shown in the result. This technique was not tested with the service oriented system and open source software.

Amarjeet and J.K. Chhabra [14] proposed a method to increase the quality of modularization of an OOS with the minimum possible class transfers between existing packages of original software modularization using multi-objective optimization approach. The widely-accepted multi-objective evolutionary method NSGA-II was used for the optimization technique. The experimental research clearly demonstrated the effectiveness of this method by improving the quality of software using very less movement of class in the existing package. If the use of dynamic and semantic relation for connection strength calculation is added, then it helps to provide better result.

G. Kakarontzas, E. Constantinou, A. Ampatzoglou, and I. Stamelos [15] focused on the technical problems and proposed a new metric simplifying the reuse of OOS based on the popular suite for object-oriented design. The linear regression was applied on a several of OOS java projects that used to derive the new metrics and this technique was compared with others. The frameworks provide more flexibility and can be used to attain the reusability of classes, which is important for future reuse. This works only focus on the characteristics of the source code and the other factor relating to the comprehensibility of the source code is also important.

The limitations of above mentioned researches is reduced with help of proposed BPNN technique.

3. Proposed method

In this research, the adaptive neural network has been applied for the OOP, which helps to improve the initial solution and provide better solution. The FCA has been used before the input is given to the neural network. This section gives the information about the proposed method. The overview of the proposed system is shown in Fig. 1.

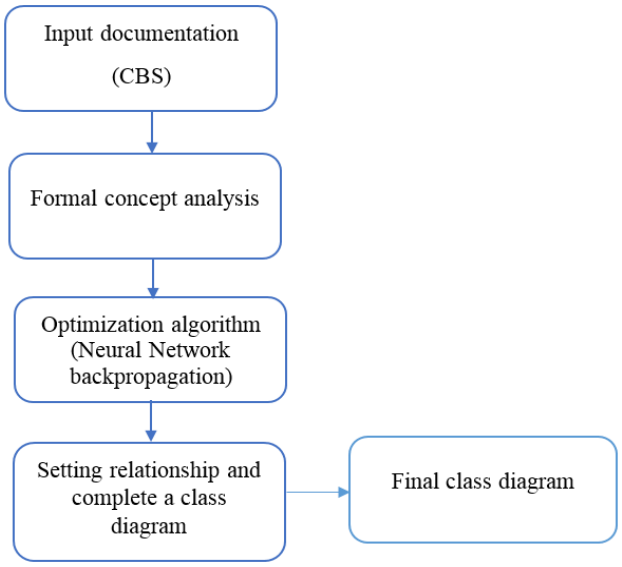

Figure. 1 Overview of the method

3.1 Optimization metric

The object-oriented metrics is the efficient way to measure the software quality design. Generally, there are three categories of object-oriented metrics: coupling between classes, inner cohesion of the class and complexity of design and these three metrics have been used for optimization. The weighted combination of four criteria has been computed to calculate the cost function. The cohesion metric has to be maximized and the other three should be minimized. The maximization has been done also for cost function and hence relation composed of combination of minimization metrics and reverse of maximization metric.

$$CostFunction(D) = w_1 cop(D) + w_2 Complexity(D) + w_3 ClassSizeStandardDeviation(D) + \frac{w_4}{Coh(D)} \quad (1)$$

Where,

$$w_1 + w_2 + w_3 + w_4 = 1 \quad (2)$$

In Eq. (1), Complexity (D), Cop (D), Class Size Standard Deviation (D) and Coh (D) show complexity of design, coupling, standard deviation of the class size, and cohesion. The right choice of weights depends on design priorities and also they can be chosen empirically.

3.2 Formal Concept Analysis

The FCA is applied on the input data of the documentation in the analysis part. The FCA is used to extract the initial candidate class using review concept lattice and this is done semi-automatically by expert opinion. This method helps to provide the ability to model the conceptual relationships using lattice diagrams. The formal things and formal features are used and the relationship between them are created. For example, the cases are considered as a formal things and Responsibilities are considered as a formal feature in responsibility assignment. These concepts can be candidate classes and each row is a use case and each column is a responsibility in the system and each mark shows the relation between them. Lattice diagram for this context is shown in Fig. 2 and each node in this lattice is considered as concepts or candidate class.

