International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 3 Issue 9, September- 2014

Analysis of Code Clone Detection using Object Oriented System and Neural Network

Balwinder Kumar
M.Tech-Information Tech.
BBSBEC, Fatehgarh Sahib
PTU, Punjab (India)

Dr. Satwinder Singh
A.P CSE & IT Department
BBSBEC, Fatehgarh Sahib
PTU, Punjab (India)

Pavitdeep Singh
Associate VP
Royal bank of Scotland
Gurgaon, Haryana (India)

Abstract- **To improve the quality and maintenance of the software projects, the prediction and identification of code clones are necessary. But in a large software system the detection of clones is very cumbersome process. In this paper two types of investigation are performed. The first is to predict the clone in a given dataset and second to identify the types of clones. Two neural network models are used. A single layer perceptron is used to predict the clone in the given dataset and a feed forward neural network is used to identify the types of clones. Object oriented design metrics like DIT, NOC, WMC, LOCM, various types of methods and variables etc which are generated from some .net project using software quality assurance tool (SQAT) are used as independent variables. The feed forward neural network gives more accurate result with faster training and testing of the neural network.**

Keywords- Clone prediction, clone identification, Software metrics, Single layer perceptron, Feed forward neural network, activation functions.

I. INTRODUCTION

Software code clones are the similar or identical parts of the source code which may be inserted either by mistake or knowingly. But the presence of these code clones may decrease the software quality and hence increase the maintenance cost. So the detection and removal of code clones are necessary in the software products. There are various techniques are proposed over the last decade for the identification and prediction of code clone. But code clone detection using object oriented system was considered the best than other techniques. The prediction and identification of code clone in software product can improve the quality of the software. In this research work we use the object oriented system with neural network for the identification and prediction of clones. The software metrics dataset was gathered using a software quality assurance tool (ASQT) from some online or offline projects made in .net with C# technology. The prediction of code clone using software metrics which has non linear and complex factors. In this research, code clone was chosen as the dependent variable and object-oriented metrics as the independent variables. Prediction of code clone is performed with the help of Single Layer Perceptron and Identification of clones is done with Multilayer FFNN [8]. In this research, multiple independent input neurons with one or more hidden layers and single dependent output neuron is used and a multilayer perceptron with sigmoid and softmax activation function are used to calculate the output neuron. The output neuron predict whether the software projects classes has the code clone or not and of which types.

II. LITERATURE SURVEY

Roy CK and Cordy [1], "Comparison and evaluation of clone detection techniques and tools"

In this paper Roy and Cordy [1] did comparison of different techniques of clone detection such as textual approach, lexical approach, semantic approach and metric based approach and also comparing and evaluating clone detection tools such as Duploc, simian and NICAD. They proposed that NICAD tool is the best among all others. Moreover they explain four category of clone viz-Type-1, Type-2, Type-3 and Type-4. Roy and cordy [1] use the clone detection process- preprocessing, transformation, match detection, formatting, filtering and aggregation. They also provide examples of how one might use the results of this study to choose the most appropriate clone detection tool or technique in the context of a particular set of goals and constraints.

Roy CK, Cordy [3], "A survey on software clone detection research"

In this paper, Roy and Cordy [3] survey the state of the art in clone detection research. Firstly they describe the clone terms commonly used in the literature along with their corresponding mappings to the commonly used clone types. Secondly they give the review of existing clone detection approaches and techniques. Applications of clone detection research to other domains of software engineering and in the same time how other domain can assist clone detection research have also been pointed out. Finally, they conclude by pointing out several open problems related to clone detection research which has the answer whether is it is solved, partially solved or unsolved.

Ettore, Michel and Bruno [4], "Advanced clone-analysis to support object-oriented system refactoring"

In this paper ettore et.al [4] has presented an advanced clone analysis for system refactoring. They present a novel approach for computer aided clone-based object-oriented system refactoring. This approach extracts clone differences and their interpretation in terms of programming language entities. It also focuses on the study of contextual dependencies of cloned methods. The clone analysis has been applied to JDK 1.1.5, a large scale system of 150 KLOC.

