International Journal of Engineering & Technology, 7 (4) (2018) 5922-5925

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET
doi: 10.14419/ijet.v7i4.22672
Research paper



A deep learning approach for cost effective learning of defective prone modules

Satya Srinivas Maddipati [1] *, Malladi Srinivas [2]

[1] Research Scholar K L EF
[2] Professor, CSE Dept., KLEF
**Corresponding author E-mail: maddipativas@gmail.com*

Abstract

Software Defect Prediction is a cost effective problem, in which the cost of majority class (Non defective) is low compared with the cost of minority class (Defective). Learning from imbalanced data bias the classifier towards majority class. In this paper we are proposing a deep learning approach for classifying Imbalanced and Cost effective data. We applied Principle Component Analysis for feature selection and then constructed a classifier using Adaptive Neuro Fuzzy Inference System. The performance of the classifier was evaluated using AuC measures. We observed the performance of the classifier was improved compared with neural networks.

Keywords: *Software Defect Prediction; Imbalanced Data; Principle Component Analysis; Adaptive Neuro Fuzzy Inference System.*

1. Introduction

Software defect prediction is a promising approach in the software development process aiming to increase software quality with reduction on implementation cost.

1.1. Issues in software defect prediction

1.1.1. Relationship between defects and attributes

Akiyama introduced an equation to relate number of defects with Lines of code (LOC). He derived the equation defects(N)=4.86+0.018*LOC. But LOC is not sufficient to identify the defects. Later in 1976 MCCube found complexity metrics the effects the defectiveness of a software module. He derived an equation for Cyclometric complexity of a program V(G)= E – V + 2. In 1977, Halsted introduced Halsted complexity measures. He derived an equation Defects(D)=$E^{2/3}$/3000. Chidamber & Kemerer introduced object oriented metrics for software defect prediction. The metrics are weighted methods per class (WMC), Depth of inheritance tree (DIT), Number of children (NOC), Coupling between objects (CBO) and Response for a class (RFC).

1.1.2. No general framework available

Some researchers proposed frameworks for implementation of software defect prediction. The applicability of the framework is evaluated using public dataset repositories. The learning methodologies are J48 decision tree, Naïve Bayes, Support Vector machines and Neural Networks. The performance of the learning methodologies are evaluated using the measures AuC (Area under ROC Curves) and Recall.

1.1.3. Economies in software defect prediction

Quality assurance is a costly part of software development process. 30% - 40% of Software development cost was assigns to quality assurance and defect removal. Banthia et al.[2] proposed a cost evaluation framework to actually evaluate whether the fault prediction is useful and if yes, when it is beneficial to use.

1.1.4. No standard measures for performance evaluation

There exist no standard criterion for comparing the models of defective prediction. The general measures for performance evaluation are precision, f-measure, recall and Area under ROC Curve. Software Defect Prediction datasets are imbalanced in nature. Srinivas et al proposed ROC curve for imbalanced datasets[3]

1.1.5. Class imbalance problem

Software defect datasets are imbalance in nature. The imbalance ratio is 1:9. The majority class in non defective and minority class is defective. Learning from imbalanced data, biases the classifier towards majority class. It will increase the misclassification rate and decreases the performance of classifier. Various techniques are proposed for balancing the data such under sampling, over sampling, Bagging, Boosting etc..

2. Literature survey

Mingxiz L proposed two stage cost sensitive learning for identifying defects in software. They incorporated cost information in feature selection and classifier. Three cost sensitive feature selection algorithms namely, Cost sensitive variance score, cost sensitive laplacian score and Cost sensitive constraint score were developed [1]. Manjula C proposed Deep Belief Network with L1-Regularization based optimization for software defect prediction [4]. Jaroslaw H et. al conducted defect prediction based on non tweaked DePress configuration and proposed quality assurance strategy[5]. Yun Z et. al studied combined classifier for cross project defect prediction which transfers a prediction model trained