The initial assignment is assigned with the help of a domain expert and this initial assignment is used as input of the next phase. In the next phase, the proposed method tries to optimize initial solution using a BPNN.

3.3 Back propagation neural network

An artificial neural network is composed of computational processing elements with weighted connections and also used the feed forward multilayer perceptron network and the back propagation training algorithm. The input layer has one neuron for each of the input variable (domain measures$-D_j, j = 1, \ldots . p$). The one hidden layer is used here and two neurons in the output layer, one for each class. The output unit with the greatest value specifies the class selected by the network. The output unit, which have the high value indicates the class selected by the network. The network learns from the connection of weight vector and reduce the

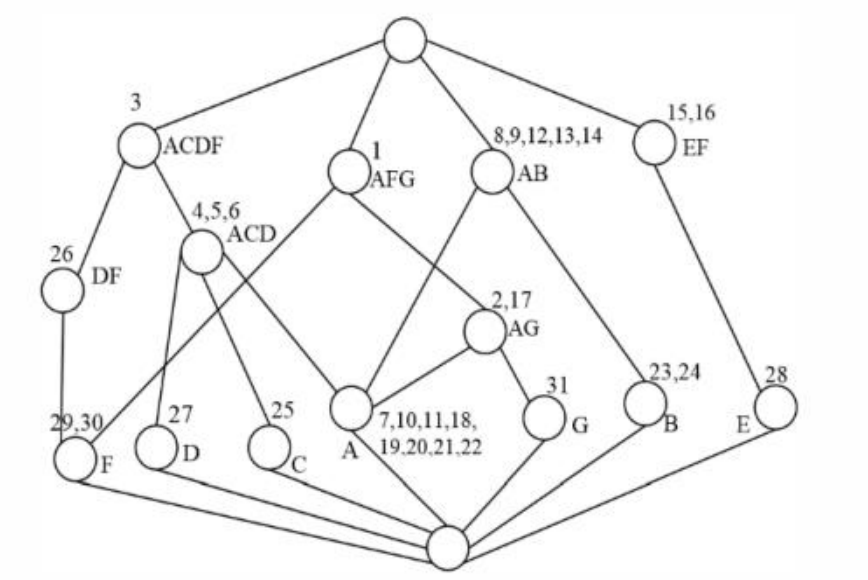

Figure. 2 Concept lattice for CBS system.

sum of squared errors available in the dataset. The one cycle pass through all training observation of training is the epoch and back propagation learning algorithm is used to train the network. The weights values are updated after fed forward in each observation and various neural networks are tested. The learning rate, number of units in the hidden layer, and the momentum rate are adjusted to determine a preferred combination. The parameters of the neural network are given in the table 1. The activation function is a logistic function with the gain parameter that controls the operations from zero to one and 'tansig' is the activation function used. The network is trained after the weights are updated after each epoch and the number of neurons are tested with different values to find the final value. After training, the network is simulated for the validation data set (Testing phase) and the classification outputs are obtained.

Pseudo code for back propagation neural network

```
Input: ProblemSize, InputPatterns, iterations_max, learn_rate
Output: Network
Network ← ConstructNetworkLayers()
Network_weights ← InitializeWeights(Network, ProblemSize)
For (i = 1 To iterations_max)
  Pattern_i ← SelectInputPattern(InputPatterns)
  Output_i ← ForwardPropagate(Pattern_i, Network)
  BackwardPropagateError(Pattern_i, Output_i, Network)
  UpdateWeights(Pattern_i, Output_i, Network, learn_rate)
End
Return (Network)
```

Where $iterations_{max}$ is the maximum iteration value and $learn_{rate}$ gives learning rate. The $Network_{weight}$ is the weight, which is compared with error value.

3.4 Setting relationship

Some class and their relationship included in the class diagram and every relationship between classes are dependency, association, generalization and abstract/realization. The element may depend on another one, this relationship between the two elements is known as dependency. Let's assume that the two elements are dependent, then change in one element causes the change in other and the association relationship is a connection between classes. M. Bowman, L.C. Briand, and Y. Labiche [8] proposed method is used to set the association relationships between the elements of the design model. For generalization, this is implemented by inheritance in programming languages using the method from H. Masoud, and S. Jalili [17].