They use new clone comparison algorithm along with a novel difference interpretation and classification. The second aspect of the analysis, context dependence, provides useful input on the cost of transferring common or particular code fragments between their original classes and other classes. It can therefore guide the choice of refactoring approaches to apply to specific clusters of clones.

The novel clone analysis provides a solid base for the development of computer-aided refactoring environments. Complete automatic approaches are also possible but they are costly and less flexibility to the user. On the other hand, manual refactoring can be difficult as hundreds of clusters are present in a system. Assisted refactoring combines the strengths of both approaches by computing all the information necessary for refactoring and allowing the user to concentrate on the refactoring decisions.

Filip VR and Serge D [5], "Evaluating Clone Detection Techniques"

In this paper Filip VR and Serge D [5], compares three representative clone detection techniques i.e simple line matching, parameterized matching, and metric fingerprints by means of five small to medium cases and analyses the differences between the reported matches. Based on these experiments, they conclude that-

- Simple line matching **(string-based techniques)** is best suited for a first crude overview of the duplicated code.
- Metric fingerprints (**parse-tree based techniques**) work best in combination with a refactoring tool that is able to remove duplicated subroutines.
- Parameterized matching **(token-based approaches)** works best in combination with more fine-grained refactoring tools that work on the statement level.

Kodhai. A and Kanmani.[6], " Clone Detection using Textual and Metric Analysis to figure out all Types of Clones"

In this paper Kodhai.A and Kanmani.[6] present a novel code clone detection method using textual analysis and metrics-based approach. It has also been implemented as a tool using Java. The tool efficiently and accurately detects type-1, type-2, type-3 and type-4 clones found in source codes at method level in JAVA open source code projects.

In this research Kodhai et.al [6] extends their previous work. In previous research only 7 metrics are used for computation but at this time they use a set of 12 existing method level metrics. These metric values are computed for each identified methods in the system. By implementing these metrics the accuracy of clone detection has been improved. Type-1 clone method found from line by line comparison whereas type-2 clone found with the comparison of templates. Matching template with exact code[4] are declared as type-3 and method whose output was same rather than they being written completely different called as type-4. Limitations of this research are:

- Language dependent because its detection only valid on JAVA open source project.

Rubala S and Kodhai [7], "Code Clones Detection in Websites using Hybrid Approach"

Code cloning is not only found in software products and programming language but it can also be found in web applications. Web based applications used the commerce functionality in web sites. Scripting languages such as ASP, JSP, PHP etc are used in the development of web sites in which code duplication practice usually involved in making of several web pages. As web application contain hyperlinked web pages which make it a large source input file. A lightweight approach is used to detect code clone which is made with textual and metrics value computation method called hybrid technique. The proposed method is implemented as a tool in .NET programming language. A set of 7 existing function level metrics are used for the detection of all types of clone functions in web application. The proposed tool gives its evaluated result in precision and recall parameter which then further compared with the other existing tool called eMetrics. eMetrics tool is a web applications that uses ASP technology which measures the size of a web application to different granularity levels and then selects similar programmer-defined functions written in JavaScript or VBScript as potential cloned functions. The result of comparison showed that the value of precision and recall in term of percentage with the proposed tool using .NET gives higher value with accuracy than the eMetrics tool. Limitation of this paper-

- Problem with working on larger and even more complex system.

Sanjay Dubey, Ajay Rana [8], "Maintainability Prediction of object Oriented Software System by Using ANN Approach"

In this paper Sanjay dubey et.al [8], uses the MLP with software metrics for the prediction of software maintainability which is an imperative attribute of software quality. However the prediction of this attribute is a tedious process. Therefore various methodologies are proposed so far to estimate the maintainability of software. But neural network is one of the sophisticated techniques which have the immense prediction capability. They choose maintenance effort as dependent variable and object-oriented metrics as the independent variables. Prediction of maintainability is performed by Multi Layer Perceptron (MLP) neural network model. The results obtained from the current study are also compared with other models and analyzed with regression analysis. These result shows that the MLP model is more useful than the previous one.