Copyright © 2018 Satya Srinivas Maddipati, Malladi Srinivas. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

using data from one project to another[6]. Satya Srinivas M et al proposed Adaptive Neuro Fuzzy Inference system for Software Defect Prediction. In this work, Sugeno Fuzzy Inference system was generated using subtractive clustering and trained using hybrid learning rule. The performance of the classifier was evaluated using AuC (Area under ROC Curve) values [7].
In the Research work [8], A novel approach utilizing metric learning for software defect prediction was proposed. They applied Maximally Collapsing Metric Learning technique to learn a Mahalanobis distance metric for use in classifiers. The effectiveness of the proposed technique was evaluated using PROMISE dataset repositories, by comparing with k-nearest neighbours. Ming Li et. al studied sample based methods for software defect prediction. There are three methods for selecting a sample: random sampling with conventional machine learners, active sampling with active semi supervised learner and random sampling with semi supervised learner[9]. In the work[10], Static analysis technique in feature selection was applied to improve the performance of Defect Prediction. In the work [11], Adaptive Neuro Fuzzy Inference System was constructed to learn from imbalanced data. Divya T et. al proposed Weighted Least Square Twin Support Vector Machines for learning software defect prone modules [12]. Romi S et. al proposed Neural network parameter optimization based on Genetic algorithm for Software defect prediction [13].

3. Methodology

In our work, we used Deep learning approach for constructing classifier to predict software defects. Figure 1 shows the methodology used in our work.

1) Load Dataset: The Dataset, we used in this experimentation,was downloaded from PROMISE Dataset repository. There are variants in the dataset such as PC1, KC1, KC2, AR1 etc. Each dataset contains different number of features. The datasets are imbalanced in nature.

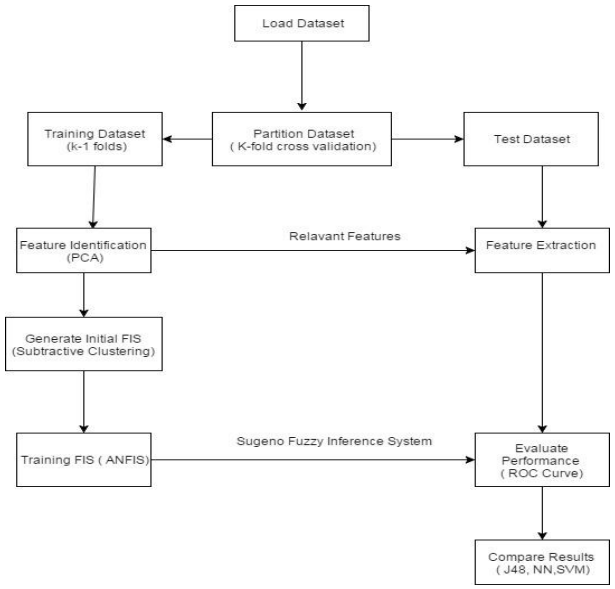


Fig. 1: Methodology.

2) Partition Dataset: In this experimentation, we partitioned the dataset using k-fold cross validation. In K-fold cross validation, the dataset was divided into K-folds. Among K-folds, (K-1) folds are used for training, one fold of dataset was used for testing. These folds were selected purely random manner.
3) Feature identification

High dimensionality problem in a real world datasets can be reduced by applying feature selection algorithms. Feature selection algorithms are divided into five categories[14].

Filter-Based Feature Ranking Methods: Filter-based feature ranking methods evaluate each feature separately by assigning individual feature a score according to an indicator.
Wrapper-Based Feature Subset Evaluation: Wrapper-based methods evaluate the merit of a feature subset with predetermined classifiers and evaluation measures.
Clustering-Based Feature Selection Methods: Clustering based feature methods clusters the group of features. Then select the number of features with highest information gain. Extraction-based Feature Selection Methods: Features are selected based on discriminant analysis. In the work [15], Principle component analysis was applied as a feature selection method, then a classifier was constructed using Neural networks, the performance of classfier was evaluated with & without Principle Component Analysis.
In our work, We used Extraction based Feature Selection methods. We applied Principle Component Analysis on PC4 dataset which was downloaded from PROMISE dataset repository. The total number of features are 38 along with target feature. We extracted 29 features based on high rotational values, as relevant features for constructing classifier. Figure 2 show the rotations of features on first two principle components.