4. Experimental study

The proposed method was conducted and measured its performances in this section, and Net Beans were used for this evaluating results. The parameter setting is given and evaluated results are given in detail below.

4.1 Parameter settings

To measure the performance of the proposed method, Cinema Booking System has been used. The

Table 1. Back propagation Neural Network settings

Model No.	No of neurons in three layers	Learning rate η	Momentum rate α	Gain	Epoches
BPNN	6,19,2	0.1	0.4	1	368

Table 2. Case study of CBS

No. of	Attributes	Methods	Responsibilities	Classes	Usage
Cinema Booking System (CBS)	16	14	31	5	39

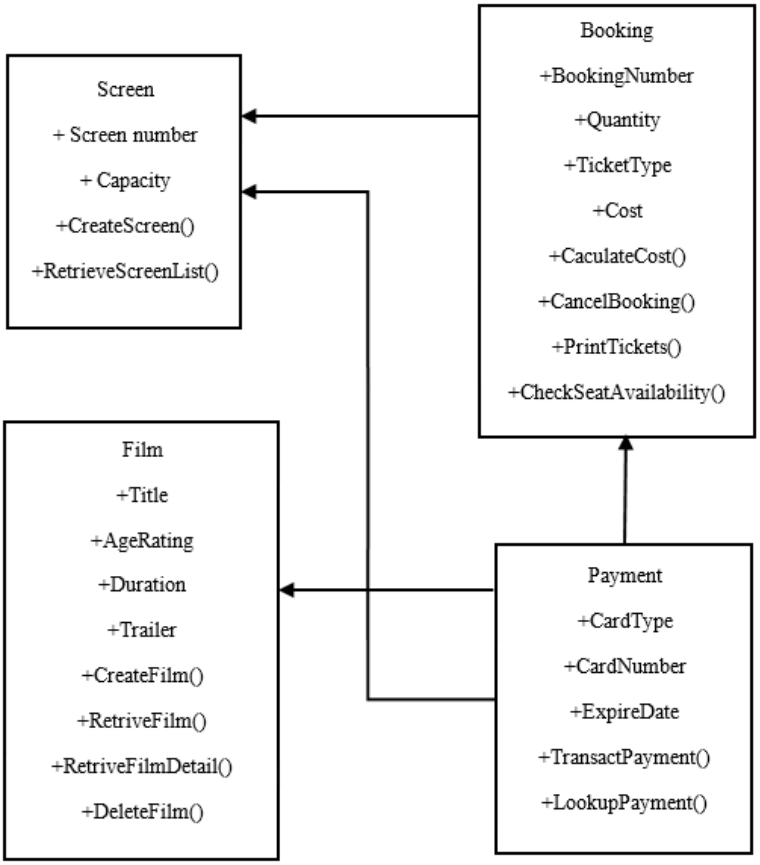

Figure. 3 Design made by proposed system

proposed methods have been compared with ICA-TS and ICA-TS with FCA. The parameter settings for the neural network back propagation is given in the table 1. This method was simulated in the system of specification of 4 GB RAM, 500 GB hard disk, 3.5 GHz processor and Intel i3. The Imperialist Competitive Algorithm (ICA) was implemented in 350 iterations for CBS. The operating revolution in ICA was like operating mutant in GA. The country number for ICA is intended as 100 and the size of Tabu list in TS is set 0.4 of all possible action. Weight for metric value is given in Eq. (2) is set as follows.

$$w_1 = w_2 = w_4 = 0.2 \wedge w_3 = 0.4$$

The value for this weight are obtained experimentally.

The Neural Network is simulated with the given settings and Model III has 368 epoches. They also consist of the different settings of neurons in various layers for evaluating purpose.