Thwin and Quah[9[, "Application of Neural Networks for Software Quality Prediction Using Object-Oriented Metrics"

Thwin and Queh [9] present a neural network modeling technique along with regression analysis called GRNN to improve the quality of software products. In this paper, ward neural network and General regression neural network are used. First on predicting the number of defects in a class and the second on predicting the number of lines changed per class.

Ward neural network is a back propagation network with different activation functions. They are applied to hidden layer slabs to detect different features in a pattern processed through a network to lead to better prediction. We use a Gaussian function in one hidden slab to detect features in the mid-range of the data and a Gaussian complement in another hidden slab to detect features for the upper and lower extremes of the data. Thus combining the two feature sets in the output layer leads to a better prediction.

On the other hand, General Regression Neural Network (GRNN) is a memory-based network that provides estimates of continuous variables and converges to the underlying (linear or nonlinear) regression surface. This is a one-pass learning algorithm with a highly parallel structure. Even with sparse data in a multidimensional measurement space; the algorithm provides smooth transitions from one observed value to another.

Object-oriented design metrics concerning inheritance related measures, complexity measures, cohesion measures, coupling measures and memory allocation measures are used as the independent variables. GRNN network model is found to predict more accurately than Ward network model.

Gayathri and M.Punithavalli [10], "Comparison and Evaluation on Metrics based Approach for Detecting Code Clone"

In this paper Gayathri et.al [10], detect the different types of clones using different algorithm like textual analysis, metric based distance algorithm and mapping algorithm. The detected clones are-extract clone, renamed clone, gapped cloned and semantic clone. They used clone detection and metrics to evaluate quality. Then they discuss several approaches used in clone detection. In textual analysis all types of codes fragments are detected. Each metrics values are stored in a particular database. The input source is identified using metrics and the similarities of code are detected. In scenario based algorithm the two methods are used –token based and line based. A token-based technique will be more expensive in terms of time and space complexity than a line-based one because a source line contains several tokens.

Metric based clone detection approach uses the metric based distance algorithm. Then they compared different types of approach using different algorithm and calculate their metrics, speed, cost and quality.

J. Mayrand, Ettore M. Merlo[11], " Experiment on the Automatic Detection of Function Clones in a Software System Using Metrics"

J. Mayrand et.al [11] present metric based technique to detect functional clones automatically from source code of any language. They developed a "DATRIX" tool framework which is a source code analyzer [11] which converts the source code into some Intermediate Representation Language. Out of which only control flow metrics and data flow metrics were selected because they provide internal characteristics information about functions. For automatic detection, this proposed approach experimented on 2-Telecommunication monitoring system in which for finding function clones, 4 points of comparison is performed which are-name of function, layout of function, expression in function and control flow.

Basis on these points of comparison, any two functions can be considered as equal, similar or distinct. Under each point of comparison except function name, there are no. of metrics whose description states that what need has to extract as given in [11] and for all these metrics a certain delta or threshold value was assigned which decide whether that function lie in that category or not. Eight levels on ordinal scale marked up for identifying clone in projects which helps in determining the programming style to was good or bad adopted in system, as level increase from level 1 to level 8. It indicates that clone in system are less and in worst case at level 1. The copies of function in system are more. Experimentation on Project A and Project B involves testing for finding each pair of function at each level from level 1 to level 8 and came to conclusion that Level 8, Distinct Control Flow captured all the pairs of function that were not considered clones which means that in both projects, more than 96% of the relations between functions are not clone relations [11].This paper also enlighten the method which can be used in controlling the level of clone in software system which includes measurement program, design principles, clone monitoring and clone reduction.