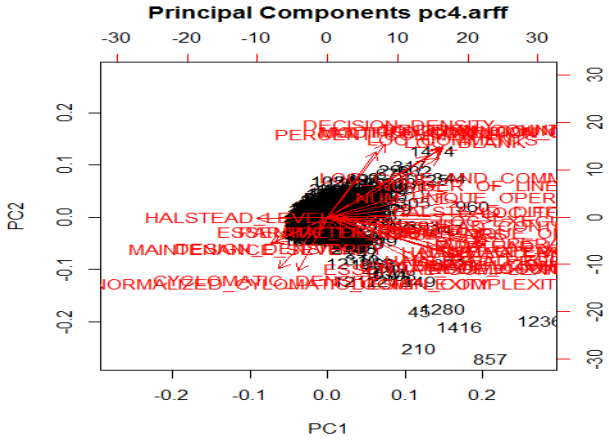


Fig. 2: Principle Component Analysis for SDP.

3.1. Generate initial FIS

Sugeno Fuzzy Inference System: A Sugeno Fuzzy Inference System generates the fuzzy rules of the form If Input1= x and Input2=y then output is Z=a*x+b*y+c. We used Subtractive Clustering method to generate Initial Fuzzy Inference System.
Subtractive Clustering is one of popular unsupervised clustering method by increasing and using new data samples. New data samples consists of some centers of the core, and these cores are gained by constructing neighborhood graph [16].Figure 3 shows Subtractive clusters for Software Defect Prediction.

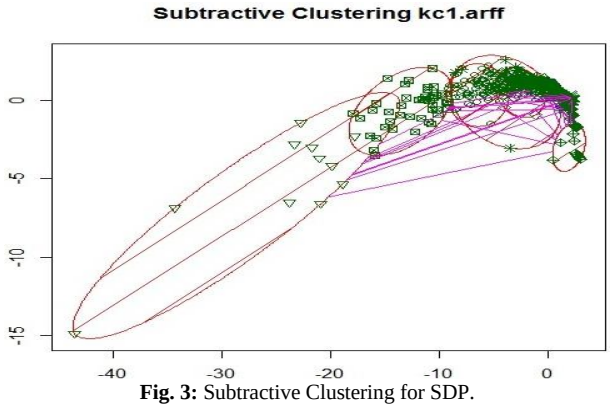


Fig. 3: Subtractive Clustering for SDP.

3.2. Training FIS

In our work, we trained Initial Fuzzy Inference System with Adapative Neuro Fuzzy Inference System, which is a five layered

architecture. First layer is an adaptive layer that computes the membership values of input variables. Second layer is a fixed layer that computes the firing strength of input variables using And method. Third layer is a fixed layer that computes normalized weight of the firing rule. Fourth layer is an adaptive layer that fits the consequent parameters of firing rule. Fifth layer is a fixed layer that computes overall sum of input variables. Figure 8 show the Fuzzy Inference System after trained by ANFIS.

4. Results

We experimented on software defect prediction using 4 datasets downloaded from PROMISE data repository to identify defectives in software modules. Neural networks are one of the methodologies for best fitting nonlinear relationships between attributes. Cost Effectiveness was applied to neural networks to improve the performance of classifier. The weight of class j instance is defined by

$W(j)=C(j)*(N/\sum C(i).\ N_i)$

W (1) =8*(2109/ (8*326+2*1783))

W (1) =8*2109/8782

W (1) =1.92

W (2) =2*2109/8782

W (2) =0.48

We conducted an experimentation using MAT lab on prediction of software defect(s). In MAT lab, there are two toolboxes Neural Network and Neuro Fuzzy tool box which implements Neural networks and Fuzzy Inference System. In Neuro Fuzzy tool box, there are three parts. In Data section, training data was loaded. In FIS section, Initial Sugeno Fuzzy Inference system is generated by using either Grid based or subtractive clustering method. In this paper, we applied Subtractive clustering which takes four parameters Range of Influence (0.2.5), Squash factor (1.5), Accept ratio(0.5) & Reject Ratio(0.15). In training section, a Sugeno Fuzzy Inference System was modeled using ANFIS algorithm. We applied the firing strength of the rules as 1.92 & 0.48 for minority class & Majority class respectively. In testing section, Fuzzy Inference System was tested with unseen data. The Receiver Operating Characteristics (ROC) was plotted against true positive rate with false positive rate

In Table 1, We presented AuC values of Cost sensitive neural networks and ANFIS applied on 4 datasets.