4.2 Experimental Result

The BPNN, and other existing methods is applied in the CBS and made the design. The result of this methods are shown in the table 3. Each algorithm was applied for 10 times and the average value is taken. The result of the proposed system is compared with design proposed by the expert designers. The proposed method has the high cohesive value and low in the other three parameters. This is clearly shown in the table that the result of this proposed method are better than design by experts. The design of the result provided by the proposed system is shown in the Fig. 3. This is efficient than the design provided manually. The design done manually by the experts are shown in Fig. 4.

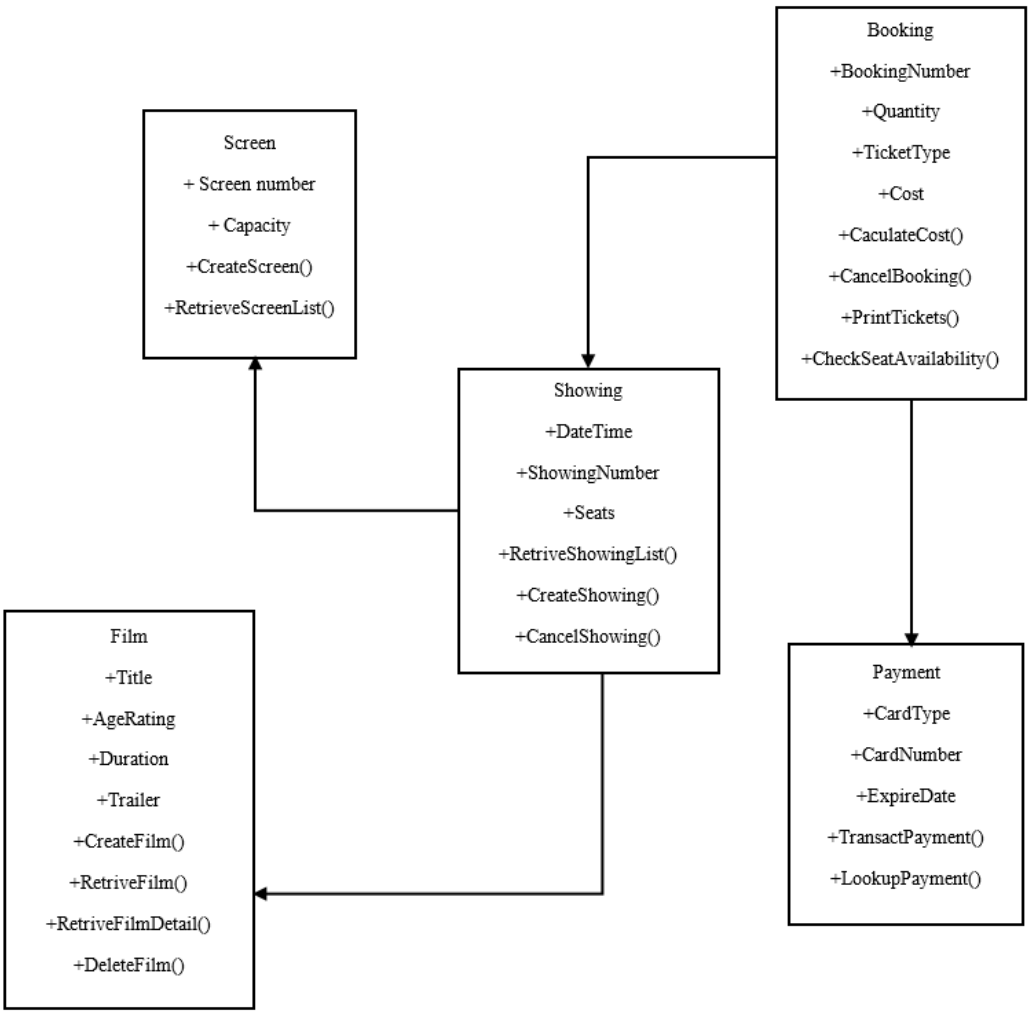

Figure. 4 Design by experts

Table 3. Performance Evaluation

Method		**Criteria**				
		Cost function	**Cohesion**	**Coupling**	**Complexity**	**CSSD**
GA [16]	Best	0.497	0.55045	0.31579	0.29446	0.03148
	Avg	0.526	0.53576	0.3512	0.30468	0.05453
	St.D	0.03	0.01618	0.03935	0.01602	0.04722
ICA [16]	Best	0.497	0.55045	0.2823	0.26221	0.03148
	Avg	0.512	0.5378	0.33481	0.29664	0.03422
	St.D	0.015	0.01291	0.03916	0.01587	0.00822
ICA-TS [16]	Best	0.497	0.55045	0.31579	0.29446	0.03148
	Avg	0.502	0.54827	0.32632	0.29774	0.03148
	St.D	0.012	0.00572	0.03158	0.00983	0
ICA-TS with FCA [16]	Best	0.469	0.54072	0.25	0.15292	0.03099
	Avg	0.473	0.5357	0.25849	0.17834	0.03099
	St.D	0.007	0.01288	0.00594	0.00934	0
Proposed	Best	0.442	0.5682	0.2156	0.12678	0.02972
	Avg	0.442	0.5522	0.2266	0.1426	0.03006
	St.D	0.061	0.014	0.00522	0.00913	0
Expert Design [16]		0.886	0.2787	0.33851	0.25631	0.12592