Pavitdeep and Satwinder. S [12], "Tool for Generating Code Metrics for C# Source Code using Abstract Syntax Tree Technique"

In this paper Pavitdeep et.al [12], developed a tool "Quality assurance tool" in dot net framework using C# as programming language. This tool generates the code metrics for C# software projects using the AST technique. This tool works at method and class level metrics. This tool also detects the clones of Type-1 and Type-2.

The proposed design of the tool consists of various steps from C# source code to syntax tree creation. Once the syntax parse is generated it is resolved to using Type system to generate the semantic tree, which is further utilized by the tool to compute various code metrics. The code metric generated at class level (LCOM , No. of Public Variables, No. of Protected Variables, No. of Private Variables, No. of Public Overridden Variables, No. of Protected Overridden Variables, No. of Children (NOC), Depth of Inheritance Tree (DIT), No. of Public Methods, No. of Private Methods, No. of Public Overridden Methods, No. of Protected Overridden Methods, Weighted Methods per Class (WMC)) and Metrics defined at the Method Level (No of Parameters, Cyclomatic Complexity (CC), No of Variables, No of Overloads).

They also compared different types of tools [12] with each other and explaining their merits and demerits of each tool. The tool of software quality assurance predicts and calculated the metrics of modern languages like c# using the technique of abstract syntax tree (AST).

Sandeep Sharawat [13], "Software maintainability prediction using neural networks"

In this paper, Sandeep Sharawat [13] uses the neural network technique for the prediction of maintainability. The object oriented metrics like DIT, NOC, SIZE, WMC, RFC, NOM, LOCM etc are used for the computation of maintenance index which is composite metric that incorporate a number of traditional source code metrics into a single number that indicate relative maintainability. Matlab is used for the implementation of project. The training data are collected from Li-henry dataset. Neural network created and various training algorithm are applied on the tested data to find the minimized error like trainlm, traingdm, trainscg, trainr, trainrp etc. Trainlm is found to be the best algorithm for the prediction of maintenance index or software maintainability.

K.K aggarwal , Yogesh Singh[14], " Application of artificial neural network for predicting maintainability using object oriented metrics".

In this paper, k.k Aggarwal et.al [14] uses ANN technique to predict the maintenance effort of the classes. This method is rarely applied in this area. The inputs to the network were all the domain metrics P1, P2 and P3 [14]. The network was trained using the back propagation algorithm. Table II [14] shows the best architecture, which was experimentally determined. The model is trained using training and test data sets and evaluated on validation data set. Table IV [14] shows the MARE, MRE, r and p-value results of ANN model evaluated on validation data. The correlation of the predicted change and the observed change is represented by the coefficient of correlation (r). The significant level of a validation is indicated by a p-value. A commonly accepted p value is 0.05. For validate data sets, the percentage error smaller than 10 percent, 27 percent and 55 percent is shown in Table V [14].

The result [14] shows that the independent variables appear to be useful in predicting maintenance effort. The ANN model demonstrated that they were able to estimate maintenance effort within 30 percent of the actual maintenance effort in more than 72 percent of the classes in the validate set, and with a MARE of 0.265. Thus ANNs have shown their ability to provide an adequate model for predicting maintenance effort.

III. PROPOSED WORK

A. *Dataset*

The dataset for the analysis of metric based clone detection has been collected from any offline or online software projects which are developed in some dot net technologies like C#. The following object oriented metrics are generated using Software quality assurance tool (SQAT) from the online Credit Risk Management System software.

TABLE I. METRIC SUITED

Metrics	**Definition**
DIT	DIT is maximum depth of the inheritance hierarchy of each class.
NOC	It counts the number of direct children for each class.
WMC	The WMC is a count of sum of complexities of all methods in a class. Consider a class K1, with methods M1,… Mn that are defined in the class. Let C1,….Cn be the complexity of the methods. $WMC = \sum_{i=1}^{n} C_i$
LOCM	It is used to measures the lack of cohesion of a class.
LOCM HS	Henderson-Sellers revises the LCOM metric to normalize it for the number of methods and variables that are present in the class
NOM (no of method)	The NOM metrics count the number of local method of a class.
PUF public Factor)	It is the ratio of sum of public (method & variable) and protected (method & variable) to the sum of public (method & variable), private (method & variable) and protected (method & variable).
VF (Visible Factor)	1-PUF

B. *Object Oriented Metrics*

The object oriented metrics are very useful in the prediction of software clones. The definitions of each object oriented metrics used in the dataset are described in Table I.