Table 1: Auc Values for Software Defect Prediction

Dataset	CS NN (l:2, n:15)	CS NN (l:2,n:30)	CS NN (l:2,n:60)	CS NN (l:3,n=90)	ANFIS
kc1	0.5	0.5	0.5	0.5	0.8482
kc2	0.64958	0.66451	0.660324	0.53430	0.8998
cm1	0.5	0.5	0.5	0.5	0.8904
pc1	0.58976	0.548134	0.637584	0.56115	0.8416

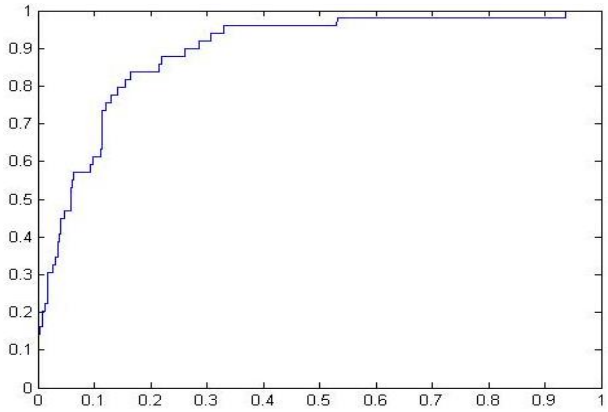


Fig. 4: ROC Curve Cm1 Dataset.

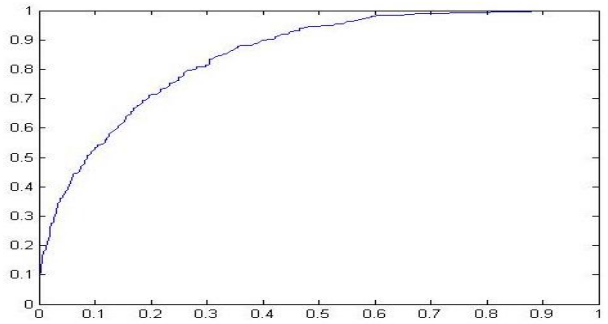


Fig. 5: ROC Curve Kc1 Dataset.

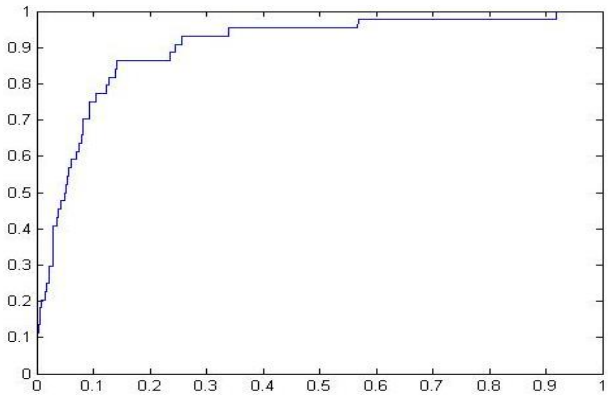


Fig. 6: ROC Curve Kc2 Dataset.

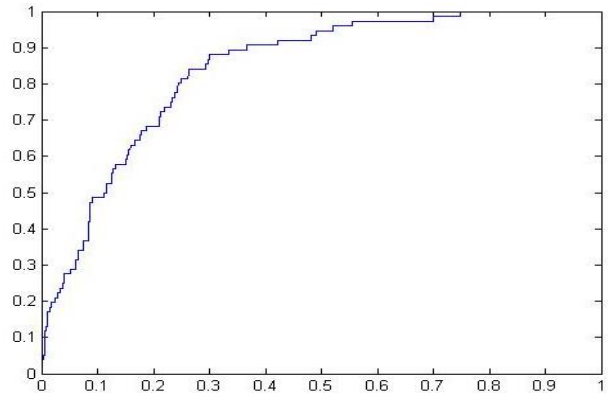


Fig. 7: ROC Curve Pc1 Dataset.

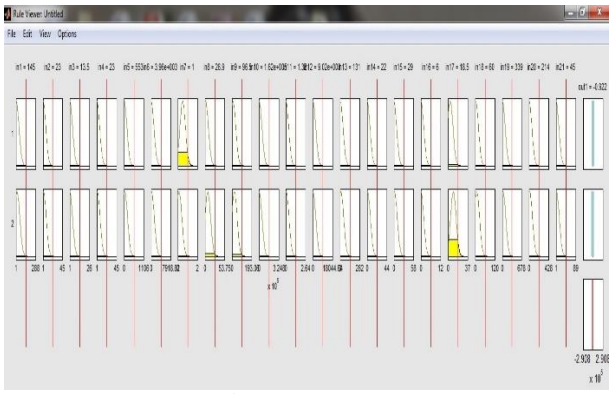


Fig. 8: Fuzzy Inference System trained by ANFIS.