From the table 3, it is clearly shows that the proposed system performed well compared to the other existing methods. The GA method provided better automatic solution for the system and Imperialist Competitive Algorithm (ICA) algorithm gave improved result than the GA algorithm. The

expert design has the low value; this shows that the system functions is hard to perform manually. The proposed methods show high performance compared to the GA, ICA, Imperialist Competitive Algorithm with Tabu Search (ICA-TS), expert design and ICA-TS with FCA. The second best performance has been provided by the ICA-TS with FCA. The different criteria were measured for these methods and the values are shown in table 3. The different functions are evaluated including cost function, cohesive, complexity, coupling and CSSD. The proposed BPNN find the best weight value by analysing each layer and hidden layer. Basically, BPNN is about understanding the impact of weight and biases in the cost function of the network. The weight of the node is compared with error in order to find the best solution for the system. The weight value with the lower error is taken as the solution. The BPNN having the advantages of high accuracy due to its complex process than the Evolutionary algorithm. The state of art method uses the Evolutionary algorithms to reduce the coupling, complexity and cost function.

The proposed system performance is efficient in terms of cost function, complexity, CSSD, coupling and cohesion. The different metrics as best, average, standard deviation for each methods is measured. It consists of high cohesion value and low value in all other measurement and the average value is taken from the execution of program at 10 times. The graphical representation of performance of the different methods are shown in Fig. 6.