C. *Feed Forward Neural Network (FFNN)*

Feed forward neural network trained with back propagation learning algorithm are the most popular neural network. They are applied to variety of problems. A FFNN consists of neurons that are ordered into layers (input, hidden and output layers) as shown in fig 2. In this research, clone is chosen as the dependent variable and 6-object-oriented metrics as the independent variables. Prediction of code clone is performed with the help of single layer perceptron using purelin activation function and identification of clone is performed using multilayer perceptron with logsig and softmax activation function. The output neuron predict whether the classes have the code clone or not and of which types.

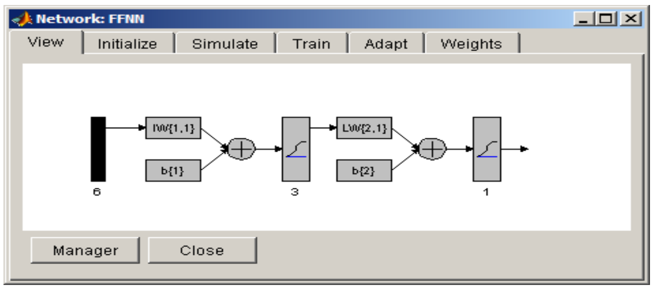

Fig. 1. Architecture of multilayer feed forward neural network

D. *Clone prediction and identification process*

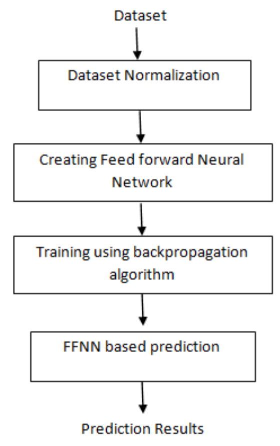
Dataset
Dataset Normalization
Creating Feed forward Neural Network
Training using backpropagation algorithm
FFNN based prediction
Prediction Results

Fig. 2. FFNN Prediction Process

E. *Neural Network Modelling*

Neural Network Toolbox (NNTOOL) in MATLAB software provides tools for designing, implementing, visualizing, and simulating neural networks. Using this we create Perceptron and Multilayer perceptron neural networks with six input neuron and one output neuron and trained them using the dataset. Neural network are used for applications where formal analysis would be difficult or impossible.

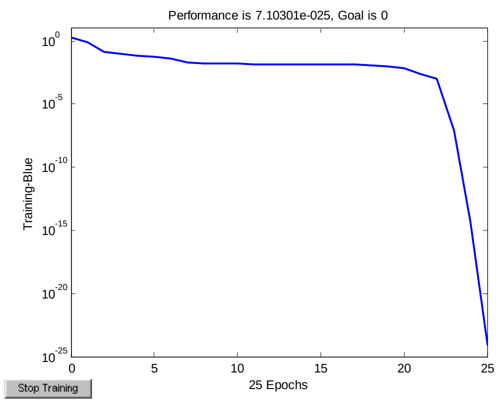

Fig. 3. Training with multilayer FFNN

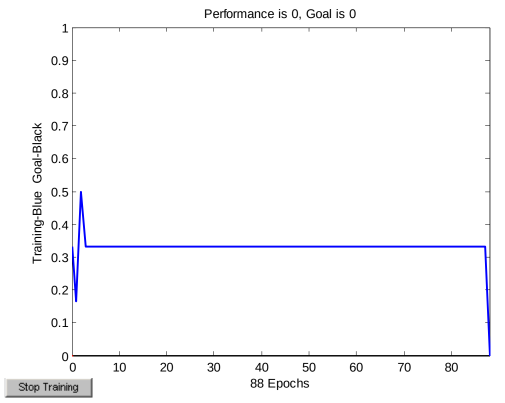

Fig. 4. Training with single layer FFNN

F. *Prediction of Clone*

For the prediction of code clone, data is collected and normalized. Then a single layer perceptron neural network using hardlim activation function is created and trained with the given dataset. After training, the network is tested using the testing dataset and it predicts whether the software project classes have the code clones or not.