5. Conclusion

Identifying Defective prone modules plays a major role in reduction of software development cost. These Defective prone modules can be identified by using deep learning approach. We applied Principle Component Analysis as a feature selection method, and then applied Adaptive Neuro Fuzzy Inference System to Construct the classifier. The performance of the classifier was evaluated using AuC values. Principle Component Analysis reduces the features and ANFIS improves the performance of the Classifier.

References

[1] Mingxia Liu,Linsong Miao & Daoqiang Zhang," Two-Stage Cost-Sensitive Learning for Software Defect Prediction", IEEE Transactions on Reliability,63(2),2014. https://doi.org/10.1109/TR.2014.2316951.

[2] Bathia D, Gupta A" A framework to assess the effectiveness of fault-prediction techniques for quality assurance" In: 7th CSI International Conference on Software Engineering. Pune; 2013. p. 40-49.

[3] Satya Srinivas M, G Pradeepini, A Yesubabu," Software Defect Prediction using Adaptive Neuro Fuzzy Inference System",International Journal of Applied Engineering & Research, 13(1), pp:394-397.

[4] Manjula C, Lilly Florence," Software Defect Prediction using Deep Belief Network with L1-Regularization Based Optimization", IJARCS,9(1), pp:864-870. https://doi.org/10.26483/ijarcs.v9i1.5476.

[5] Jaroslaw Hryszko,Lech Madeyski," Cost Effectiveness of Software Defect Prediction in an industrial project",Foundations of Computing and Decision Sciencs,43(1). https://doi.org/10.1515/fcds-2018-0002.

[6] Yun ZHANG, David LO, Xin XIA, Jianling SUN," Combined classifier for cross-project defect prediction: an extended empirical study", Frontiers Computer Science,2018, pp:1-17.

[7] Satya Srinivas Maddipati, G Pradeepini, A Yesubabu," Software Defect Prediction using Adaptive Neuro Fuzzy Inference System", International Journal of Applied Engineering Research", 13(1), pp:394-397.

[8] Linh Nhat Chu,"Metric learning for Software Defect Prediction",Monash University,(2015).

[9] Ming Li Hongyu zhang, Rongxin Wu, Zhi-hua Zhou," Sample based Software Defect Prediction with active and semi supervised learning", Autom Softw Eng,2011.

[10] Hao Tang, Tian Lan, Dan hao & Lu Zhang,"Enhancing Defect Prediction with Static Defect Analysis" , Proceedings of the 7th Asia-Pacific Symposium on Internetware,PP:43-51

[11] Satya Srinivas M, G Pradeepini," Class Imbalance Learning of Defective Prone Modules using Adaptive Neuro Fuzzy Inference System", International Journal of Pure and Applied Mathematics,118(5), pp:739-744.

[12] Divya Tomar, Sonali Agarwal," Prediction of Defective Software Modules Using Class Imbalance Learning",Applied Computational Intelligence and Soft Computing,2016. https://doi.org/10.1155/2016/7658207.

[13] Romi Satria Wahono, N Suryana & Sabrina Ahmad," Neural Network Parameter Optimization Based on Genetic Algorithm for Software Defect Prediction",Advanced Science Letters,20,pp:10-12. https://doi.org/10.1166/asl.2014.5641.

[14] Zhou Xu,Jin Liu,Zijiang Yang, Gerge An & Xiangyang Jia," The Impact of Feature Selection on Defect Prediction Performance: An Emipirical Comparision", 2016 IEEE 27th International Symposium on Software Reliability Engineering,2016.

[15] Satya Srinivas M, A Yesubabu, G Pradeepini," Feature Selection Based Neural Networks for Software Defect Prediction", IOSR Journal of Computer Engineering,18(6),2016.

[16] Lei Gu," A novel subtractive clustering method by increasing and using new data samples", 2016 IEEE International Conference of Online Analysis and Computing Science,2016. https://doi.org/10.1109/ICOACS.2016.7563053.