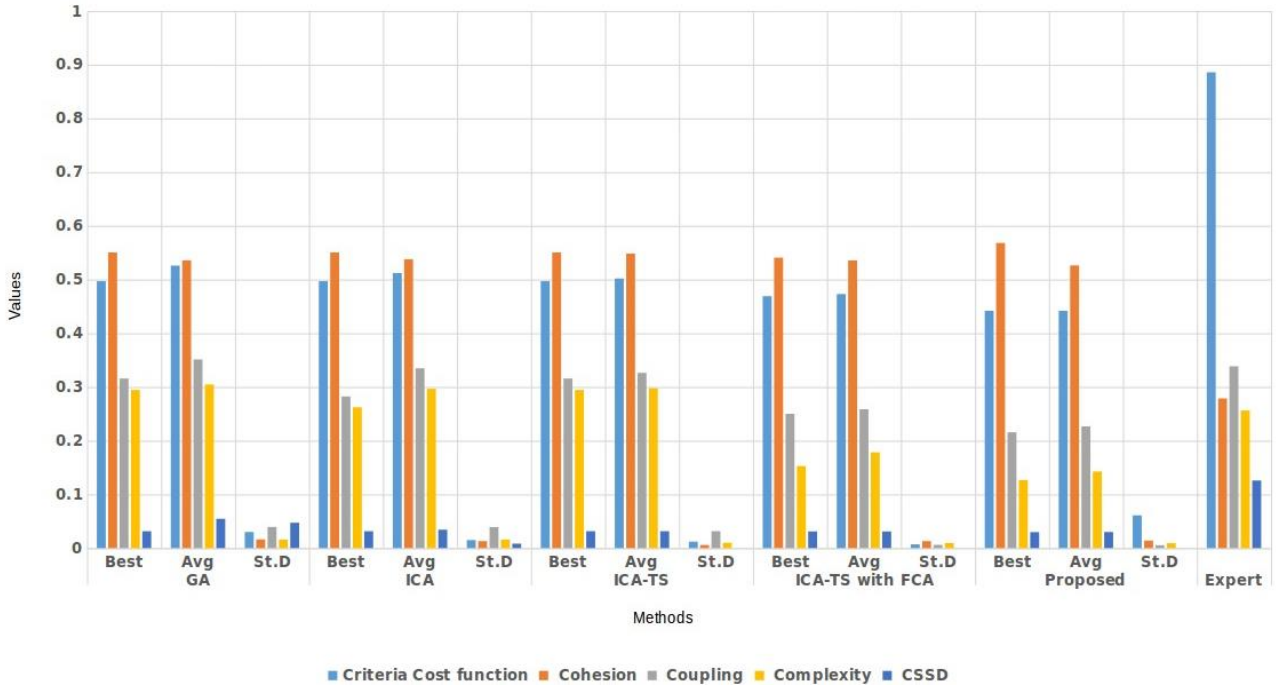

Figure. 5 The performance of the methods

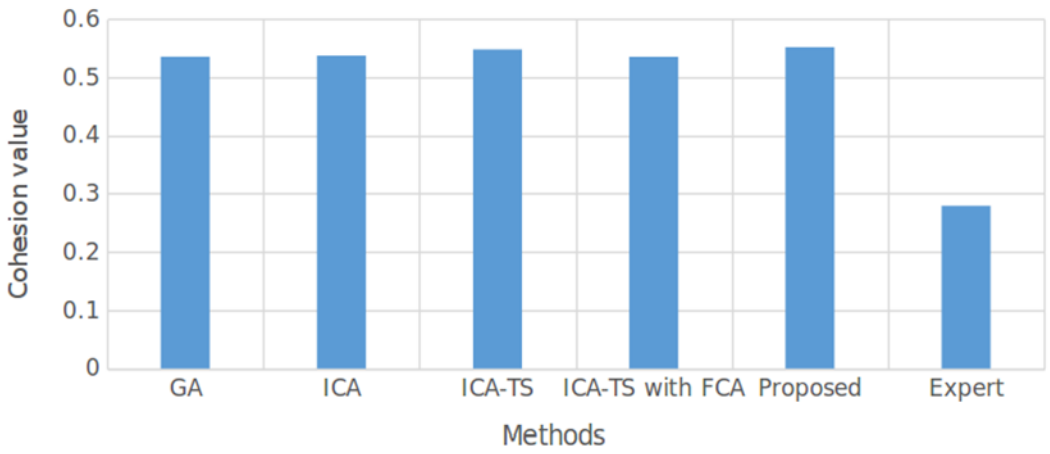

Figure. 6 Cohesion value for different methods

The value of cohesion for the different methods are given in Fig. 5. The Methods with high cohesion is better, due to it has robustness, reusability, reliability and understand ability. The proposed system has more cohesion value of 0.5522 and the second best solution provided in terms of cohesion is ICA-TS of 0.54827. The cohesion value is taken with the average value of the run of programs in 10 times.

5. Conclusion and future scope

The analysis and design of object oriented software is one of the difficult and important task in computer programming. Designing the large number of existing program takes high execution time. In this paper, the BPNN was proposed for the object oriented software. The CBS was taken and given as the input document in the formal concept analysis, which helped to provide the relationship of the element in the lattice manner. The BPNN analysed the system and gave better solution for the design. The relationship between the settings were evaluated and formulated the design and it was compared with the other design made by the expert manually. The performances of the system were evaluated and compared with the other methods such as GA, ICA-TS, and ICA-TS with FCA. The experimental result showed that the proposed system performance better than the existing system. The proposed method achieves the cohesion values up to 0.5682 while existing system cohesion value is 0.54072. This result attained by using the BPNN due to it find the best value by analysing every layer and hidden layer in the network. The relationship is set between the classes based on solution of BPNN and it increases the cohesion value of the program. The other parameter such as complexity, coupling, and cost function is reduced. This method provides the better function of the OOP and helps in the software maintenance. The Future work of this method will be based on the Cuckoo search algorithm in the BPNN to improve the performance of this method.