1) *Hardlim:* hardlim is a neural network transfer function which calculates a layer's output from its net input.

$$Y=\text{hardlim}(n) \quad = 1 \text{ if } n \geq 0$$
$$= 0 \text{ otherwise}$$

G. *Identification of Clone Type*

For the identification of code clone, data is collected and normalized. Then a Multilayer perceptron neural network using logsig and softmax activation function is created and trained with the given dataset. After training, the network is tested using the testing dataset and it clearly identify that the class have which types of clone that means weather tyep1, type 2 or no clone.

1) *Log sigmoid Activation function:* Logsig is a non-linear transfer function used to train neural networks. It is simply called sigmoid function. It calculates the layer's output from the net input. It has the "s" shaped curve.

$$S(t) = \frac{1}{1+e^{-t}}.$$

2) *Softmax activation Function:* if you want the outputs of a network to be interpretable as posterior probabilities for a categorical target variable, it is highly desirable for those outputs to lie between zero and one and to sum to one. The purpose of the softmax activation function is to enforce these constraints on the outputs. Let the net input to each output unit be q_i, i=1,...,c, where c is the number of categories. Then the softmax output p_i is:

```
         Exp (q_i)
p_i = ------------
        c
        sum Exp(q_j)
        j=1
```

IV. SIMULATION OF NETWORKS

Now we test and validate the neural network using normalized testing dataset to get properly weighted & biased neural networks. Now using these trained networks we supply five test dataset with known target values and record the output as clone type and existence of clone in the particular classes.

TABLE II. PREDICTION OF CLONE

Test case	Input Metrics (Normalized dataset)	Output / Target	Clone Prediction
1	[0.0417, 1, 0.0102, 0, 0,0]	1	Yes
2	[0.0833,1,0.0000,0,0,1]	0	No
3	[0.2292,1,0.0102,1, 0.00017, 0.75]	1	Yes
4	[0.1042,1,0.0000,0,0,1]	0	No
5	[0.1458,1,0.0000,0,0,1]	1	Yes

TABLE III. IDENTIFICATION OF CLONE TYPE

Test case	Input Metrics (Normalized dataset)	Output / Target	Clone Prediction	Clone Type
1	[0.0417, 1, 0.0102, 0, 0,0]	1	Yes	Type 1
2	[0.0833,1,0.0000,0, 0,1]	0	No	No clone
3	[0.2292,1 ,0.0102,1, 0.00017, 0.75]	1	Yes	Type 2
4	[0.1042,1,0.0000,0, 0,1]	0	No	No clone
5	[0.1458,1,0.0000,0 0,1]	1	Yes	Type 1

V. CONCLUSION

We observed from Table II that a single layer feed forward neural network with linear activation function is much suitable for the prediction of code clone using the object oriented system dataset and from table III, we observed that a multilayer feed forward neural network with log sigmoid and softmax activation function is best suitable for the identification of categorical output that means the identification of no clone, type 1 and type 2 clones from the dataset which is generated using the software quality assurance tool (SQAT) from some online or offline software project developed using some .net technologies such as C#. The metrics generated by SQAT tool from the online Credit Risk Management Software is best suited for the prediction and identification of code clones. This study can be further extended for the prediction and identification of type 3 and type 4 clones. The overall performance of the dataset generated by the tool is good and hence the analysis of code clone detection using object oriented system and neural network is much more effective technique as compared to regression analysis and other clone detection techniques.