References

[1] N. Jali, D. Greer, and P. Hanna, "Class Responsibility Assignment (CRA) for Use Case Specification to Sequence Diagrams (UC2SD)", In: *Proc. of the 8th Malaysian on Software Engineering Conf.*, pp.13-18, 2014.

[2] G. Glavas and K. Fertalj, "Metaheuristic approach to class responsibility assignment problem", In: *Proc. of the 33rd International Conf. on Information Technology Interfaces*, pp.591-596, 2011.

[3] C. Pinto and L. Benini, "A Novel Object-Oriented Software Cache for Scratchpad-Based Multi-Core Clusters", *Journal of Signal Processing Systems*, Vol.77, No.1-2, pp.77-93, 2014.

[4] G.D. Manioudakis and S.D. Likothanassis, "An Object-Oriented Toolbox for Adaptive Neural Networks Implementation", *International Journal of Artificial Intelligence Tools*, Vol.10, No.3, pp.345-371, 2001.

[5] C.Y. Chong and S.P. Lee, "Automatic clustering constraints derivation from object-oriented software using weighted complex network with graph theory analysis", *Journal of Systems and Software*, Vol.133, pp.28-53, 2017.

[6] A. Shatnawi, A.D. Seriai, H. Sahraoui, and Z. Alshara, "Reverse engineering reusable software components from object-oriented APIs", *Journal of Systems and Software*, Vol.131, pp.442-460, 2017.

[7] A. Parashar and J.K. Chhabra, "Mining software change data stream to predict changeability of classes of object-oriented software system", *Evolving Systems*, Vol.7, No.2, pp.117-128, 2016.

[8] M. Bowman, L.C. Briand, and Y. Labiche, "Solving the class responsibility assignment problem in object-oriented analysis with multi-objective genetic algorithms", *IEEE Transactions on Software Engineering*, Vol.36, No.6, pp.817-837, 2010.

[9] C.L. Simons, I.C. Parmee, and R. Gwynllyw, "Interactive, evolutionary search in upstream object-oriented class design", *IEEE Transactions on Software Engineering*, Vol.36, No.6, pp.798-816, 2010.

[10] J.K. Chhabra, "Harmony search based remodularization for object-oriented software systems", *Computer Languages, Systems & Structures*, Vol.47, pp.153-169, 2017.

[11] J. de AG Saraiva, M.S. De França, S.C. Soares, J. C. L. Fernando Filho, and R. M. de Souza, "Classifying metrics for assessing object-oriented software maintainability: A family of metrics catalogs", *Journal of Systems and Software*, Vol.103, pp.85-101, 2015.

[12] V. Gupta and J.K. Chhabra, "Dynamic cohesion measures for object-oriented software", *Journal of Systems Architecture*, Vol.57, No.4, pp.452-462, 2011.

[13] L. Kumar and S.K. Rath, "Hybrid functional link artificial neural network approach for predicting maintainability of object-oriented software", *Journal of Systems and Software*, Vol.121, pp.170-190, 2016.

[14] J.K. Chhabra, "Improving package structure of object-oriented software using multi-objective optimization and weighted class connections", *Journal of King Saud University-Computer and Information Sciences,* Vol.29, No.3, pp.349-364, 2017.
[15] G. Kakarontzas, E. Constantinou, A. Ampatzoglou, and I. Stamelos, "Layer assessment of object-oriented software: A metric facilitating white-box reuse", *Journal of Systems and Software*, Vol.86, No.2, pp.349-366, 2013.
[16] Z. Javidi, R. Akbari, and O. Bushehrian, "Semi-automatic object-oriented software design using metaheuristic algorithms", In: *Proc. of the 2nd International Conference on Swarm Intelligence and Evolutionary Computation*, pp.123-128, 2017.
[17] H. Masoud and S. Jalili, "A clustering-based model for class responsibility assignment problem in object-oriented analysis", *Journal of Systems and Software*, Vol.93, pp.110-131, 2014.