Feed forward neural network has the immense capability of data prediction. Prediction and identification of clones using FFNN can decrease the maintenance cost and hence increase the software quality which leads to faster software development.

REFERENCES

[1] Roy CK, Cordy JR, Koschke R. Comparison and evaluation of clone detection techniques and tools: A qualitative approach. Science of Computer Programming 2009; 74:470–495.
[2] Cláudio Nogueira Sant'Anna. "On the Reuse and Maintenance of Aspect- Oriented Software: An Assessment Framework ", August 2003.
[3] Roy CK, Cordy JR. A survey on software clone detection research. TR 2007-541, Queen's School of Computing, 2007; 115
[4] Bellon, R. Koschke, G. Antoniol, J. Krinke, E. Merlo, Comparison and evaluation of clone detection tools, Transactions on Software Engineering 33 (9) (2007) 577–591.
[5] Filip Van Rysselberghe, Serge Demeyer. Evaluating Clone Detection Techniques. In Proceedings of the International Workshop on Evolution of Large Scale Industrial Applications (ELISA'03), 12pp., Amsterdam, The Netherlands, Sept 2003.
[6] Kodhai. E, Perumal. A, and Kanmani. S,"Clone Detection using Textual and Metric Analysis to figure out all Types of Clones", in IJCCIS, Vol2. No1. ISSN: 0976–1349 July-Dec 2010.
[7] Rubala Sivakumar, Kodhai. E,*"Code Clones Detection in Websites using Hybrid Approach", in* IJCA (0975 – 888) Volume 48–No.13, June 2012.
[8] Yajnaseni Dash, Sanjay Kumar Dubey, Ajay Rana, "Maintainability Prediction of Object Oriented Software System by Using Artificial Neural Network Approach", in IJSCE ISSN: 2231-2307, Volume-2, Issue-2, May 2012.
[9] Mie Mie Thet Thwin, Tong-Seng Quah "Application of Neural Networks for Software quality Prediction Using Object-Oriented Metrics" in ICSM, ISSN: 1063-6773 Sep-2003.
[10] D. Gayathri Devi, Dr.M.Punithavalli "Comparison and Evaluation on Metrics based Approach for Detecting Code Clone" in IJCSE, ISSN: 0976-5166 Vol. 2 No. 5 Oct-Nov 2011 page no- 750.
[11] Jean Mayrand, Claude Leblanc, Ettore M. Merlo " Experiment on the Automatic Detectionof Function Clones in a Software System Using Metrics" ICMS 1996 1063-6773
[12] Pavitdeep Singh, Prof. Satwinder Singh, Prof. Jatinder Kaur "Tool for Generating Code Metrics for C# Source Code using Abstract Syntax Tree Technique" ACM Sigsoft, software engineering notes-vol-3 sep-2013 pages1-6
[13] Sandeep Sharawat : Software maintainability prediction using Neural Network, Vol. 2, Issue 2,Mar-Apr 2012, pp.750-755
[14] K.K Aggarwal, yogesh singh et.al : Application of artificial neural network for predicting maintainability using object oriented metrics, world academy of science, engineering and technology, 22, 2006.
[15] E. Burd, J. Bailey, Evaluating clone detection tools for use during preventative maintenance, in: Proceedings of the 2nd IEEE International Workshop on Source Code Analysis and Manipulation, SCAM 2002, 2002, pp. 36–43.
[16] R. Koschke, R. Falke, P. Frenzel, Clone detection using abstract syntax suffix trees, in: Proceedings of the 13th Working conference on Reverse Engineering, WCRE 2006, 2006, pp. 253–262
[17] iClones Website [Online]. Last Accessed August-2014. URLhttp://softwareclones.org/iclones.php.
[18] F. Brooks, The mythical man-month: essays on software engineering. Addison Wesley Pub. Co, 1975
[19] Yogita sharma, rajesh bhatia, "Hybrid technique for object oriented software clone detection'- a thesis submitted in june-2011, thapar university, patiala.